aws

Guia do desenvolvedor de banco de dados

Amazon Redshift



Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon Redshift: Guia do desenvolvedor de banco de dados

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

As marcas comerciais e imagens de marcas da Amazon não podem ser usadas no contexto de nenhum produto ou serviço que não seja da Amazon, nem de qualquer maneira que possa gerar confusão entre os clientes ou que deprecie ou desprestigie a Amazon. Todas as outras marcas comerciais que não pertencem à Amazon pertencem a seus respectivos proprietários, que podem ou não ser afiliados, patrocinados pela Amazon ou ter conexão com ela.

Table of Contents

..... xxxiii
Introdução 1
Pré-requisitos para usar o Amazon Redshift 1
Arquitetura do Amazon Redshift 2
Arquitetura de data warehouse 2
desempenho 6
Armazenamento colunar 9
Gerenciamento do workload 11
Usar Amazon Redshift com outros serviços 13
Banco de dados de exemplo 14
Práticas recomendadas 22
Realizar uma prova de conceito 23
Etapa 1: Definir o escopo da POC 23
Etapa 2: Iniciar o Amazon Redshift 25
Etapa 3: Carregar os dados 26
Etapa 4: Analisar os dados 27
Etapa 5: Otimizar 29
Melhores práticas para projetar tabelas 30
Escolha a melhor chave de classificação 31
Escolha o melhor estilo de distribuição 32
Uso da compactação automática 33
Definição das limitações 34
Uso do menor tamanho possível de coluna 34
Uso dos tipos de dados de data/hora para colunas de data 34
Melhores práticas para carregamento de dados 34
Saiba como carregar dados 35
Uso de um comando COPY para carregar dados 35
Uso de um único comando COPY 35
Carregar arquivos de dados 36
Compactar arquivos de dados 37
Verificação de arquivos de dados antes de depois de um carregamento 37
Uso de uma inserção de múltiplas linhas 37
Uso de uma inserção em massa 38
Carregamento de dados por ordem de chave de classificação 38

Carregamento de dados em blocos sequenciais ... 39
Uso de tabelas de séries temporais ... 39
Agendamento em torno de janelas de manutenção ... 40
Melhores práticas para projetar consultas ... 40
Trabalhar com recomendações do Advisor ... 43
Como o Advisor funciona ... 43
Otimizar arquiteturas de compartilhamento de dados ... 44
Regiões do Amazon Redshift em que o Advisor é oferecido ... 44
Visualizar recomendações do Advisor ... 45
Recomendações do Advisor ... 46
Tutoriais ... 63
Otimização automática de banco de dados ... 65
Alocação de recursos de computação ... 67
Gerenciar recursos extras de computação para operações autônomas ... 67
Controle de custos de operações autônomas que usam recursos extras de computação ... 71
Faturamento por operações autônomas ... 72
Faturamento de operações autônomas para clusters provisionados ... 72
Faturamento de operações autônomas para grupos de trabalho sem servidor ... 73
Métricas de uso de operações autônomas ... 74
Otimização automática de tabelas ... 75
Habilitar, desabilitar e monitorar a otimização automática de tabelas ... 76
Habilitar a otimização automática de tabelas ... 76
Remover a otimização automática de tabelas de uma tabela ... 76
Monitorar a otimização automática de tabelas ... 77
Gerenciar exclusões de workload do Autonomics ... 78
Gerenciar recursos da lista de negação no console do Amazon Redshift ... 79
Adicionar recursos de lista de negação ... 79
Remover recursos de lista de negação ... 79
Compactação de colunas ... 79
Codificações de compactação ... 81
Teste de codificações de compactação ... 92
Distribuição de dados ... 99
Conceitos de distribuição de dados ... 100
Estilos de distribuição ... 101
Visualização dos estilos de distribuição ... 103
Avaliação dos padrões de consulta ... 104

Designação de estilos de distribuição ... 105
Avaliação do plano de consulta ... 106
Exemplo de plano de consulta ... 109
Exemplos de distribuição ... 113
Chaves de classificação ... 116
Classificação do layout de dados multidimensionais ... 117
Chave de classificação composta ... 118
Chave de classificação intercalada ... 119
Restrições de tabela ... 120
Carregamento de dados ... 122
Carregar tabelas com COPY ... 126
Credenciais e permissões de acesso ... 128
Preparação de dados de entrada ... 130
Carregar dados do Amazon S3 ... 131
Carregar dados do Amazon EMR ... 145
Carregamento de dados de hosts remotos ... 151
Carregar do Amazon DynamoDB ... 160
Como verificar se os dados foram carregados corretamente ... 164
Validação de dados de entrada ... 164
Carregamento de tabelas com compactação automática ... 165
Otimização para tabelas estreitas ... 168
Carregamento de valores padrão de coluna ... 168
Solução de problemas de carregamento de dados ... 169
Criar uma integração de eventos do S3 para copiar arquivos automaticamente ... 177
Pré-requisitos ... 178
Criar uma integração de eventos do S3 ... 183
Criar e monitorar um COPY JOB ... 185
Considerações ... 187
Regiões com suporte ... 187
Considerações ao restaurar um destino de integração de eventos do S3 ... 189
Carregar tabelas com DML ... 191
Atualização e inserção ... 191
Execução de uma cópia profunda ... 200
Análise de tabelas ... 204
Análise automática ... 204
Análise de novos dados da tabela ... 205

Histórico do comando ANALYZE ... 210
Vacuum de tabelas ... 211
Classificação automática de tabela ... 212
Exclusão de vacuum automática ... 213
Frequência de VACUUM ... 214
Estágio do classificação e estágio de mesclagem ... 214
Limite de vacuum ... 215
Tipos de vacuum ... 215
Minimizar tempos de limpeza ... 215
Gerenciamento de operações de gravação simultâneas ... 224
Níveis de isolamento ... 226
Operações de gravação e de leitura/gravação ... 231
Exemplos de gravações simultâneas ... 233
Solução de problemas ... 236
Tutorial: Carregar dados do Amazon S3 ... 239
Pré-requisitos ... 239
Visão geral ... 240
Etapa 1: criar um cluster ... 240
Etapa 2: Fazer download dos arquivos de dados ... 242
Etapa 3: Carregar arquivos para um bucket do Amazon S3 ... 243
Etapa 4: Criar as tabelas de exemplo ... 244
Etapa 5: Executar os comandos COPY ... 247
Etapa 6: Vacuum e análise do banco de dados ... 265
Etapa 7: limpar os recursos ... 265
Resumo ... 266
Descarregamento de dados ... 268
Descarregar dados do Amazon S3 ... 268
Descarregamento de arquivos de dados criptografados ... 272
Descarregamento de dados em formato delimitado ou de largura fixa ... 274
Recarregamento de dados descarregados ... 275
Funções definidas pelo usuário ... 277
Segurança e permissões de UDFs ... 278
Evitar conflitos na nomeação da UDF ... 278
Sobrecarga de nomes de função ... 279
Evitar conflitos com funções integradas do Amazon Redshift ... 279
UDFs escalares de SQL ... 280

Exemplo ... 280
UDFs escalares do Python ... 281
Exemplo ... 282
Tipos de dados da UDF Python ... 283
Suporte da linguagem Python ... 284
Restrições ... 289
Registro em log de erros e alertas ... 290
UDFs escalares do Lambda ... 291
Gerenciar a segurança e as permissões de UDFs do Lambda ... 293
Configurar o parâmetro de autorização para UDFs do Lambda ... 294
Usar a interface JSON entre o Amazon Redshift e o Lambda ... 296
Exemplos de casos de uso referentes a UDFs ... 299
Criação de procedimentos armazenados ... 301
Visão geral do procedimento armazenado ... 301
Nomeação de procedimentos armazenados ... 305
Segurança e privilégios ... 306
Retorno de um conjunto de resultados ... 307
Gerenciamento de transações ... 310
Retenção de erros ... 323
Registro em log de procedimentos armazenados ... 331
Limitações ... 332
Referência da linguagem PL/pgSQL ... 333
Convenções de referência da PL/pgSQL ... 334
Estrutura da PL/pgSQL ... 334
Instruções da PL/pgSQL compatíveis ... 340
Visualizações materializadas ... 357
Visões materializadas ... 360
Regravação automática de consulta para usar visões materializadas ... 361
Observações de uso ... 362
Como funciona a regravação automática de consulta de visões materializadas ... 363
Limitações ... 363
Visões materializadas em tabelas externas de data lake ... 364
Limitações ... 365
Visualizações materializadas armazenadas como tabelas do Apache Iceberg ... 366
Como as visualizações materializadas do Iceberg funcionam ... 367
Requisitos da tabela de origem ... 367

Interoperabilidade entre mecanismos ... 367
Atualização incremental ... 368
Atualização simultânea ... 368
Compactação e manutenção de tabelas ... 369
Considerações sobre segurança ... 369
Monitoramento ... 371
Limitações ... 372
Atualizar uma visualização materializada ... 373
Visões materializadas aninhadas ... 374
Limitações ... 376
Atualizar automaticamente uma visão materializada ... 377
Visões materializadas automatizadas ... 378
Escopo e considerações de SQL para visões materializadas automatizadas ... 381
Visualizações materializadas automatizadas ... 381
Faturamento para visões materializadas automatizadas ... 382
Recursos adicionais ... 382
Como usar uma função definida pelo usuário (UDF) em uma visão materializada ... 382
Como fazer referência a uma UDF em uma visão materializada ... 382
Ingestão de streaming para uma visão materializada ... 385
Como os dados fluem de um serviço de streaming para o Redshift ... 385
Práticas recomendadas de análise de dados para melhorar a performance ... 385
Comportamento de ingestão de streaming e tipos de dados ... 386
Conceitos básicos da ingestão de streaming do Amazon Kinesis Data Streams ... 394
Conceitos básicos da ingestão de streaming de origens do Apache Kafka ... 399
Tutorial de ingestão de streaming de dados de estação de veículos elétricos usando o Kinesis ... 414
Visualizações do Catálogo de Dados ... 418
Pré-requisitos ... 419
Exemplo completo ... 420
Considerações ... 421
Consultar dados espaciais ... 422
Tutorial: Usando funções SQL espaciais ... 426
Pré-requisitos ... 426
Etapa 1: Criar tabelas e carregar dados de teste ... 427
Etapa 2: Consultar dados espaciais ... 430
Etapa 3: Limpar os recursos ... 434

Carregar um shapefile ... 434
Terminologia ... 436
Caixa delimitadora ... 436
Validade geométrica ... 436
Simplicidade geométrica ... 439
H3 ... 441
Considerações ... 441
Consultar dados com consultas federadas ... 443
Conceitos básicos do uso de consultas federadas no PostgreSQL ... 444
Conceitos básicos do uso de consultas federadas no PostgreSQL com o CloudFormation ... 445
Iniciar uma pilha do CloudFormation para consultas federadas do Redshift ... 446
Consultar dados do esquema externo ... 448
Conceitos básicos do uso de consultas federadas no MySQL ... 448
Criar um segredo e uma função do IAM ... 450
Pré-requisitos ... 450
Exemplo de uso de uma consulta federada ... 453
Exemplo de uso de uma consulta federada no PostgreSQL ... 453
Exemplo de uso de um nome com maiúsculas e minúsculas ... 455
Exemplo de uso de uma consulta federada com MySQL ... 457
Diferenças dos tipos de dados ... 458
Considerações ... 462
Versões compatíveis de bancos de dados federados ... 464
Permissões federadas ... 465
Principais conceitos ... 465
Benefícios ... 466
Casos de uso ... 467
Considerações ... 467
Pré-requisitos ... 471
Requisitos de política do IAM ... 471
Configuração da aplicação ... 472
Pré-requisitos do Lake Formation ... 478
Configuração de rede ... 480
Privilégios Connect ... 480
Integração ... 481
Registro do cluster do Redshift ... 481
Registro do namespace do Redshift sem servidor ... 484

Habilitar a propagação de identidades do Centro de Identidade do AWS IAM 486
ALTER USER SET GLOBAL IDENTITY 489
Consultar catálogos 490
Pré-requisitos 491
Consultar bancos de dados federados 491
Segurança e controle de acesso 491
Controle de acesso 492
Concessão/revogação 492
Controle de acesso refinado 494
Exemplos 498
Pré-requisitos 498
Na instância do catálogo, conceda o perfil sys:secadmin ao perfil do IAM correspondente. . 498
Configurar o ambiente 500
Configurar permissões em nível de coluna (CLP) 501
Criar políticas de segurança por linha (RLS) 501
Criar políticas de mascaramento dinâmico de dados (DDM) 502
Auditar políticas disponíveis/aplicadas 503
Testar padrões de acesso 503
Alterar políticas 504
Políticas de desvinculação e descarte 504
Limpeza: desative políticas de RLS e descarte. 505
Redefinir a identidade global 505
Configuração do usuário federado 505
Configurações de usuário que são consistentes em todos os warehouses do Amazon Redshift 506
Gerenciamento de conexão e configuração 507
Desabilitar a propagação do Centro de Identidade do IAM 507
Clusters provisionados 507
Namespaces 508
Cancelar o registro de AWS Glue Data Catalog 509
Amazon Redshift Spectrum 512
Visão geral do Amazon Redshift Spectrum 513
Regiões do Amazon Redshift Spectrum 514
Limitações do Amazon Redshift Spectrum 514
Conceitos básicos do Amazon Redshift Spectrum 516
Pré-requisitos 516

CloudFormation ... 517
Conceitos básicos detalhados do Amazon Redshift Spectrum ... 517
Etapa 1. Criar um perfil do IAM ... 518
Etapa 2: Associar uma função do IAM ao cluster ... 522
Etapa 3: Criar um esquema e uma tabela externos ... 523
Etapa 4: Consultar os dados no Amazon S3 ... 524
Iniciar a pilha do CloudFormation e consultar seus dados ... 527
Consultar Tabelas do S3 ... 531
Pré-requisitos ... 531
Consultar Tabelas do S3 por meio do Amazon Redshift ... 532
Políticas do IAM do Amazon Redshift Spectrum ... 539
Permissões do Amazon S3 ... 540
Permissões entre contas do Amazon S3 ... 541
Conceder ou restringir acesso usando o Redshift Spectrum ... 542
Permissões mínimas ... 543
Encadeamento de funções do IAM ... 544
Acesso a dados do AWS Glue ... 545
Redshift Spectrum e Lake Formation ... 554
Usar filtros de dados para segurança em nível de linha e de célula ... 556
Arquivos de dados para consultas no Amazon Redshift Spectrum ... 557
Formatos de dados do Redshift Spectrum ... 557
Tipos de compactação do Redshift Spectrum ... 559
Encriptação do Redshift Spectrum ... 560
Esquemas externos ... 560
Trabalhar com catálogos externos ... 562
Tabelas externas ... 567
Pseudocolunas ... 569
Dividir as tabelas externas do Redshift Spectrum ... 570
Mapeamento para colunas do ORC ... 576
Criar tabelas externas para dados gerenciados no Hudi ... 579
Criar tabelas externas para dados do Delta Lake ... 581
Usar tabelas do Apache Iceberg ... 583
Considerações ao usar tabelas do Apache Iceberg ... 584
Tipos de dados compatíveis ... 586
Fazer referência a tabelas do Iceberg ... 590
Gravar em tabelas do Apache Iceberg ... 595

Recursos do Iceberg v3 ... 618
Performance de consulta do Amazon Redshift Spectrum ... 624
Opções de tratamento de dados ... 628
Realizar subconsultas correlacionadas ... 629
Métricas ... 630
Observação sobre clusters RG provisionados ... 630
Solução de problemas de consulta ... 632
Número de tentativas excedido ... 632
Acesso limitado ... 633
Limite de recursos excedido ... 634
Nenhuma linha foi retornada para uma tabela particionada ... 634
Erro de falta de autorização ... 635
Formatos de dados incompatíveis ... 635
Erro de sintaxe ao usar a DDL do Hive no Amazon Redshift ... 636
Permissão para criar tabelas temporárias ... 636
Intervalo inválido ... 636
Número da versão do Parquet inválida ... 636
Campo obrigatório ausente nas informações da partição externa ... 637
Tutorial: Consultar dados aninhados com o Amazon Redshift Spectrum ... 638
Visão geral ... 638
Etapa 1: Crie uma tabela externa que contém dados aninhados ... 639
Etapa 2: Consultar os dados aninhados no Amazon S3 com extensões SQL ... 640
Casos de uso de dados aninhados ... 644
Limitações de dados aninhados (visualização) ... 647
Serializar JSON aninhado complexo ... 649
Esboços do HyperLogLog ... 652
Considerações ... 653
Limitações ... 654
Exemplos ... 655
Exemplo: retornar cardinalidade em uma subconsulta ... 655
Exemplo: retorna um tipo HLLSKETCH de esboços combinados em uma subconsulta ... 656
Exemplo: retorna um esboço do HyperLogLog da combinação de vários esboços ... 656
Exemplo: gerar esboços do HyperLogLog sobre dados do S3 usando tabelas externas ... 657
Consultas entre bancos de dados ... 661
Considerações ... 664
Limitações ... 664

Exemplos 665
Usar consultas entre bancos de dados com o editor de consultas 671
Compatibilidade com o Apache Iceberg 673
Regiões onde a compatibilidade com o Apache Iceberg está disponível 673
Considerações e limitações ao usar catálogos do Amazon Redshift no AWS Glue Data Catalog 674
Requisitos de política do IAM 676
Registrar clusters e namespaces 676
Cancelar o registro de clusters e namespaces 678
Grupos de trabalho gerenciados 679
Consultar catálogos registrados no AWS Glue Data Catalog 679
Consulta de catálogos federados do IRC do Glue 681
Visão geral da arquitetura 681
Configuração entre contas 682
Configuração entre contas 683
Práticas recomendadas e limitações 687
Compartilhamento de dados 688
Casos de uso de compartilhamento de dados 688
Casos de uso de gravações em vários warehouses 689
Compartilhamento de dados em diferentes níveis 689
Gerenciamento de consistência do compartilhamento de dados 690
Considerações 690
Considerações gerais sobre compartilhamento de dados 691
Considerações sobre leituras e gravações de compartilhamento de dados 693
Considerações sobre o compartilhamento de dados na restauração sem servidor 695
Considerações sobre compartilhamento de dados com tabelas de data lake 696
Considerações sobre compartilhamento de dados com o AWS Lake Formation 696
Considerações sobre compartilhamento de dados com o AWS Data Exchange 697
Permissões que você pode conceder a unidades de compartilhamento de dados 699
Instruções SQL aceitas para gravações de compartilhamento de dados em consumidores . 700
Instruções SQL não aceitas para gravações de compartilhamento de dados em consumidores 701
disponíveis Regiões da AWS 702
Introdução 706
Conceitos básicos do compartilhamento de dados somente leitura no console 706
Conceitos básicos do compartilhamento de dados somente leitura com a interface SQL 726

Conceitos básicos das gravações em vários data warehouses ... 775
Conceitos básicos do CloudFormation ... 811
Tipos de unidade de compartilhamento de dados ... 817
Unidades de compartilhamento de dados comuns ... 818
AWS Data ExchangeUnidades de compartilhamento de dados do ... 819
AWS Lake FormationUnidades de compartilhamento de dados gerenciadas pelo ... 823
Status da unidade de compartilhamento de dados ... 825
Gerenciar o acesso a operações de API de compartilhamento de dados com políticas do IAM ... 827
Conectar-se a bancos de dados de consumidor ... 828
Acesso a dados compartilhados ... 829
Acessar metadados referentes a objetos compartilhados ... 829
Integração do compartilhamento de dados do Amazon Redshift com ferramentas de business intelligence ... 830
Monitoramento e auditoria do compartilhamento de dados ... 831
Integrar o compartilhamento de dados do Amazon Redshift com o AWS CloudTrail ... 832
Dados semiestruturados no Amazon Redshift ... 834
Práticas recomendadas para trabalhar com dados semiestruturados ... 834
Conceitos para uso do tipo de dados SUPER ... 835
Considerações sobre o uso de dados do tipo SUPER ... 837
PartiQL para Amazon Redshift ... 837
O que é PartiQL? ... 837
PartiQL no Amazon Redshift ... 838
Carregamento de dados semiestruturados ... 838
Usar JSON_PARSE para inserir dados JSON ... 839
Usar COPY para carregar dados JSON ... 839
Descarregamento de dados semiestruturados ... 845
Consultar dados semiestruturados ... 847
Navegação ... 847
Desaninhar consultas ... 848
Transformar colunas em linhas de objetos ... 850
Digitação dinâmica ... 851
Consulta sem distinção entre maiúsculas e minúsculas ... 854
Semântica lax ... 855
Order by (Ordenar por) ... 856
Operadores e funções ... 856

Operadores aritméticos ... 857
Funções aritméticas ... 857
Funções de array ... 858
Comportamento do agrupamento ... 860
Funções de informação ... 861
Funções de objetos ... 861
Funções de string ... 862
Configurações SUPER ... 864
Modos lax e estrito para SUPER ... 864
Acessar campos JSON com letras maiúsculas ou mistas ... 864
Opções de análise ... 865
Limitações ... 866
Tipo de dados SUPER com visões materializadas ... 869
Acelerar consultas do PartiQL ... 870
Fragmentar dados semiestruturados em colunas SUPER com visões materializadas ... 870
Criar colunas escalares do Amazon Redshift a partir de dados fragmentados ... 872
Exemplos ... 874
Carregamento de dados semiestruturados ... 874
Consultar dados semiestruturados aninhados ... 881
Uso de `enable_case_sensitive_super_attribute` com dados semiestruturados ... 882
Filtrar dados semiestruturados ... 883
Desaninhamento de dados semiestruturados ... 884
Desaninhamento de matrizes aninhadas ... 884
Usar dados semiestruturados em subconsultas ... 885
Agregação de consultas usando dados semiestruturados ... 886
Usar dados semiestruturados em visões materializadas ... 886
Usar PIVOT e UNPIVOT com dados semiestruturados ... 886
Machine learning ... 889
Visão geral do Machine Learning ... 891
Como o machine learning pode resolver seu problema ... 892
Termos e conceitos do Amazon Redshift ML ... 894
Machine Learning para iniciantes e especialistas ... 895
Custos para usar o Amazon Redshift ML ... 898
Custos para usar o Amazon Redshift ML com o SageMaker AI ... 898
Custos para usar o Amazon Redshift ML com o Amazon Bedrock ... 898
Conceitos básicos: Amazon Redshift ML ... 900

Configuração administrativa ... 900
Usar a explicabilidade do modelo com o Amazon Redshift ML ... 906
Métricas de probabilidade do Amazon Redshift ML ... 906
Tutoriais para o Amazon Redshift ML ... 909
Tutorial: Criar modelos de rotatividade de clientes ... 910
Tutorial: Como construir modelos de agrupamento K-means ... 919
Tutorial: Como criar modelos de classificação multiclasse ... 929
Tutorial: Como construir modelos XGBoost ... 939
Tutorial: Como criar modelos de regressão ... 945
Tutorial: Como construir modelos de regressão com o aprendizado linear ... 958
Tutorial: Como criar modelos de classificação com aprendizado linear ... 967
Integração do Amazon Redshift ML com o Amazon Bedrock ... 989
Criar uma função do IAM ... 989
Criar um modelo externo ... 991
Usar um modelo externo ... 992
Engenharia rápida ... 994
Ajustar o desempenho da consulta ... 996
Processamento de consulta ... 996
Planejamento de consulta e fluxo de trabalho de execução ... 997
Criar e interpretar um plano de consulta ... 999
Revisar as etapas do plano de consulta ... 1008
Fatores que afetam a performance da consulta ... 1010
Análise e aprimoramento de consultas ... 1012
Fluxo de trabalho da análise de consulta ... 1012
Revisar alertas da consulta ... 1013
Analisar o plano de consulta ... 1016
Analisar o resumo da consulta ... 1017
Aprimoramento da consulta ... 1024
Consultas de diagnóstico para ajuste da consulta ... 1030
Solução de problemas de consulta ... 1034
Falhas na conexão ... 1035
Consultas travadas ... 1035
A consulta leva muito tempo ... 1037
Falha do carregamento ... 1038
O carregamento leva muito tempo ... 1039
Os dados de carregamento estão incorretos ... 1039

Como configurar o parâmetro JDBC para o tamanho da busca 1040
Gerenciamento do workload 1042
Alternar o modo do WLM 1044
Modificar a configuração do WLM 1044
Migrar do WLM manual para o WLM automático 1044
WLM automático 1046
Prioridade 1048
Modo de escalabilidade da simultaneidade 1048
User groups (Grupos de usuários) 1048
Funções de usuário 1048
Grupos de consultas 1048
Curingas 1049
Regras de monitoramento de consulta 1049
Verificação do WLM automático 1049
Prioridade da consulta 1050
WLM manual 1055
Modo de escalabilidade da simultaneidade 1057
Nível de simultaneidade 1057
User groups (Grupos de usuários) 1059
Funções de usuário 1060
Grupos de consultas 1060
Curingas 1060
Porcentagem de memória do WLM a ser usada 1060
Tempo limite do WLM 1061
Regras de monitoramento de consulta 1062
Salto na fila de consultas do WLM 1062
Tutorial: Configuração de filas de WLM manual 1066
Escalabilidade simultânea. 1085
Recursos de escalabilidade de simultaneidade 1086
Limitações para a escalabilidade de simultaneidade 1086
Regiões para escalabilidade de simultaneidade 1087
Candidatos da escalabilidade da simultaneidade 1089
Configurar filas da escalabilidade da simultaneidade 1052
Monitoramento da escalabilidade da simultaneidade 1090
Visualizações do sistema de escalabilidade da simultaneidade 1091
Aceleração de consulta breve 1092

Tempo máximo de execução da SQA ... 1093
Monitoramento da SQA ... 1093
Regras de atribuição de fila do WLM ... 1094
Exemplo das atribuições de fila ... 1097
Atribuir consultas a filas ... 1099
Atribuir consultas a filas com base em perfis de usuário ... 1099
Atribuir consultas a filas com base em grupos de usuários ... 1100
Atribuir uma consulta a um grupo de consultas ... 1100
Atribuir consultas à fila de superusuários ... 1101
Propriedades dinâmicas e estáticas ... 1102
Alocação de memória dinâmica do WLM ... 1103
Exemplo do WLM dinâmico ... 1104
Regras de monitoramento de consulta ... 1107
Definir uma regra do monitor de consulta ... 1108
Métricas de monitoramento de consultas para o Amazon Redshift provisionado ... 1110
Métricas de monitoramento de consultas para o Amazon Redshift Serverless ... 1114
Modelos de regras de monitoramento de consulta ... 1116
Tabelas de sistema e visualizações para regras de monitoramento de consultas ... 1118
Tabelas de sistema e visualizações do WLM ... 1118
IDs da classe de serviço do WLM ... 1120
Segurança do banco de dados ... 1121
Visão geral da segurança do Amazon Redshift ... 1122
Permissões de usuário padrão do banco de dados ... 1123
Superusuários ... 1124
Usuários ... 1125
Criar, alterar e excluir usuários ... 1125
Bloqueio de usuário ... 1126
Grupos ... 1128
Criar, alterar e excluir grupos ... 1128
Exemplo para controlar acesso de usuário e grupo ... 1129
Esquemas ... 1130
Caminho de pesquisa ... 1131
Criar, alterar e excluir esquemas ... 1131
Permissões ... 1132
Controle de acesso com base em função ... 1132
Hierarquia de funções ... 1133

Atribuição de função ... 1134
Funções definidas pelo sistema do Amazon Redshift ... 1135
Permissões do sistema ... 1139
Permissões de objetos do banco de dados ... 1152
ALTER DEFAULT PRIVILEGES para o RBAC ... 1152
Considerações sobre o uso de funções ... 1152
Gerenciamento de funções ... 1153
Tutorial: Criar funções e consultar com o RBAC ... 1154
Segurança por linha ... 1173
Utilização de políticas de RLS em instruções SQL ... 1174
Combinar várias políticas por usuário ... 1175
Propriedade e gerenciamento da política de RLS ... 1177
Objetos e princípios dependentes de políticas ... 1178
Considerações e limitações ... 1180
Práticas recomendadas ... 1185
Exemplo completo ... 1188
Segurança de metadados ... 1193
Mascaramento dinâmico de dados ... 1194
Comandos SQL para políticas de DDM ... 1195
Hierarquia de políticas de DDM ... 1197
Uso de DDM com caminhos do tipo SUPER ... 1199
Mascaramento dinâmico de dados condicional ... 1204
Visualizações do sistema de DDM ... 1205
Considerações ... 1207
Exemplo completo ... 1212
Permissões em escopo ... 1215
Considerações ... 1216
Referência SQL ... 1217
SQL do Amazon Redshift ... 1217
Funções SQL compatíveis no nó de liderança ... 1218
Amazon Redshift e PostgreSQL ... 1220
Uso de SQL ... 1229
Convenções de referência do SQL ... 1229
Elementos básicos ... 1230
Expressões ... 1288
Condições ... 1293

Comandos SQL 1321
ABORT 1326
ALTER DATABASE 1328
ALTER DATASHARE 1335
ALTER DEFAULT PRIVILEGES 1339
ALTER EXTERNAL SCHEMA 1343
ALTER EXTERNAL VIEW 1345
ALTER FUNCTION 1347
ALTER GROUP 1348
ALTER IDENTITY PROVIDER 1349
ALTER MASKING POLICY 1352
ALTER MATERIALIZED VIEW 1353
ALTER RLS POLICY 1361
ALTER ROLE 1362
ALTER PROCEDURE 1363
ALTER SCHEMA 1365
ALTER SYSTEM 1367
ALTER TABLE 1369
ALTER TABLE APPEND 1396
ALTER TEMPLATE 1402
ALTER USER 1405
ANALYZE 1412
ANALYZE COMPRESSION 1415
ATTACH MASKING POLICY 1418
ATTACH RLS POLICY 1420
BEGIN 1422
CALL 1425
CANCEL 1428
CLOSE 1431
COMMENT 1431
COMMIT 1434
COPY 1435
CREATE DATABASE 1539
CREATE DATASHARE 1555
CREATE EXTERNAL FUNCTION 1556
CREATE EXTERNAL MODEL 1568

CREATE EXTERNAL SCHEMA ... 1572
CREATE EXTERNAL TABLE ... 1584
CREATE EXTERNAL VIEW ... 1614
CREATE FUNCTION ... 1615
CREATE GROUP ... 1622
CREATE IDENTITY PROVIDER ... 1623
CREATE LIBRARY ... 1626
CREATE MASKING POLICY ... 1629
CREATE MATERIALIZED VIEW ... 1631
CREATE MODEL ... 1640
CREATE PROCEDURE ... 1671
CREATE RLS POLICY ... 1678
CREATE ROLE ... 1680
CREATE SCHEMA ... 1681
CRIAR TABELA ... 1684
CREATE TABLE AS ... 1709
CREATE TEMPLATE ... 1722
CREATE USER ... 1726
CREATE VIEW ... 1734
DEALLOCATE ... 1739
DECLARE ... 1740
DELETE ... 1746
DESC DATASHARE ... 1749
DESC IDENTITY PROVIDER ... 1751
DETACH MASKING POLICY ... 1752
DETACH RLS POLICY ... 1753
DROP DATABASE ... 1755
DROP DATASHARE ... 1756
DROP EXTERNAL VIEW ... 1758
DROP FUNCTION ... 1759
DROP GROUP ... 1761
DROP IDENTITY PROVIDER ... 1762
DROP LIBRARY ... 1763
DROP MASKING POLICY ... 1763
DROP MODEL ... 1764
DROP MATERIALIZED VIEW ... 1765

DROP PROCEDURE 1767
DROP RLS POLICY 1768
DROP ROLE 1769
DROP SCHEMA 1770
DESCARTAR TABELA 1773
DROP 1777
DROP USER 1778
DROP VIEW 1780
END 1782
EXECUTE 1783
EXPLAIN 1784
FETCH 1793
GRANT 1795
INSERT 1827
INSERT (tabela externa) 1835
LOCK 1837
MERGE 1839
PREPARE 1846
REFRESH MATERIALIZED VIEW 1848
RESET 1854
REVOKE 1855
ROLLBACK 1876
SELECT 1878
SELECT INTO 1960
SET 1961
SET SESSION AUTHORIZATION 1966
SET SESSION CHARACTERISTICS 1967
SHOW 1967
SHOW USER LOCKOUT 1969
SHOW COLUMN GRANTS 1970
SHOW COLUMNS 1972
SHOW CONSTRAINTS 1975
SHOW EXTERNAL TABLE 1978
SHOW DATABASES 1981
SHOW FUNCTIONS 1985
SHOW GRANTS 1987

SHOW MODEL ... 1995
SHOW DATASHARES ... 1998
SHOW PARAMETERS ... 1999
SHOW POLICIES ... 2003
SHOW PROCEDURE ... 2006
SHOW PROCEDURES ... 2007
SHOW SCHEMAS ... 2009
SHOW TABLE ... 2012
SHOW TABLES ... 2014
SHOW TEMPLATE ... 2017
SHOW TEMPLATES ... 2018
SHOW VIEW ... 2020
START TRANSACTION ... 2022
TRUNCATE ... 2022
UNLOAD ... 2024
UPDATE ... 2059
USE ... 2068
VACUUM ... 2078
Referência de funções SQL ... 2086
Função de apenas nó líder ... 2087
Funções agregadas ... 2089
Funções de array ... 2118
Funções agregadas bit-wise ... 2143
Expressões condicionais ... 2151
Funções de formatação de tipo de dados ... 2166
Perfis de data e hora ... 2205
Funções de hash ... 2278
Funções HyperLogLog ... 2288
Funções JSON ... 2294
Funções de machine learning ... 2316
Funções matemáticas ... 2319
Funções de objetos ... 2359
Funções espaciais ... 2370
Funções de string ... 2525
Funções de informação de tipo SUPER ... 2606
Funções VARBYTE ... 2621

Funções de janela ... 2631
Funções de administração do sistema ... 2699
Funções de informação do sistema ... 2710
Palavras reservadas ... 2745
Referência de visualizações e tabelas do sistema ... 2750
Tipos de tabelas e visualizações de sistema ... 2751
Visibilidade de dados em tabelas e visualizações de sistema ... 2752
Filtrar consultas geradas pelo sistema ... 2753
Migrar consultas somente provisionadas para consultas de visualização de monitoramento de SYS ... 2753
Migrar clusters provisionados para o Amazon Redshift sem servidor ... 2753
Atualizar consultas enquanto estiver em um cluster provisionado ... 2754
Melhoria do rastreamento do identificador de consultas usando as exbições de monitoramento SYS ... 2754
Exemplo ... 2755
Consulta de tabela do sistema, processo e IDs de sessão ... 2762
Visualizações SVV de metadados ... 2763
SVV_ACTIVE_CURSORS ... 2765
SVV_ALL_COLUMNS ... 2766
SVV_ALL_SCHEMAS ... 2768
SVV_ALL_TABLES ... 2770
SVV_ALTER_TABLE_RECOMMENDATIONS ... 2772
SVV_ATTACHED_MASKING_POLICY ... 2774
SVV_COLUMNS ... 2776
SVV_COLUMN_PRIVILEGES ... 2779
SVV_COPY_JOB_INTEGRATIONS ... 2781
SVV_DATABASE_PRIVILEGES ... 2782
SVV_DATASHARE_PRIVILEGES ... 2784
SVV_DATASHARES ... 2785
SVV_DATASHARE_CONSUMERS ... 2789
SVV_DATASHARE_OBJECTS ... 2790
SVV_DEFAULT_PRIVILEGES ... 2792
SVV_DISKUSAGE ... 2794
SVV_EXTERNAL_COLUMNS ... 2797
SVV_EXTERNAL_DATABASES ... 2799
SVV_EXTERNAL_PARTITIONS ... 2800

SVV_EXTERNAL_SCHEMAS 2801
SVV_EXTERNAL_TABLES 2802
SVV_FUNCTION_PRIVILEGES 2805
SVV_GEOGRAPHY_COLUMNS 2807
SVV_GEOMETRY_COLUMNS 2808
SVV_IAM_PRIVILEGES 2809
SVV_IDENTITY_PROVIDERS 2811
SVV_INTEGRATION 2813
SVV_INTEGRATION_TABLE_MAPPING 2815
SVV_INTEGRATION_TABLE_STATE 2818
SVV_INTERLEAVED_COLUMNS 2820
SVV_LANGUAGE_PRIVILEGES 2821
SVV_MASKING_POLICY 2823
SVV_ML_MODEL_INFO 2824
SVV_ML_MODEL_PRIVILEGES 2825
SVV_MV_DEPENDENCY 2826
SVV_MV_INFO 2828
SVV_QUERY_INFLIGHT 2830
SVV_QUERY_STATE 2832
SVV_REDSHIFT_COLUMNS 2835
SVV_REDSHIFT_DATABASES 2838
SVV_REDSHIFT_FUNCTIONS 2840
SVV_REDSHIFT_SCHEMA_QUOTA 2842
SVV_REDSHIFT_SCHEMAS 2843
SVV_REDSHIFT_TABLES 2845
SVV_RELATION_PRIVILEGES 2847
SVV_RLS_APPLIED_POLICY 2848
SVV_RLS_ATTACHED_POLICY 2850
SVV_RLS_POLICY 2852
SVV_RLS_RELATION 2853
SVV_ROLE_GRANTS 2855
SVV_ROLES 2856
SVV_SCHEMA_PRIVILEGES 2857
SVV_SCHEMA_QUOTA_STATE 2859
SVV_SYSTEM_PRIVILEGES 2860
SVV_TABLE_INFO 2861

SVV_TABLES ... 2866
SVV_TRANSACTIONS ... 2867
SVV_USER_GRANTS ... 2870
SVV_USER_INFO ... 2871
SVV_VACUUM_PROGRESS ... 2873
SVV_VACUUM_SUMMARY ... 2875
Visualizações de monitoramento de SYS ... 2877
SYS_ANALYZE_COMPRESSION_HISTORY ... 2879
SYS_ANALYZE_HISTORY ... 2882
SYS_APPLIED_MASKING_POLICY_LOG ... 2883
SYS_AUTOMATIC_OPTIMIZATION ... 2885
SYS_AUTO_TABLE_OPTIMIZATION ... 2889
SYS_CHILD_QUERY_TEXT ... 2891
SYS_CONNECTION_LOG ... 2892
SYS_COPY_JOB ... 2896
SYS_COPY_JOB_DETAIL ... 2897
SYS_COPY_JOB_INFO ... 2899
SYS_COPY_REPLACEMENTS ... 2900
SYS_DATASHARE_CHANGE_LOG ... 2902
SYS_DATASHARE_CROSS_REGION_USAGE ... 2905
SYS_DATASHARE_USAGE_CONSUMER ... 2906
SYS_DATASHARE_USAGE_PRODUCER ... 2907
SYS_EXTERNAL_QUERY_DETAIL ... 2909
SYS_EXTERNAL_QUERY_ERROR ... 2913
SYS_EXTRA_COMPUTE_FOR_AUTOMATIC_OPTIMIZATION ... 2915
SYS_INTEGRATION_ACTIVITY ... 2917
SYS_INTEGRATION_TABLE_ACTIVITY ... 2918
SYS_INTEGRATION_TABLE_STATE_CHANGE ... 2920
SYS_LOAD_DETAIL ... 2922
SYS_LUDF_DETAIL ... 2925
SYS_LOAD_ERROR_DETAIL ... 2927
SYS_LOAD_HISTORY ... 2930
SYS_MV_REFRESH_HISTORY ... 2934
SYS_MV_STATE ... 2939
SYS_PROCEDURE_CALL ... 2942
SYS_PROCEDURE_MESSAGES ... 2945

SYS_QUERY_DETAIL 2946
SYS_QUERY_EXPLAIN 2955
SYS_QUERY_HISTORY 2959
SYS_QUERY_TEXT 2972
SYS_REDSHIFT_TEMPLATE 2974
SYS_RESTORE_LOG 2976
SYS_RESTORE_STATE 2979
SYS_SCHEMA_QUOTA_VIOLATIONS 2982
SYS_SERVERLESS_USAGE 2983
SYS_SESSION_HISTORY 2987
SYS_SPATIAL_SIMPLIFY 2988
SYS_STREAM_SCAN_ERRORS 2990
SYS_STREAM_SCAN_STATES 2992
SYS_TRANSACTION_HISTORY 2994
SYS_UDF_LOG 2996
SYS_UNLOAD_DETAIL 2998
SYS_UNLOAD_HISTORY 3000
SYS_USERLOG 3003
SYS_VACUUM_HISTORY 3005
Mapeamento de visualizações do sistema para migrar para visualizações de monitoramento de SYS 3009
SYS_QUERY_HISTORY 3010
SYS_QUERY_DETAIL 3011
SYS_RESTORE_LOG 3012
SYS_RESTORE_STATE 3012
SYS_TRANSACTION_HISTORY 3012
SYS_QUERY_TEXT 3013
SYS_CONNECTION_LOG 3013
SYS_SESSION_HISTORY 3013
SYS_LOAD_DETAIL 3013
SYS_LOAD_HISTORY 3013
SYS_LOAD_ERROR_DETAIL 3014
SYS_UNLOAD_HISTORY 3014
SYS_UNLOAD_DETAIL 3014
SYS_COPY_REPLACEMENTS 3014
SYS_DATASHARE_USAGE_CONSUMER 3014

SYS_DATASHARE_USAGE_PRODUCER ... 3015
SYS_DATASHARE_CROSS_REGION_USAGE ... 3015
SYS_DATASHARE_CHANGE_LOG ... 3015
SYS_EXTERNAL_QUERY_DETAIL ... 3015
SYS_EXTERNAL_QUERY_ERROR ... 3015
SYS_VACUUM_HISTORY ... 3016
SYS_ANALYZE_HISTORY ... 3016
SYS_ANALYZE_COMPRESSION_HISTORY ... 3016
SYS_MV_REFRESH_HISTORY ... 3016
SYS_MV_STATE ... 3016
SYS_PROCEDURE_CALL ... 3017
SYS_PROCEDURE_MESSAGES ... 3017
SYS_UDF_LOG ... 3017
SYS_USERLOG ... 3017
SYS_SCHEMA_QUOTA_VIOLATIONS ... 3017
SYS_SPATIAL_SIMPLIFY ... 3017
Monitoramento do sistema (somente provisionado) ... 3018
Visualizações STL de registro em log ... 3018
Tabelas STV para dados de snapshot ... 3167
Visualizações SVCS para o cluster principal e clusters de escalabilidade de simultaneidade ... 3227
Visualizações SVL para o cluster principal ... 3257
Tabelas de catálogo do sistema ... 3339
PG_ATTRIBUTE_INFO ... 3339
PG_CLASS_INFO ... 3340
PG_DATABASE_INFO ... 3342
PG_DEFAULT_ACL ... 3343
PG_EXTERNAL_SCHEMA ... 3345
PG_LIBRARY ... 3346
PG_PROC_INFO ... 3347
PG_STATISTIC_INDICATOR ... 3348
PG_TABLE_DEF ... 3349
PG_USER_INFO ... 3352
Consultar as tabelas de catálogo ... 3353
Referência da configuração ... 3360
Modificar a configuração do servidor ... 3361

analyze_threshold_percent 3362
Valores (padrão em negrito) 3362
Descrição 3362
Exemplos 3363
cast_super_null_on_error 3363
Valores (padrão em negrito) 3363
Descrição 3363
datashare_break_glass_session_var 3363
Valores (padrão em negrito) 3363
Descrição 3363
Exemplo 3364
datestyle 3364
Valores (padrão em negrito) 3364
Descrição 3363
Exemplo 3365
default_array_search_null_handling 3365
Valores (padrão em negrito) 3365
Descrição 3363
Exemplos 3365
default_geometry_encoding 3366
Valores (padrão em negrito) 3366
Descrição 3363
describe_field_name_in_uppercase 3366
Valores (padrão em negrito) 3366
Descrição 3363
Exemplo 3365
downcase_delimited_identifier 3367
Valores (padrão em negrito) 3367
Descrição 3363
Observações sobre o uso 3367
enable_case_sensitive_identifier 3369
Valores (padrão em negrito) 3369
Descrição 3369
Exemplos 3369
Observações sobre o uso 3371
enable_case_sensitive_super_attribute 3372

Valores (padrão em negrito) 3372
Descrição 3372
Exemplos 3373
Observações sobre o uso 3374
enable_numeric_rounding 3374
Valores (padrão em negrito) 3374
Descrição 3375
Exemplo 3375
enable_result_cache_for_session 3376
Valores (padrão em negrito) 3376
Descrição 3376
Exemplo 3377
enable_vacuum_boost 3377
Valores (padrão em negrito) 3377
Descrição 3363
error_on_nondeterministic_update 3377
Valores (padrão em negrito) 3377
Descrição 3363
Exemplo 3365
extra_float_digits 3378
Valores (padrão em negrito) 3378
Descrição 3378
Exemplo 3378
interval_forbid_composite_literals 3379
Valores (padrão em negrito) 3379
Descrição 3363
json_serialization_enable 3380
Valores (padrão em negrito) 3380
Descrição 3363
json_serialization_parse_nested_strings 3380
Valores (padrão em negrito) 3380
Descrição 3363
max_concurrency_scaling_clusters 3381
Valores (padrão em negrito) 3381
Descrição 3381
max_cursor_result_set_size 3381

Valores (padrão em negrito) 3381
Descrição 3381
max_failed_login_attempts 3381
Valores (padrão em negrito) 3381
Descrição 3382
Exemplo 3382
mv_enable_aqmv_for_session 3382
Valores (padrão em negrito) 3382
Descrição 3382
navigate_super_null_on_error 3382
Valores (padrão em negrito) 3382
Descrição 3363
parse_super_null_on_error 3383
Valores (padrão em negrito) 3383
Descrição 3363
pg_federation_repeatable_read 3383
Valores (padrão em negrito) 3383
Descrição 3363
Exemplos 3384
query_group 3384
Valores (padrão em negrito) 3384
Descrição 3384
search_path 3385
Valores (padrão em negrito) 3385
Descrição 3385
Exemplo 3386
spectrum_enable_pseudo_columns 3387
Valores (padrão em negrito) 3387
Descrição 3387
Exemplo 3387
enable_spectrum_oid 3387
Valores (padrão em negrito) 3387
Descrição 3388
Exemplo 3388
spectrum_query_maxerror 3388
Valores (padrão em negrito) 3388

Descrição ... 3388
Exemplo ... 3388
statement_timeout ... 3389
Valores (padrão em negrito) ... 3389
Descrição ... 3389
Exemplo ... 3389
stored_proc_log_min_messages ... 3389
Valores (padrão em negrito) ... 3389
Descrição ... 3363
timezone ... 3390
Valores (padrão em negrito) ... 3390
Sintaxe ... 3390
Descrição ... 3390
Formatos de fuso horário ... 3391
Exemplos ... 3393
use_fips_ssl ... 3394
Valores (padrão em negrito) ... 3394
Descrição ... 3363
wlm_query_slot_count ... 3394
Valores (padrão em negrito) ... 3394
Descrição ... 3394
Exemplos ... 3395
Histórico do documento ... 3396
Atualizações anteriores ... 3411

O Amazon Redshift deixará de oferecer suporte ao uso de UDFs em Python após 30 de junho de 2026. Começaremos a aplicar essa alteração em fases. Consulte mais informações sobre os detalhes do fim da vida útil do Python e as opções de migração na publicação do blog publicada em 30 de junho de 2025.

Introdução ao Amazon Redshift

Boas-vindas! Este é o Guia do desenvolvedor de banco de dados do Amazon Redshift. O principal objetivo deste guia é ajudar você a compreender como o Amazon Redshift é usado para criar e gerenciar um data warehouse. Se você trabalha com bancos de dados como designer, desenvolvedor de software ou administrador, este guia fornece as informações necessárias para projetar, construir, consultar e manter seu data warehouse.

O Amazon Redshift é um serviço de data warehouse em escala de petabytes totalmente gerenciado na nuvem. Com o Amazon Redshift sem servidor, você pode acessar e analisar dados sem as configurações usuais de um data warehouse provisionado. Os recursos são provisionados automaticamente e a capacidade do data warehouse escala de maneira inteligente para oferecer performance rápida até mesmo às workloads mais exigentes e imprevisíveis. O tempo em que o data warehouse fica ocioso não é cobrado, portanto você paga apenas pelo que usa. Independentemente do tamanho do conjunto de dados, você pode carregar dados e começar a consultar imediatamente no editor de consultas v2 do Amazon Redshift ou na sua ferramenta de business intelligence (BI) favorita. Aproveite a melhor relação preço/performance e recursos de SQL familiares em um ambiente fácil de usar e que não exige administração.

Tópicos

- Pré-requisitos para usar o Amazon Redshift
- Arquitetura do Amazon Redshift
- Banco de dados de exemplo

Pré-requisitos para usar o Amazon Redshift

Este tópico descreve os pré-requisitos para usar o Amazon Redshift.

Antes de usar este guia, leia Criar um data warehouse com o Amazon Redshift sem servidor, que explica como concluir as tarefas a seguir.

- Crie um data warehouse com o Amazon Redshift sem servidor.
- Carregar dados de amostra com o editor de consultas v2 do Amazon Redshift
- Carregar dados do Amazon S3.

Você também saber como usar seu cliente SQL e deve ter uma compreensão fundamentalmente do idioma SQL.

Arquitetura do Amazon Redshift

Este tópico ajuda a entender os elementos que compõem o Amazon Redshift.

Um data warehouse do Amazon Redshift é um sistema de gerenciamento e consulta de banco de dados relacional de classe empresarial.

O Amazon Redshift oferece suporte a conexões de clientes com muitos tipos de aplicações, incluindo business intelligence (BI), relatórios, dados e ferramentas analíticas.

Ao executar consultas analíticas, você recupera, compara e avalia grandes volumes de dados em operações de várias etapas para produzir um resultado final.

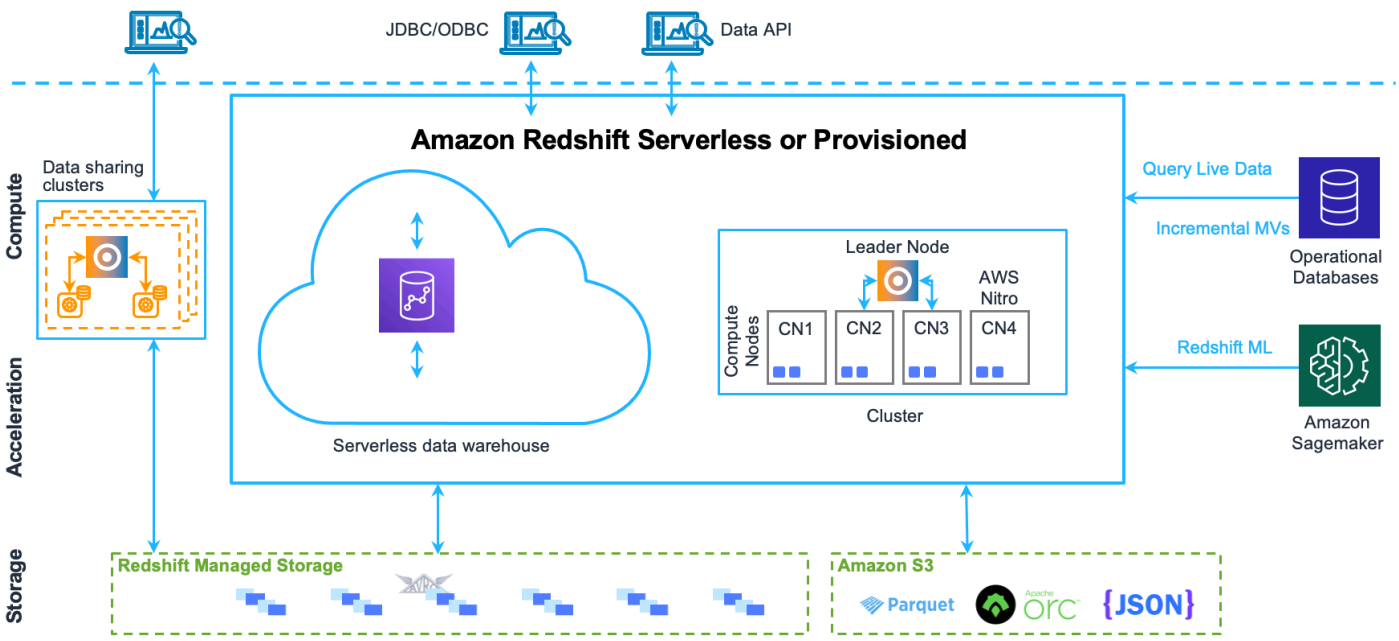
O Amazon Redshift atinge armazenamento eficiente e performance de consulta ideal por meio de uma combinação de processamento massivamente paralelo, armazenamento de dados em colunas e esquemas de codificação de compactação de dados direcionados muito eficientes. Esta seção apresenta uma introdução à arquitetura do sistema Amazon Redshift.

Tópicos

- Arquitetura de sistema do data warehouse
- Performance do Amazon Redshift
- Armazenamento colunar
- Gerenciamento do workload
- Usar Amazon Redshift com outros serviços

Arquitetura de sistema do data warehouse

Esta seção explica os componentes da arquitetura de data warehouse do Amazon Redshift, conforme mostrado na figura a seguir.



Aplicativos cliente

O Amazon Redshift se integra a diversas ferramentas de carregamento de dados e ETL (extração, transformação e carregamento), além de ferramentas de BI (business intelligence), relatórios, mineração de dados e analíticas. O Amazon Redshift se baseia no PostgreSQL de padrão aberto, portanto a maioria das aplicações cliente SQL existentes funcionará somente com alterações mínimas. Para obter informações sobre diferenças importantes entre Amazon Redshift SQL e PostgreSQL, consulte Amazon Redshift e PostgreSQL.

Clusters

O principal componente da infraestrutura de um data warehouse do Amazon Redshift é um cluster.

Um cluster é composto de um ou mais nós de computação. Se um cluster for provisionado com dois ou mais nós de computação, um nó líder adicional coordenará os nós de computação e processará a comunicação externa. O aplicativo cliente interage diretamente somente com o nó líder. Os nós de computação são transparentes a aplicativos externos.

Nó líder

O nó líder gerencia a comunicação com programas cliente e toda a comunicação com nós de computação. Ele analisa e desenvolve planos de execução para realizar operações de banco de dados, em especial, a série de etapas necessárias a fim de obter resultados para consultas

complexas. Com base no plano de execução, o nó líder compila código, distribui o código compilado aos nós de computação e atribui uma parte dos dados a cada nó de computação.

O nó líder distribui instruções SQL para os nós de computação somente quando uma consulta referencia tabelas armazenadas nos nós de computação. Todas as outras consultas são executadas de maneira exclusiva no nó de liderança. O Amazon Redshift foi projetado para implementar determinadas funções SQL somente no nó líder. Uma consulta que usa qualquer uma dessas funções retornará um erro se referenciar tabelas que residam nos nós de computação. Para obter mais informações, consulte Funções SQL compatíveis no nó de liderança.

Nós de computação

O nó líder compila código de elementos individuais do plano de execução e atribui o código aos nós de computação individuais. Os nós de computação executam o código compilado e reenviam os resultados intermediários ao nó líder para agregação final.

Cada nó de computação tem as próprias CPU e memória dedicadas, determinadas pelo tipo de nó. À medida que sua workload cresce, você pode aumentar a capacidade computacional de um cluster aumentando o número de nós, atualizando o tipo de nó ou ambos.

O Amazon Redshift oferece vários tipos de nós para suas necessidades de computação. Para obter detalhes de cada tipo de nó, consulte “Clusters do Amazon Redshift” no Guia de gerenciamento de clusters do Amazon Redshift.

Redshift Managed Storage

Os dados do data warehouse são armazenados em outro nível de armazenamento do Redshift Managed Storage (RMS). O RMS oferece a capacidade de escalar seu armazenamento para petabytes usando o armazenamento do Amazon S3. Com o RMS, é possível escalar e pagar por computação e armazenamento de forma independente, para que você possa dimensionar o cluster com base apenas nas suas necessidades de computação. Ele usa automaticamente o armazenamento local baseado em SSD de alto desempenho como cache de nível 1. Também aproveita as otimizações, como temperatura do bloco de dados, idade do bloco de dados e padrões de workload, para oferecer alta performance e escalar o armazenamento automaticamente para o Amazon S3, quando necessário, sem exigir nenhuma ação.

Fatias de nó

Um nó de computação é particionado em fatias. Cada fatia recebe uma parte da memória do nó e do espaço em disco, em que processa uma parte do workload atribuído ao nó. O nó líder gerencia

dados a distribuição de dados para as fatias e divide o workload para todas as consultas ou outras operações de banco de dados para as fatias. Assim, as fatias funcionam em paralelo para completar a operação.

O número de fatias por nó é determinado pelo tamanho do nó do cluster. Para obter mais informações sobre o número de fatias para cada tamanho de nó, acesse “Clusters e nós no Amazon Redshift” no Guia de gerenciamento de clusters do Amazon Redshift.

Ao criar uma tabela, você também pode especificar uma coluna como a chave de distribuição. Quando a tabela é carregada com dados, as linhas são distribuídas para as fatias de nó de acordo com a chave de distribuição definida para uma tabela. A escolha de uma boa chave de distribuição permite que o Amazon Redshift use processamento paralelo para carregar dados e executar consultas com eficiência. Para obter informações sobre como escolher uma chave de distribuição, consulte Escolha o melhor estilo de distribuição.

Rede interna

O Amazon Redshift tira proveito de conexões de alta largura de banda, proximidade e protocolos de comunicação personalizados para fornecer comunicação de rede privada e de alta velocidade entre o nó líder e os nós de computação. Os nós de computação são executados em uma rede isolada, separada, que aplicativos cliente jamais acessam diretamente.

Bancos de dados na

Um cluster contém um ou mais bancos de dados. Os dados do usuário são armazenados nos nós de computação. O cliente SQL se comunica com o nó líder, que, por sua vez, coordena a execução de consultas com os nós de computação.

O Amazon Redshift é um sistema de gerenciamento de banco de dados relacional (RDBMS), portanto, é compatível com outras aplicações RDBMS. Embora forneça a mesma funcionalidade de um RDBMS típico, incluindo funções de processamento de transações online (OLTP), como inserção e exclusão de dados, o Amazon Redshift é otimizado para análise de alta performance e relatórios de conjuntos de dados muito grandes.

O Amazon Redshift é baseado no PostgreSQL. O Amazon Redshift e o PostgreSQL têm uma série de diferenças muito importantes que você precisa levar em consideração ao projetar e desenvolver suas aplicações de data warehouse. Para obter informações sobre como o Amazon Redshift SQL difere do PostgreSQL, consulte Amazon Redshift e PostgreSQL.

Performance do Amazon Redshift

Este tópico descreve os componentes do Amazon Redshift que aumentam a performance. O conhecimento sobre esses componentes ajudará você a ajustar a performance e solucionar problemas de baixa performance com o Amazon Redshift.

O Amazon Redshift alcança uma execução de consultas extremamente rápida ao empregar esses recursos de performance.

Tópicos

- Processamento paralelo em massa
- Armazenamento de dados colunar
- Compactação de dados
- Otimizador de consultas
- Armazenamento em cache dos resultados
- Código compilado

Processamento paralelo em massa

O processamento paralelo em massa (MPP) permite a execução rápida das consultas mais complexas que utilizam grandes volumes de dados. Vários nós de computação fazem todo o processamento de consultas chegando à agregação do resultado final, com cada núcleo de cada nó executando os mesmos segmentos de consulta compilados em partes de todos os dados.

O Amazon Redshift distribui as linhas de uma tabela para os nós de computação para que os dados possam ser processados em paralelo. Selecionando uma chave de distribuição apropriada para cada tabela, você pode otimizar a distribuição de dados para equilibrar o workload e minimizar o movimento de dados entre nós. Para obter mais informações, consulte Escolha o melhor estilo de distribuição.

O carregamento de dados de arquivos simples aproveita o processamento paralelo, distribuindo o workload entre vários nós enquanto lê vários arquivos simultaneamente. Para obter mais informações sobre como carregar dados em tabelas, consulte Práticas recomendadas do Amazon Redshift para carregamento de dados.

Armazenamento de dados colunar

O armazenamento colunar para tabelas de bancos de dados reduz drasticamente os requisitos de E/S de disco gerais e é um fator importante na otimização da performance de consulta analítica. Para obter mais informações, consulte Armazenamento colunar.

Quando as colunas estão classificadas corretamente, o processador de consultas pode filtrar rapidamente um grande subconjunto de blocos de dados. Para obter mais informações, consulte Escolha a melhor chave de classificação.

Compactação de dados

A compactação de dados reduz requisitos de armazenamento, o que diminui a E/S de disco, melhorando a performance da consulta. Quando você executa uma consulta, os dados compactados são lidos para a memória, depois são descompactados durante a execução da consulta. Carregar menos dados na memória permite que o Amazon Redshift aloque mais memória para analisar os dados. Como o armazenamento colunar armazena dados semelhantes sequencialmente, o Amazon Redshift é capaz de aplicar codificações de compressão adaptáveis especificamente vinculadas a tipos de dados colunares. A melhor maneira de habilitar a compactação de dados nas colunas da tabela é permitir que o Amazon Redshift aplique as codificações de compactação ideais ao carregar a tabela com dados. Para saber mais sobre como usar a compactação de dados automática, consulte Carregamento de tabelas com compactação automática.

Otimizador de consultas

O mecanismo de execução de consultas do Amazon Redshift incorpora um otimizador de consultas com reconhecimento de MPP e também aproveita o armazenamento de dados orientado por colunas. O otimizador de consulta Amazon Redshift implementa melhorias e extensões significativas para o processamento de consultas analíticas complexas que geralmente incluem junções de várias tabelas, subconsultas e agregação. Para saber mais sobre como otimizar consultas, consulte Ajustar o desempenho da consulta.

Armazenamento em cache dos resultados

Para reduzir o tempo de execução da consulta e melhorar a performance do sistema, o Amazon Redshift armazena em cache os resultados de certos tipos de consultas na memória no nó líder. Quando um usuário envia uma consulta, o Amazon Redshift verifica o cache de resultados em busca de uma cópia em cache válida dos resultados da consulta. Se for encontrada uma correspondência no cache de resultados, o Amazon Redshift usa os resultados armazenados em cache e não executa a consulta. O cache de resultados é transparente para o usuário.

O armazenamento em cache de resultados é ativado por padrão. Para desativar o armazenamento em cache de resultados na sessão atual, defina o parâmetro enable_result_cache_for_session como `off`.

O Amazon Redshift usa resultados em cache para uma nova consulta quando todas as seguintes opções são verdadeiras:

- O usuário que envia a consulta possui permissão de acesso aos objetos usados na consulta.
- A tabela ou as exibições na consulta não foram modificadas.
- A consulta não usa uma função que precisa ser avaliada cada vez que ela é executada, por exemplo, GETDATE.
- A consulta não faz referência a tabelas externas do Amazon Redshift Spectrum.
- Os parâmetros de configuração que podem afetar os resultados da consulta estão inalterados.
- A consulta coincide sintaticamente com a consulta armazenada em cache.

Para maximizar a eficácia do cache e da utilização dos recursos, o Amazon Redshift não armazena em cache alguns conjuntos grandes de resultados de consultas. O Amazon Redshift determina se os resultados da consulta serão ou não armazenados em cache. Esses fatores incluem o número de entradas no cache e o tipo de instância do seu cluster Amazon Redshift.

Para determinar se uma consulta usou o cache de resultados, abra a visualização do sistema SVL_QLOG. Se uma consulta utilizar o cache de resultados, a coluna source_query retornará o ID da consulta de origem. Se o cache de resultados não tiver sido utilizado, o valor da coluna source_query será NULL.

O exemplo a seguir mostra que as consultas enviadas pelo userid 104 e o userid 102 utilizaram o cache de resultados das consultas realizadas pelo userid 100.

```
select userid, query, elapsed, source_query from svl_qlog
where userid > 1
order by query desc;

userid | query  | elapsed  | source_query
-------+--------+----------+-------------
   104 | 629035 |       27 |       628919
   104 | 629034 |       60 |       628900
   104 | 629033 |       23 |       628891
   102 | 629017 |  1229393 |
   102 | 628942 |       28 |       628919
```

```
  102 | 628941 |       57 |       628900
  102 | 628940 |       26 |       628891
  100 | 628919 | 84295686 |
  100 | 628900 | 87015637 |
  100 | 628891 | 58808694 |
```

Código compilado

Compilação de código: o Amazon Redshift gera e compila o código otimizado para cada plano de execução de consulta. O código compilado é executado mais rápido porque elimina a sobrecarga de usar um interpretador. Para minimizar a latência de novas consultas e, ao mesmo tempo, preservar os benefícios de performance do código compilado, o Amazon Redshift usa uma técnica chamada composição. A composição gera um arranjo leve de lógica preexistente para processar novas consultas imediatamente e, ao mesmo tempo, compilar código altamente otimizado e específico da consulta em segundo plano. Isso remove a compilação do caminho crítico da execução da consulta, para que novas consultas sejam iniciadas mais rapidamente e ofereçam performance consistente com as execuções subsequentes.

O Amazon Redshift também usa um serviço de compilação sem servidor para escalar compilações de consulta além dos recursos de computação de um cluster do Amazon Redshift. Os segmentos de código compilados são armazenados em cache localmente no cluster e em um cache praticamente ilimitado que se mantém após a reinicialização do cluster. As execuções subsequentes da mesma consulta são mais rápidas porque podem pular a fase de compilação. Usando um serviço de compilação escalável, o Amazon Redshift compele código em paralelo para oferecer uma performance consistentemente rápida.

Armazenamento colunar

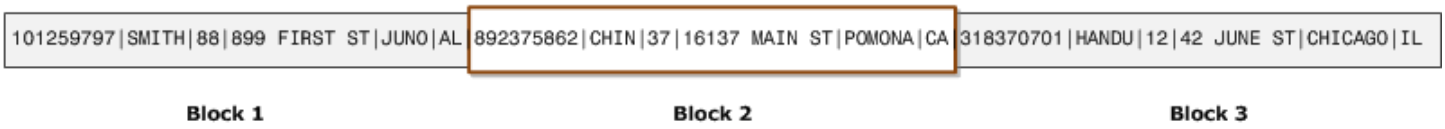
Esta seção descreve o armazenamento colunar, que é o método que o Amazon Redshift usa para armazenar dados tabulares de forma eficiente.

O armazenamento colunar para tabelas de bancos de dados é um fator importante na otimização da performance de consulta analítica, pois reduz drasticamente os requisitos de E/S de disco gerais. Ele reduz a quantidade de dados que você precisa carregar do disco.

A série de ilustrações a seguir descreve como o armazenamento de dados colunar implementa eficiências e se converte em eficiências durante a recuperação de dados na memória.

A primeira ilustração mostra como registros de tabelas de banco de dados normalmente são armazenados em blocos de discos por linha.

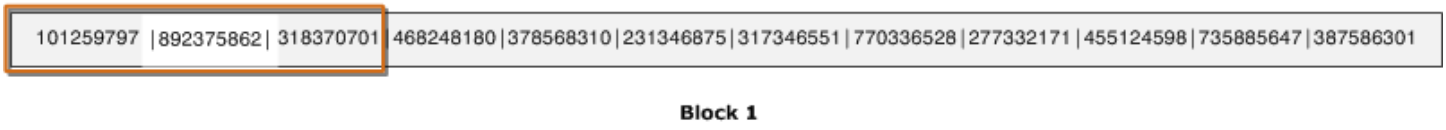
SSN	Name	Age	Addr	City	St
101259797	SMITH	88	899 FIRST ST	JUNO	AL
892375862	CHIN	37	16137 MAIN ST	POMONA	CA
318370701	HANDU	12	42 JUNE ST	CHICAGO	IL



Em uma tabela de banco de dados relacional típica, cada linha contém valores de campo para um único registro. No armazenamento de banco de dados compatível com linha, os blocos de dados armazenam valores sequencialmente para cada coluna consecutiva que compõe toda a linha. Se o tamanho do bloco for menor que o tamanho de um registro, o armazenamento de um registro inteiro poderá usar mais de um bloco. Se o tamanho do bloco for maior que o tamanho de um registro, o armazenamento de um registro inteiro poderá usar menos de um bloco, resultando em um uso ineficiente do espaço em disco. Em aplicativos OLTP, a maioria das transações envolve sempre leitura e gravação de todos os valores de registros inteiros, normalmente um registro ou um pequeno número de registros por vez. Dessa forma, o armazenamento compatível com linha é ideal para bancos de dados OLTP.

A próxima ilustração mostra como, com armazenamento colunar, os valores de cada coluna são armazenados sequencialmente em blocos em disco.

SSN	Name	Age	Addr	City	St
101259797	SMITH	88	899 FIRST ST	JUNO	AL
892375862	CHIN	37	16137 MAIN ST	POMONA	CA
318370701	HANDU	12	42 JUNE ST	CHICAGO	IL



Usando armazenamento columnar, cada bloco de dados armazena valores de uma única coluna de várias linhas. Conforme os registros entram no sistema, o Amazon Redshift converte de forma transparente os dados em armazenamento colunar para cada uma das colunas.

Neste exemplo simplificado, usando armazenamento colunar, cada bloco de dados mantém valores de campo da coluna até três vezes mais do que muitos registros de armazenamento baseado

em linha. Isso significa que ler o mesmo número de valores do campo de coluna para o mesmo número de registros exige um terço das operações de E/S em comparação com o armazenamento compatível com linha. Na prática, usando tabelas com números de colunas e contagens de linhas muito grandes, a eficiência do armazenamento é ainda maior.

Uma vantagem adicional é que, como cada bloco contém o mesmo tipo de dados, os dados do bloco podem usar um esquema de compactação selecionado especificamente para o tipo de dados da coluna, reduzindo ainda mais o espaço em disco e E/S. Para obter mais informações sobre codificações de compactação com base em tipos de dados, consulte Codificações de compactação.

A economia em termos de espaço para armazenar dados em disco também chega à recuperação e ao armazenamento desses dados na memória. Como muitas operações de banco de dados precisam somente acessar ou operar uma ou um pequeno número de coluna por vez, você pode economizar espaço na memória somente recuperando blocos para colunas de que realmente precisa para uma consulta. Quando as transações OLTP normalmente envolvem a maioria ou todas as colunas em uma linha para um número pequeno de registros, as consultas de data warehouse normalmente leem somente algumas colunas para um número muito grande de linhas. Isso significa que ler o mesmo número de valores do campo de coluna para o mesmo número de linhas exige uma fração das operações de E/S. Ele usa uma fração da memória que seria necessária para processar blocos em linha. Na prática, usando tabelas com números de colunas e contagens de linhas muito grandes, os ganhos de eficiência são proporcionalmente maiores. Por exemplo, suponhamos que uma tabela contenha 100 colunas. Uma consulta que usa cinco colunas precisará ler somente cerca de cinco por cento dos dados contidos na tabela. Essa economia se repete possivelmente para bilhões ou até mesmo trilhões de registros para bancos de dados grandes. Por outro lado, um banco de dados compatível com linha também leria blocos que contivessem as 95 colunas indesejadas.

Os tamanhos típicos de blocos de banco de dados variam de 2 KB a 32 KB. O Amazon Redshift usa um tamanho de bloco de 1 MB, o que é mais eficiente e reduz ainda mais o número de solicitações de E/S necessárias para executar qualquer carregamento de banco de dados ou outras operações que façam parte da execução de consultas.

Gerenciamento do workload

Esta seção descreve o gerenciamento de workloads (WLM), que ajuda a entender como o Amazon Redshift prepara e executa consultas.

O gerenciamento de workload (WLM) do Amazon Redshift permite prioridades de gerenciamento flexível dentro dos workloads para que consultas curtas e de execução rápida não fiquem presas

em filas atrás de consultas de longa duração. O Amazon Redshift cria filas de consulta no runtime de acordo com as classes de serviço, que definem os parâmetros de configuração para vários tipos de filas, incluindo filas internas do sistema e filas acessíveis ao usuário. Do ponto de vista de um usuário, uma classe de serviço acessível e uma fila são funcionalmente equivalentes. Tendo em vista a consistência, esta documentação usa o termo fila para indicar uma classe de serviço acessível pelo usuário, bem como uma fila de tempo de execução.

O Redshift oferece o gerenciamento de workload automático, chamado de WLM automático, que é ajustado para lidar com workloads variáveis e é o padrão recomendado. Com o WLM automático, o Redshift determina a utilização dos recursos à medida que as consultas chegam e estabelece dinamicamente se elas devem ser executadas no cluster principal, em um cluster com escalabilidade simultânea ou se devem ser enviadas para uma fila. (Quando as consultas são enfileiradas, o WLM automático prioriza consultas de menor duração.) O WLM automático maximiza o throughput total para que você possa manter recursos eficientes do data warehouse. Você executa workloads sem precisar se preocupar com o tamanho ou com a forma como estão programadas. O WLM automático é o padrão para clusters provisionados. Consulte mais informações em Implementar o WLM automático.

> **Note**
>
> Os grupos de trabalho do Amazon Redshift sem servidor sempre usam o WLM automático.

Em momentos em que muitas consultas ou consultas que consomem muitos recursos estão sendo executadas, é possível escalar o gerenciamento do workload para recursos de computação adicionais quando os workloads são colocados na fila em recursos locais. A escalabilidade da simultaneidade com o WLM automático é compatível com um desempenho consistente de usuários e consultas simultâneos praticamente ilimitados.

Os clusters provisionados do Redshift oferecem o WLM manual caso você precise de uma otimização manual refinada. Aqui, o cliente gerencia a alocação de recursos, a simultaneidade de consultas e o enfileiramento. Quando uma consulta é executada, o WLM atribui a consulta a uma fila de acordo com o grupo de usuários ou comparando um grupo de consultas listado na configuração da fila. Isso é configurado com um rótulo de grupo de consulta definido pelo usuário. Consulte mais informações em Implementar o WLM manual.

Embora o WLM manual possa ser ajustado ao longo do tempo para corresponder aos padrões do workload, na maioria dos casos, desencorajamos seu uso porque sua natureza estática pode

dificultar a adaptação às mudanças nos workloads ao longo de um dia ou por um longo período. Isso requer mais monitoramento e ajustes contínuos. Além disso, o WLM manual, em muitos casos, não usa recursos de computação de forma tão eficiente quanto o WLM automático, como, por exemplo, se as filas são definidas manualmente para limitar a memória alocada a elas.

Uma métrica importante para medir o sucesso da configuração do gerenciamento do workload é o throughput do sistema, que, em outras palavras, indica quantas consultas são concluídas com êxito. O throughput do sistema é medido em metros por segundo. Consulte mais informações sobre as métricas do sistema em Monitorar a performance do cluster do Amazon Redshift.

A maneira mais fácil de gerenciar a configuração do WLM é usando o console de gerenciamento do Amazon Redshift. Você também pode usar a interface de linha de comandos (CLI) do Amazon Redshift ou a API do Amazon Redshift. Consulte mais informações sobre como implementar e usar o gerenciamento de workload em Como implementar o gerenciamento do workload.

Usar Amazon Redshift com outros serviços

Esta seção descreve como você pode usar outros serviços como origem e destino para dados do Amazon Redshift.

O Amazon Redshift integra-se a outros serviços da AWS para que seja possível mover, transformar e carregar seus dados de maneira rápida e confiável, usando recursos de segurança de dados.

S3

O Amazon Simple Storage Service (Amazon S3) é um serviço da Web que armazena dados na nuvem. O Amazon Redshift utiliza o processamento paralelo para ler e carregar dados de vários arquivos de dados armazenados em buckets do Amazon S3. Para obter mais informações, consulte Carregar dados do Amazon S3.

Você também pode usar o processamento paralelo para exportar dados de seu data warehouse do Amazon Redshift para vários arquivos de dados no Amazon S3. Para obter mais informações, consulte Descarregar dados no Amazon Redshift.

DynamoDB

O Amazon DynamoDB é um serviço de banco de dados NoSQL totalmente gerenciado. Você pode usar o comando COPY para carregar uma tabela do Amazon Redshift com dados de uma única tabela do Amazon DynamoDB. Para obter mais informações, consulte Carregar dados de uma tabela do Amazon DynamoDB.

SSH

Você pode usar o comando COPY no Amazon Redshift para carregar dados de um ou mais hosts remotos, como clusters do Amazon EMR, instâncias do Amazon EC2 ou outros computadores. COPY se conecta aos hosts remotos usando SSH e executa comandos nos hosts remotos para gerar dados. O Amazon Redshift suporta várias conexões simultâneas. O comando COPY lê e carrega a saída de várias origens de host em paralelo. Para obter mais informações, consulte Carregamento de dados de hosts remotos.

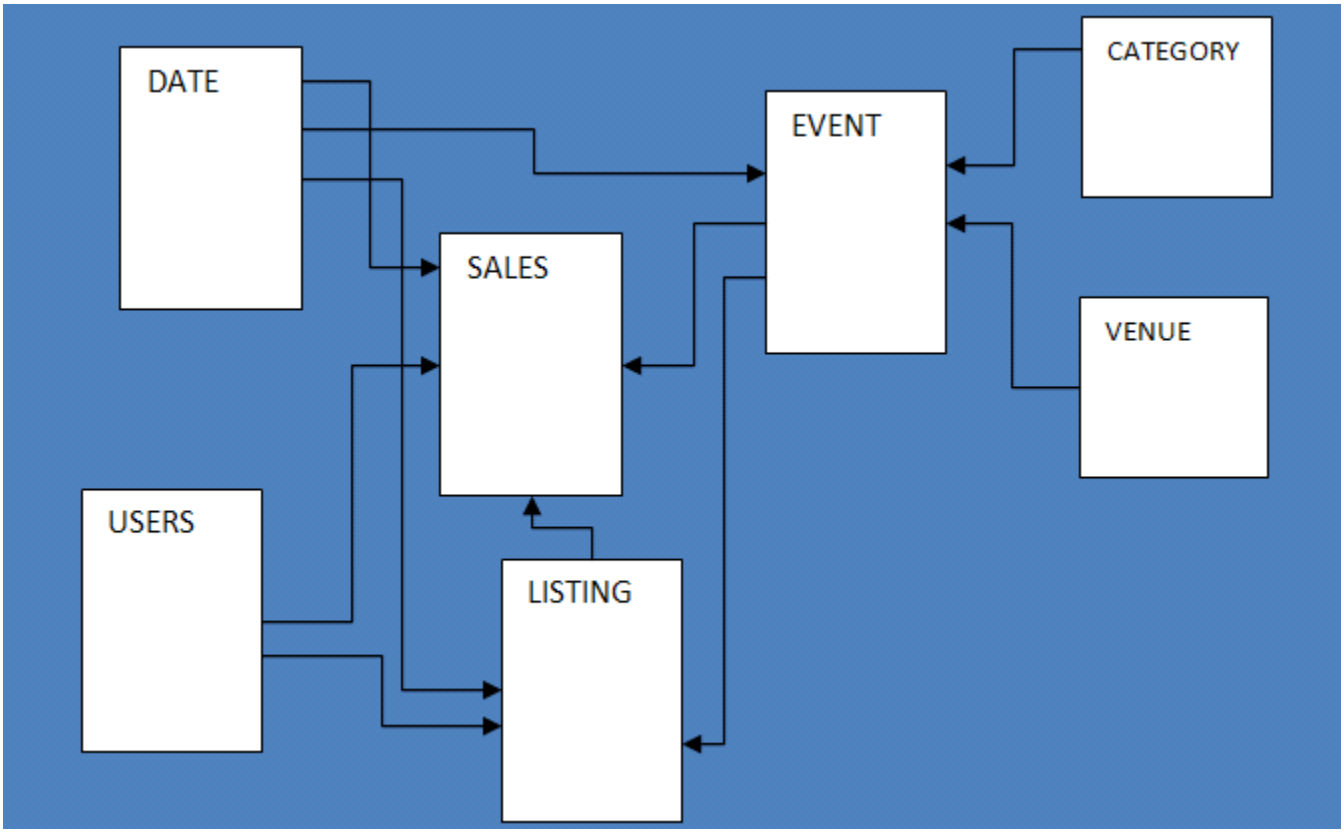
AWS DMS

É possível migrar dados para o Amazon Redshift usando o AWS Database Migration Service. O AWS DMS pode migrar seus dados de e para os bancos de dados comerciais e de código aberto mais usados, como Oracle, PostgreSQL, Microsoft SQL Server, Amazon Redshift, cluster de banco de dados do Aurora, DynamoDB, Amazon S3, MariaDB e MySQL. Para obter mais informações, consulte Usar um banco de dados do Amazon Redshift como destino do AWS Database Migration Service.

Banco de dados de exemplo

Esta seção descreve o TICKIT, um banco de dados que os exemplos da documentação do Amazon Redshift usam.

Este pequeno banco de dados consiste em sete tabelas: duas tabelas de fatos e cinco dimensões. É possível carregar o conjunto de dados TICKIT seguindo as etapas em Etapa 4: Carregar dados do Amazon S3 para o Amazon Redshift no Guia de conceitos básicos do Amazon Redshift.



Esse aplicativo de banco de dados de exemplo ajuda analistas a acompanhar a atividade de vendas do site fictício TICKIT, onde usuários compram e vendem ingressos online para eventos esportivos, shows e concertos. Especificamente, os analistas podem identificar o movimento dos ingressos ao longo do tempo, as taxas de sucesso para vendedores e os eventos, locais e estações mais bem vendidos. Os analistas podem usar essas informações para fornecer incentivos para compradores e vendedores que frequentam o site, para atrair novos usuários e para promover publicidade e promoções.

Por exemplo, a seguinte consulta localiza os cinco principais vendedores em San Diego com base no número de ingressos vendidos em 2008:

```
select sellerid, username, (firstname ||' '|| lastname) as name,
city, sum(qtysold)
from sales, date, users
where sales.sellerid = users.userid
and sales.dateid = date.dateid
and year = 2008
and city = 'San Diego'
group by sellerid, username, name, city
order by 5 desc
```

```
limit 5;

sellerid | username |       name        |   city    | sum
---------+----------+-------------------+-----------+-----
49977 | JJK84WTE | Julie Hanson      | San Diego |  22
19750 | AAS23BDR | Charity Zimmerman | San Diego |  21
29069 | SVL81MEQ | Axel Grant        | San Diego |  17
43632 | VAG08HKW | Griffin Dodson    | San Diego |  16
36712 | RXT40MKU | Hiram Turner      | San Diego |  14
(5 rows)
```

O banco de dados usado para os exemplos neste guia contém um pequeno conjunto de dados; as duas tabelas de fatos contêm menos de 200.000 linhas e as dimensões variam de 11 linhas na tabela CATEGORY a, aproximadamente, 50.000 linhas na tabela USERS.

Especificamente, os exemplos do banco de dados neste guia demonstram os recursos chave do design de tabelas do Amazon Redshift:

- Distribuição de dados
- Classificação de dados
- Compactação de colunas

Para obter informações sobre os esquemas das tabelas no banco de dados TICKIT, escolha as seguintes guias:

CATEGORY table

Nome da coluna	Tipo de dados	Descrição
CATID	SMALLINT	Chave primária, um valor de ID exclusivo para cada linha. Cada linha representa um tipo específico de evento para o qual ingressos são comprados e vendidos.
CATGROUP	VARCHAR (10)	Nome descritivo para um grupo de eventos, como **`Shows`** e **`Sports`**.
CATNAME	VARCHAR (10)	Nome descritivo curto para um tipo de evento dentro de um grupo, como **`Opera`** e **`Musicals`**.

Nome da coluna	Tipo de dados	Descrição
CATDESC	VARCHAR(50)	Nome descritivo mais longo do tipo de evento, como **`Musical theatre`**.

DATE table

Nome da coluna	Tipo de dados	Descrição
DATEID	SMALLINT	Chave primária, um valor de ID exclusivo para cada linha. Cada linha representa um dia no ano civil.
CALDATE	DATE	Data do calendário, como **`2008-06-24`** .
DAY	CHAR(3)	Dia de semana (forma resumida), como **`SA`**.
WEEK	SMALLINT	Número da semana, como **`26`**.
MONTH	CHAR(5)	Nome do mês (forma resumida), como **`JUN`**.
QTR	CHAR(5)	Número do trimestre (**`1`** a **`4`**).
YEAR	SMALLINT	Ano de quatro dígitos (**`2008`**).
HOLIDAY	BOOLEAN	Bandeira que denota se o dia é um feriado público (EUA).

EVENT table

Nome da coluna	Tipo de dados	Descrição
EVENTID	INTEGER	Chave primária, um valor de ID exclusivo para cada linha. Cada linha representa um evento distinto que ocorre em um local específico e em uma hora específica.

Nome da coluna	Tipo de dados	Descrição
VENUEID	SMALLINT	Referência de chave estrangeira para a tabela VENUE.
CATID	SMALLINT	Referência de chave estrangeira para a tabela CATEGORY.
DATEID	SMALLINT	Referência de chave estrangeira para a tabela DATE.
EVENTNAME	VARCHAR (200)	Nome do evento, como **`Hamlet`** ou **`La Traviata`**.
STARTTIME	TIMESTAMP	Data completa e hora de início do evento, como **`2008-10-10 19:30:00`** .

VENUE table

Nome da coluna	Tipo de dados	Descrição
VENUEID	SMALLINT	Chave primária, um valor de ID exclusivo para cada linha. Cada linha representa um local específico onde os eventos ocorrem.
VENUENAME	VARCHAR (100)	Nome exato do local do evento, como **`Cleveland Browns Stadium`**.
VENUECITY	VARCHAR (30)	Nome da cidade, como **`Cleveland`** .
VENUESTATE	CHAR(2)	Abreviação de duas letras do estado ou da província (Estados Unidos e Canadá), como **`OH`**.
VENUESEATS	INTEGER	Número máximo de assentos disponíveis no local do evento, se conhecido, como **`73200`**. Para fins de demonstração, essa coluna contém alguns valores nulos e zeros.

USERS table

Nome da coluna	Tipo de dados	Descrição
USERID	INTEGER	Chave primária, um valor de ID exclusivo para cada linha. Cada linha representa um usuário registrado (um comprador ou vendedor ou ambos) que tenha listado ou comprado ingressos para, pelo menos, um evento.
USERNAME	CHAR(8)	Um nome de usuário alfanumérico de 8 caracteres, como **`PGL08LJI`**.
FIRSTNAME	VARCHAR (30)	O primeiro nome do usuário, como **`Victor`**.
LASTNAME	VARCHAR (30)	O sobrenome do usuário, como **`Hernandez`** .
CITY	VARCHAR (30)	A cidade de residência do usuário, como **`Naperville`** .
STATE	CHAR(2)	O estado de residência do usuário, como **`GA`**.
EMAIL	VARCHAR (100)	O endereço de e-mail do usuário; essa coluna contém valores latinos aleatórios, como **`turpis@accumsanlaoreet.org`** .
PHONE	CHAR(14)	O número de telefone com 14 caracteres do usuário, como **`(818) 765-4255`**.
LIKESPORTS, ...	BOOLEAN	Uma série de 10 colunas diferentes que identifica o que usuário gosta e não gosta com os valores **`true`** e **`false`**.

LISTING table

Nome da coluna	Tipo de dados	Descrição
LISTID	INTEGER	Chave primária, um valor de ID exclusivo para cada linha. Cada linha representa uma listagem de um lote de ingressos para um evento específico.
SELLERID	INTEGER	Referência de chave estrangeira para a tabela USERS, identificando o usuário que está vendendo os ingressos.
EVENTID	INTEGER	Referência de chave estrangeira para a tabela EVENT.
DATEID	SMALLINT	Referência de chave estrangeira para a tabela DATE.
NUMTICKETS	SMALLINT	O número de ingressos disponíveis para venda, como **2** ou **20**.
PRICEPERTICKET	DECIMAL(8,2)	O preço fixo de um ingresso individual, como **27.00** ou **206.00**.
TOTALPRICE	DECIMAL(8,2)	O preço total para esta oferta (NUMTICKETS*PRICEPERTICKET).
LISTTIME	TIMESTAMP	A data e hora completas quando a oferta foi publicada, como **2008-03-18 07:19:35** .

SALES table

Nome da coluna	Tipo de dados	Descrição
SALESID	INTEGER	Chave primária, um valor de ID exclusivo para cada linha. Cada linha representa uma venda de um ou mais ingressos para um evento específico, conforme oferecido na oferta específica.

Nome da coluna	Tipo de dados	Descrição
LISTID	INTEGER	Referência de chave estrangeira para a tabela LISTING.
SELLERID	INTEGER	Referência de chave estrangeira para a tabela USERS (o usuário que vendeu os ingressos).
BUYERID	INTEGER	Referência de chave estrangeira para a tabela USERS (o usuário que comprou os ingressos).
EVENTID	INTEGER	Referência de chave estrangeira para a tabela EVENT.
DATEID	SMALLINT	Referência de chave estrangeira para a tabela DATE.
QTYSOLD	SMALLINT	O número de ingressos que foram vendidos, de **1** a **8**. (No máximo, 8 ingressos podem ser vendidos em uma única transação.)
PRICEPAID	DECIMAL(8,2)	O preço total pago pelos ingressos, como **75.00** ou **488.00**. O preço individual de um ingresso é PRICEPAID/QTYSOLD.
COMMISSION	DECIMAL(8,2)	A comissão de 15% que a empresa coleta da venda, como **11.25** ou **73.20**. O vendedor recebe 85% do valor de PRICEPAID.
SALETIME	TIMESTAMP	A data e hora completas quando a venda foi efetivada, como **2008-05-24 06:21:47** .

Práticas recomendadas do Amazon Redshift

A seguir, você pode encontrar as práticas recomendadas para planejar uma prova de conceito, projetar tabelas, carregar dados em tabelas e gravar consultas para o Amazon Redshift e também uma discussão sobre como trabalhar com o Amazon Redshift Advisor.

O Amazon Redshift não é o mesmo que outros sistemas de banco de dados SQL. Para aproveitar totalmente os benefícios da arquitetura do Amazon Redshift, você deve projetar, construir e carregar especificamente suas tabelas para usar o processamento paralelo maciço, armazenamento de dados colunares e compactação de dados colunares. Se os tempos de execução de carregamento de dados e consulta são mais longos do que o esperado ou desejado, é possível que você esteja negligenciando informações chave.

Se você for um desenvolvedor de banco de dados SQL experiente, é altamente recomendável revisar este tópico antes de começar a desenvolver seu data warehouse do Amazon Redshift.

Se você é novo no desenvolvimento de bancos de dados SQL, é melhor não começar por este tópico. Recomendamos que primeiro você leia Executar comandos para definir e usar um banco de dados no data warehouse no Guia de conceitos básicos do Amazon Redshift e experimente os exemplos por conta própria.

Neste tópico, você encontra uma visão geral dos princípios de desenvolvimento mais importantes, além de dicas, exemplos e melhores práticas específicos para a implementação desses princípios. Nenhuma prática individual pode se aplicar a todos os aplicativos. Avalie todas as suas opções antes de finalizar um design de banco de dados. Para obter mais informações, consulte Otimização automática de tabelas, Carregar dados no Amazon Redshift, Ajustar o desempenho da consulta e os capítulos de referência.

Tópicos

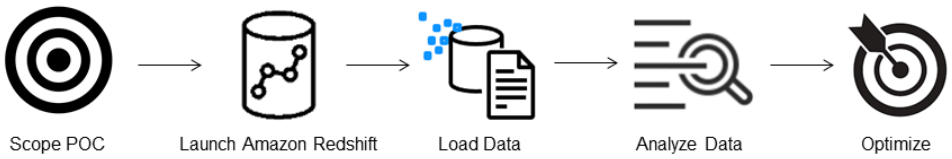
- Realizar uma prova de conceito (POC) para o Amazon Redshift
- Práticas recomendadas do Amazon Redshift para projetar tabelas
- Práticas recomendadas do Amazon Redshift para carregamento de dados
- Práticas recomendadas do Amazon Redshift para criar consultas
- Trabalhar com recomendações do Amazon Redshift Advisor

Realizar uma prova de conceito (POC) para o Amazon Redshift

O Amazon Redshift, um conhecido data warehouse na nuvem, oferece um serviço totalmente gerenciado baseado em nuvem que se integra ao data lake do Amazon Simple Storage Service de uma organização, fluxos de trabalho em tempo real, fluxos de trabalho de machine learning (ML), fluxos de trabalho transacionais e muito mais. As seções a seguir orientam você no processo de realização de uma prova de conceito (POC) no Amazon Redshift. As informações aqui ajudam você a definir metas para a POC e aproveitam as ferramentas que podem automatizar o provisionamento e a configuração de serviços para a POC.

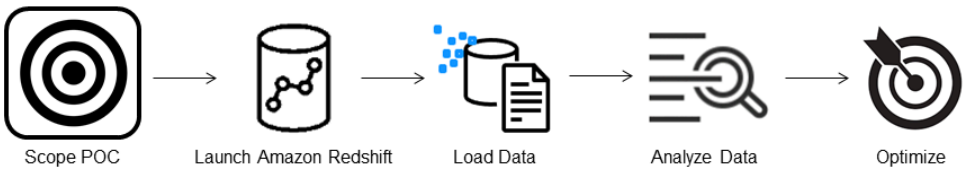
> **Note**
>
> Tenha uma cópia dessas informações em PDF no link Execute sua própria POC do Redshift na página Recursos do Amazon Redshift.

Ao fazer uma POC do Amazon Redshift, você testa, comprova e adota recursos que vão desde recursos de segurança de ponta, escalabilidade elástica, facilidade de integração e ingestão e opções flexíveis de arquitetura de dados descentralizada.



Siga estas etapas para realizar uma POC bem-sucedida.

Etapa 1: Definir o escopo da POC



Ao realizar uma POC, você pode optar por usar seus próprios dados ou usar conjuntos de dados de referência. Ao optar por seus próprios dados, você executa suas próprias consultas nos dados.

Com os dados de referência, são fornecidas consultas de exemplo com a referência. Consulte Usar conjuntos de dados de amostra para obter mais detalhes se você achar que ainda não é o momento certo para realizar uma POC com seus próprios dados.

Geralmente, recomendamos usar duas semanas de dados para uma POC do Amazon Redshift.

Primeiro, faça o seguinte:

1. Identifique seus requisitos comerciais e funcionais e, seguida, trabalhe de trás para frente. Exemplos comuns são: desempenho mais rápido, custos mais baixos, teste de uma nova workload ou um novo recurso ou comparação entre o Amazon Redshift e outro data warehouse.
2. Estabeleça metas específicas para que se tornem os critérios de sucesso para a POC. Por exemplo, com base no desempenho mais rápido, crie uma lista dos cinco principais processos que você deseja acelerar e inclua os tempos de execução atuais com o tempo de execução necessário. Podem ser relatórios, consultas, processos de ETL, ingestão de dados ou qualquer um de seus pontos problemáticos atuais.
3. Identifique o escopo e os artefatos específicos necessários para executar os testes. Quais conjuntos de dados você precisa migrar ou ingerir continuamente no Amazon Redshift e quais consultas e processos são necessários para executar os testes e avaliar os critérios de sucesso? Há duas maneiras de fazer isso:

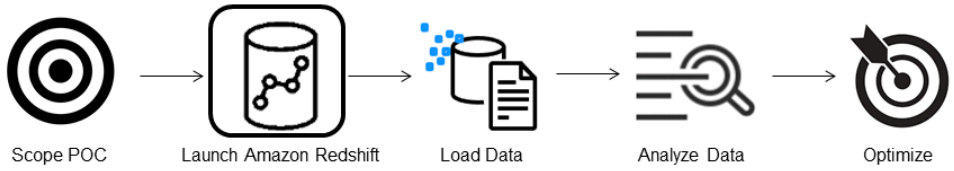
   Traga seus próprios dados

   - Para testar seus próprios dados, crie uma lista mínima viável de artefatos de dados necessários para testar seus critérios de sucesso. Por exemplo, se seu data warehouse atual tem 200 tabelas, mas os relatórios que você deseja testar precisam apenas de 20, sua POC pode ser realizada mais rapidamente usando somente o subconjunto menor de tabelas.

   Usar conjuntos de dados de amostra

   - Se você não tiver seus próprios conjuntos de dados prontos, mesmo assim poderá começar a realizar uma POC no Amazon Redshift usando os conjuntos de dados de referência padrão do setor, como TPC-DS ou TPC-H, e executar consultas de exemplo de referência para aproveitar o potencial do Amazon Redshift. Depois de criados, esses conjuntos de dados podem ser acessados por meio do data warehouse do Amazon Redshift. Para obter instruções detalhadas sobre como acessar esses conjuntos de dados e consultas de exemplo, consulte Etapa 2: Iniciar o Amazon Redshift.

Etapa 2: Iniciar o Amazon Redshift



O Amazon Redshift acelera seu tempo de obtenção de insights com armazenamento de dados rápido, fácil, seguro e em grande escalar. Você pode começar rapidamente iniciando seu warehouse no console do Redshift sem servidor e passar dos dados aos insights em segundos. Com o Redshift sem servidor, você pode se concentrar na obtenção de resultados comerciais sem se preocupar em gerenciar o data warehouse.

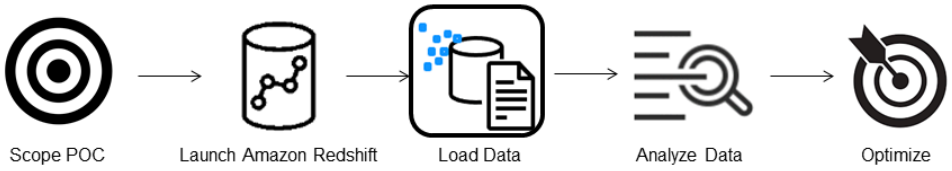
Configurar o Amazon Redshift sem servidor

Na primeira vez que você usa o Redshift sem servidor, o console conduz você pelas etapas necessárias para iniciar o warehouse. Você também pode ser elegível para receber um crédito pelo uso do Redshift sem servidor em sua conta. Para obter mais informações sobre a escolha de um teste gratuito, consulteTeste gratuito do Amazon Redshift. Siga as etapas em Criar um data warehouse com o Amazon Redshift sem servidor no Guia de conceitos básicos do Amazon Redshift para criar um data warehouse com o Redshift sem servidor. Se você não tiver um conjunto de dados que gostaria de carregar, o guia também contém etapas sobre como carregar um conjunto de dados de exemplo.

Se você já iniciou o Redshift sem servidor em sua conta, siga as etapas em Criar um grupo de trabalho com um namespace no Guia de gerenciamento do Amazon Redshift. Depois que o warehouse estiver disponível, você poderá optar por carregar os dados de amostra disponíveis no Amazon Redshift. Para obter mais informações sobre o Editor de Consultas do Amazon Redshift v2, acesse Carregar dados de exemplo no Guia de gerenciamento do Amazon Redshift.

Se você estiver trazendo seus próprios dados, e não carregando o conjunto de dados de amostra, consulte Etapa 3: Carregar os dados.

Etapa 3: Carregar os dados



Depois de iniciar o Redshift sem servidor, a etapa seguinte é carregar os dados para a POC. Se você estiver carregando um arquivo CSV simples, ingerindo dados semiestruturados do S3 ou transmitindo dados diretamente, o Amazon Redshift oferece a flexibilidade de mover os dados da fonte para as tabelas do Amazon Redshift com rapidez e facilidade.

Escolha um dos métodos a seguir para carregar os dados.

Carregar um arquivo local

Para agilizar a ingestão e análise, você pode usar o Editor de Consultas do Amazon Redshift v2 para carregar facilmente arquivos de dados do seu desktop local. Ele tem capacidade para processar arquivos em vários formatos, como CSV, JSON, AVRO, PARQUET, ORC e muito mais. Para permitir que seus usuários, como um administrador, carreguem dados de um desktop local usando o editor de consultas v2, é necessário especificar um bucket comum do Amazon S3 e a conta do usuário deve ser configurada com as permissões adequadas. Você pode acompanhar a publicação Data load made easy and secure in Amazon Redshift using Query Editor V2 para obter orientações detalhadas.

Carregar um arquivo do Amazon S3

Para carregar dados de um bucket do Amazon S3 no Amazon Redshift, primeiro use o comando COPY e especifique o local de origem do Amazon S3 e a tabela de destino do Amazon Redshift. Verifique se as permissões e os perfis do IAM estão configurados corretamente para permitir que o Amazon Redshift acesse o bucket designado do Amazon S3. Siga o Tutorial: Carregar dados do Amazon S3 para obter orientações detalhadas. Você também pode escolher a opção Carregar dados no editor de consultas v2 para carregar dados diretamente do bucket do S3.

Ingestão de dados contínua

A cópia automática (em versão de demonstração) é uma extensão do comando COPY e automatiza o carregamento contínuo de dados dos buckets do Amazon S3. Quando você cria um trabalho

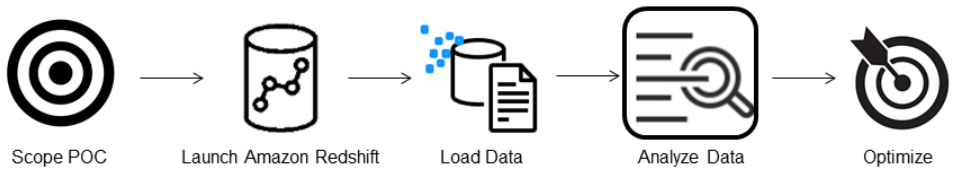
COPY, o Amazon Redshift detecta quando são criados arquivos do Amazon S3 em um caminho especificado e os carrega automaticamente sem sua intervenção. O Amazon Redshift monitora os arquivos carregados para verificar se eles são carregados apenas uma vez. Para obter instruções sobre como criar trabalhos de cópia, consulte COPY JOB

> **Note**
>
> No momento, a cópia automática está em versão de demonstração e só pode ser usada em clusters provisionados em específicas Regiões da AWS. Para criar um cluster de visualização prévia para cópia automática, consulte Criar uma integração de eventos do S3 para copiar automaticamente arquivos de buckets do Amazon S3.

Carregar dados de streaming

A ingestão de streaming fornece ingestão de dados de fluxo de baixa latência e alta velocidade do Amazon Kinesis Data Streams e do Amazon Managed Streaming for Apache Kafka no Amazon Redshift. A ingestão de streaming do Amazon Redshift usa uma visão materializada, que é atualizada diretamente do fluxo utilizando atualização automática. A exibição materializada mapeia para a fonte de dados da transmissão. Você pode executar filtragem e agregações nos dados de fluxo como parte da definição de visão materializada. Para obter orientações detalhadas sobre como carregar dados de um fluxo, consulte Conceitos básicos da ingestão de streaming do Amazon Kinesis Data Streams ou Conceitos básicos da ingestão de streaming do Amazon Managed Streaming para Apache Kafka.

Etapa 4: Analisar os dados



Depois de criar seu grupo de trabalho e namespace do Redshift sem servidor e carregar os dados, você pode executar consultas imediatamente abrindo o Editor de consultas v2 no painel de navegação do console do Redshift sem servidor. É possível usar o editor de consultas v2 para testar a funcionalidade de consulta ou o desempenho da consulta em seus próprios conjuntos de dados.

Realizar consultas com o Editor de Consultas do Amazon Redshift v2

É possível usar o editor de consultas v2 pelo console do Amazon Redshift. Consulte Simplify your data analysis with Amazon Redshift query editor v2 para obter um guia completo sobre como configurar, conectar e executar consultas com o editor de consultas v2.

Como alternativa, se você quiser executar um teste de carga como parte da POC, é possível fazê-lo seguindo as etapas a seguir para instalar e executar o Apache JMeter.

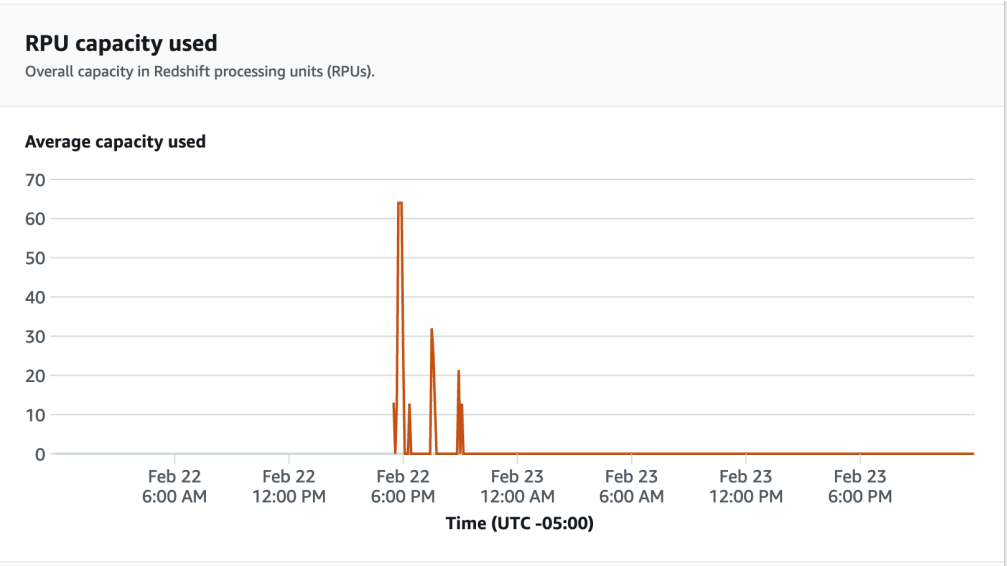
Executar um teste de carga usando o Apache JMeter

Para realizar um teste de carga e simular “N” usuários que estão enviando consultas simultaneamente ao Amazon Redshift, você pode usar o Apache JMeter, uma ferramenta de código aberto baseada em Java.

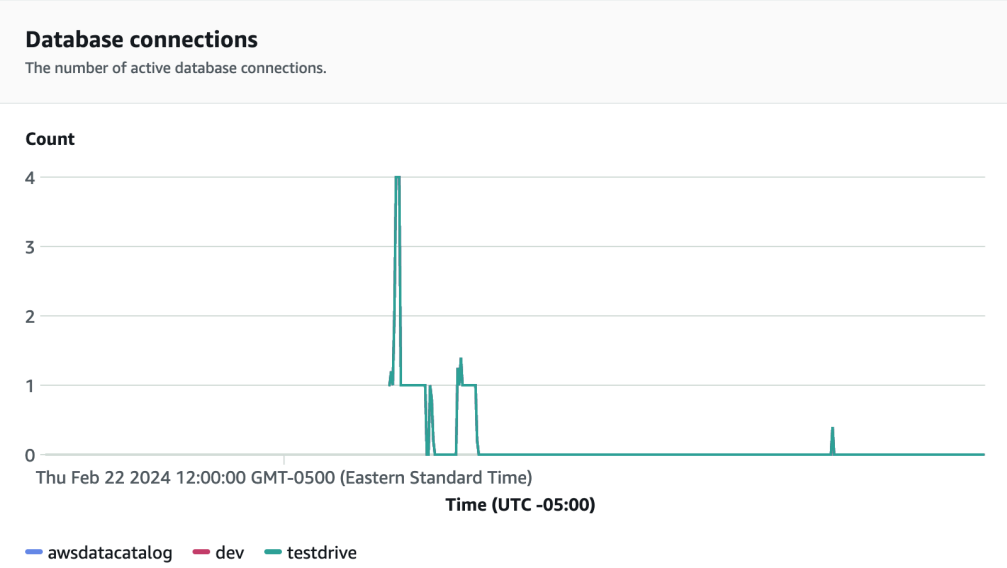
Para instalar e configurar o Apache JMeter para ser executado em seu grupo de trabalho do Redshift sem servidor, siga as instruções em Automate Amazon Redshift load testing with the AWS Analytics Automation Toolkit. Ele usa o kit de ferramentas AWS Analytics Automation (AAA), um utilitário de código aberto destinado a implantar dinamicamente as soluções do Redshift, para iniciar automaticamente esses recursos. Se você carregou seus próprios dados no Amazon Redshift, execute a opção Etapa 5: Personalizar SQL para que você forneça as instruções SQL apropriadas que gostaria de testar em suas tabelas. Teste cada uma dessas instruções SQL uma vez usando o editor de consultas v2 para garantir que elas sejam executadas sem erros.

Depois de concluir a personalização de suas instruções SQL e finalizar seu plano de teste, salve-o e execute-o em seu grupo de trabalho do Redshift sem servidor. Para monitorar o andamento do seu teste, abra o console do Redshift sem servidor, acesse Monitoramento de consultas e bancos de dados, escolha a guia Histórico de consultas e visualize informações sobre suas consultas.

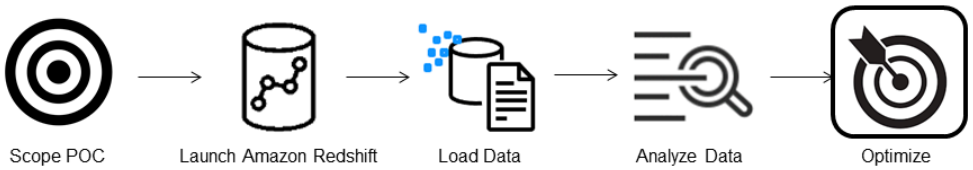
Para métricas de desempenho, escolha a guia Desempenho do banco de dados no console do Redshift sem servidor para monitorar métricas como Conexões de banco de dados e Utilização da CPU. Aqui você pode ver um grafo para monitorar a capacidade de RPU usada e observar como o Redshift sem servidor é escalado automaticamente para atender às demandas simultâneas de workload enquanto o teste de carga está sendo executado no grupo de trabalho.



As conexões de banco de dados são outra métrica útil a ser monitorada durante a execução do teste de carga para ver como o grupo de trabalho está lidando com várias conexões simultâneas em determinado momento e atender às crescentes demandas da workload.



Etapa 5: Otimizar



O Amazon Redshift possibilita que dezenas de milhares de usuários processem exabytes de dados todos os dias e potencializem suas workloads de análise. Para isso, ele oferece uma variedade de configurações e recursos para atender a casos de uso individuais. Ao escolher entre essas opções, os clientes procuram ferramentas que os ajudem a determinar a configuração ideal de data warehouse para atender à workload do Amazon Redshift.

Test Drive

Você pode usar o Test Drive para reproduzir automaticamente a workload existente em configurações possíveis e analisar as saídas correspondentes para avaliar o destino ideal para o qual migrar sua workload. Consulte Find the best Amazon Redshift configuration for your workload using Redshift Test Drive para obter informações sobre como usar o Test Drive para avaliar diferentes configurações do Amazon Redshift.

Práticas recomendadas do Amazon Redshift para projetar tabelas

À medida que você planeja seu banco de dados, certas decisões fundamentais relativas ao planejamento da tabela influenciarão fortemente a performance geral de consultas. Essas escolhas de design também têm um efeito significativo nos requisitos de armazenamento, que afetam a performance da consulta ao reduzir o número de operações de E/S e minimizar a memória necessária para processar consultas.

Nesta seção, você encontra um resumo das decisões de planejamento mais importantes e práticas recomendadas para otimizar a performance de consultas. O Otimização automática de tabelas fornece explicações mais detalhadas e exemplos de opções de design de tabela.

Tópicos

- Escolha a melhor chave de classificação
- Escolha o melhor estilo de distribuição
- Permitir que COPY selecione as codificações de compactação
- Definição das limitações da chave primária e chave estrangeira
- Uso do menor tamanho possível de coluna
- Uso dos tipos de dados de data/hora para colunas de data

Escolha a melhor chave de classificação

O Amazon Redshift armazena seus dados no disco em ordem classificada de acordo com a chave de classificação. O otimizador de consulta Amazon Redshift usa a ordem de classificação quando determina os planos de consulta ideais.

> **Note**
>
> Ao usar a otimização automática de tabela, você não precisa escolher a chave de classificação de sua tabela. Para obter mais informações, consulte Otimização automática de tabelas.

Seguem algumas sugestões para a melhor abordagem:

- Para que o Amazon Redshift escolha a ordem de classificação apropriada, especifique AUTO para a chave de classificação.
- Se dados recentes forem mais consultados, especifique a coluna de time stamp como a coluna principal da chave de classificação.

  As consultas são mais eficientes, pois podem ignorar blocos inteiros que estão fora do período.
- Se você fizer filtragem de intervalos frequentes ou filtragem de igualdade em uma coluna, especifique esta coluna como a chave de classificação.

  O Amazon Redshift pode ignorar a leitura de blocos inteiros de dados para essa coluna. É possível fazer isso, pois ele rastreia os valores mínimo e máximo da coluna armazenados em cada bloco e pode ignorar blocos que não se aplicam ao intervalo previsto.
- Se você costuma ingressar em uma tabela, especifique a coluna de união como a chave de classificação e a chave de distribuição.

  Isso permite que o otimizador de consulta escolha uma junção de mesclagem de classificação em vez de uma junção hash mais lenta. Como os dados já são classificados na chave de junção, o otimizador de consulta pode ignorar a fase de classificação da junção de mesclagem de classificação.

Escolha o melhor estilo de distribuição

Quando você executa uma consulta, o otimizador de consulta redistribui as linhas aos nós de computação conforme necessário para executar junções e agregações. O objetivo da seleção de um estilo de distribuição da tabela é minimizar o impacto da etapa de redistribuição colocando os dados no local em que eles precisam estar antes que a consulta seja executada.

> **Note**
>
> Quando você usa a otimização automática de tabelas, não é necessário escolher o estilo de distribuição da tabela. Para obter mais informações, consulte Otimização automática de tabelas.

Seguem algumas sugestões para a melhor abordagem:

1. Distribuir a tabela de fatos e uma tabela de dimensões nas colunas comuns.

   Sua tabela de fatos pode ter apenas uma chave de distribuição. Qualquer tabela que ingressar em outra chave não é disposta com a tabela de fatos. Escolha uma dimensão para disposição com base na frequência em que ela é ingressada e no tamanho das linhas de junção. Designe a chave primária da tabela de dimensões e a chave estrangeira correspondente da tabela de fatos como a DISTKEY.
2. Escolha a maior dimensão com base no tamanho do conjunto de dados filtrado.

   Como somente as linhas que são usadas na junção precisam ser distribuídas, considere o tamanho do conjunto de dados após a filtragem e não apenas o tamanho da tabela.
3. Escolha uma coluna com alta cardinalidade no conjunto de resultados filtrados.

   Se você distribui uma tabela de vendas em uma coluna de data, por exemplo, provavelmente você obterá uma distribuição de dados razoavelmente uniforme, a não ser que a maioria de suas vendas sejam sazonais. Entretanto, se você usa um predicado restrito por intervalo para filtrar um período de data curto, a maioria das linhas filtradas ocorrem em um conjunto limitado de fatias e o workload da consulta é distorcido.
4. Alterar algumas tabelas de dimensões para usar a distribuição ALL.

   Se uma tabela de dimensão não pode ser disposta com a tabela de fato ou outras tabelas de junção importantes, você pode melhorar a performance da consulta significativamente ao distribuir

toda a tabela para todos os nós. O uso de distribuição ALL multiplica as exigências de espaço de armazenamento e aumenta os tempos de carregamento e operações de manutenção, portanto você deve considerar todos os fatores antes de optar pela distribuição ALL.

Para que o Amazon Redshift escolha o estilo de distribuição apropriado, especifique `AUTO` para o estilo de distribuição.

Para obter mais informações sobre a escolha de estilos de distribuição, consulte Distribuição de dados para otimização de consultas.

Permitir que COPY selecione as codificações de compactação

Você pode especificar codificações de compactação ao criar uma tabela mas, na maioria dos casos, a compactação automática produz os melhores resultados.

ENCODE AUTO é o padrão para tabelas. Quando a tabela é definida como ENCODE AUTO, o Amazon Redshift gerencia automaticamente a codificação de compactação para todas as colunas da tabela. Para obter mais informações, consulte CRIAR TABELA e ALTER TABLE.

O comando COPY analisa seus dados e aplica codificações de compactação em uma tabela vazia automaticamente como parte da operação de carregamento.

A compactação automática equilibra a performance geral ao escolher codificações de compactação. As varreduras restritas por intervalo podem apresentar má performance se as colunas de chaves de classificação forem mais altamente compactadas do que outras colunas na mesma consulta. Como resultado, a compactação automática escolhe uma codificação menos eficiente para manter as colunas de chaves de classificação equilibradas com outras colunas.

Suponha que a chave de classificação da sua tabela seja uma data ou um time stamp e que a tabela use colunas varchar muito grandes. Nesse caso, você pode obter melhor performance descompactando a coluna de chave de classificação. Execute o comando ANALYZE COMPRESSION na tabela e use as codificações para criar uma nova tabela, mas omita a codificação de compactação para a chave de classificação.

Há um custo de performance pela codificação de compactação automática, mas apenas se a tabela estiver vazia e ainda não tiver codificação de compactação. Para tabelas de vida curta e tabelas que você cria com frequência, como tabelas de preparação, carregue a tabela uma vez com compactação automática ou execute o comando ANALYZE COMPRESSION. Em seguida, use essas codificações para criar novas tabelas. Você pode adicionar as codificações à instrução CRIAR TABELA ou usar CRIAR TABELA COMO para criar uma nova tabela com a mesma codificação.

Para obter mais informações, consulte Carregamento de tabelas com compactação automática.

Definição das limitações da chave primária e chave estrangeira

Defina as limitações da chave primária e estrangeira entre as tabelas sempre que apropriado. Embora elas sejam apenas informativas, o otimizador de consulta utiliza estas limitações para gerar planos de consulta mais eficientes.

Não defina limitações para a chave primária e chave estrangeira a menos que sua aplicação exija as limitações. O Amazon Redshift não impõe limitações exclusivas para a chave primária e chave estrangeira.

Consulte Restrições de tabela para obter mais informações sobre como o Amazon Redshift usa limitações.

Uso do menor tamanho possível de coluna

Não crie o hábito de usar o tamanho máximo de coluna por conveniência.

Em vez disso, considere os maiores valores que você provavelmente armazenará nas colunas e dimensione-as de acordo. Por exemplo, uma coluna CHAR para armazenar abreviações de estados e territórios dos EUA usada pelos correios só precisa ser CHAR(2).

Uso dos tipos de dados de data/hora para colunas de data

O Amazon Redshift armazena dados DATE e TIMESTAMP com mais eficiência do que CHAR ou VARCHAR, o que resulta em melhor performance de consulta. Use o tipo de dados DATA ou DATA/HORA, dependendo da solução necessária, em vez de um tipo de caractere ao armazenar informações de data/hora. Para obter mais informações, consulte Tipos de datetime.

Práticas recomendadas do Amazon Redshift para carregamento de dados

O carregamento de conjuntos de dados muito grandes pode levar tempo e consumir muitos recursos de computação. A forma como seus dados são carregados também pode afetar a performance da consulta. Esta seção apresenta as melhores práticas para carregar dados com eficiência usando comandos COPY, inserções em massa e tabelas de preparação.

Tópicos

- Saiba como carregar dados com um tutorial
- Uso de um comando COPY para carregar dados
- Uso de um único comando COPY para carregar a partir de vários arquivos
- Carregar arquivos de dados
- Compactar arquivos de dados
- Verificação de arquivos de dados antes de depois de um carregamento
- Uso de uma inserção de múltiplas linhas
- Uso de uma inserção em massa
- Carregamento de dados por ordem de chave de classificação
- Carregamento de dados em blocos sequenciais
- Uso de tabelas de séries temporais
- Agendamento em torno de janelas de manutenção

Saiba como carregar dados com um tutorial

Tutorial: Carregar dados do Amazon S3O orienta você do começo ao fim pelas etapas para carregar dados em um bucket do Amazon S3 e, em seguida, usar o comando COPY para carregar os dados nas suas tabelas. O tutorial inclui ajuda na solução de erros de carregamento e compara a diferença de performance entre o carregamento a partir de um único arquivo e o carregamento a partir de vários arquivos.

Uso de um comando COPY para carregar dados

O comando COPY carrega dados em paralelo do Amazon S3, Amazon EMR, Amazon DynamoDB ou várias origens dos dados em hosts remotos. COPY carrega grandes quantidades de dados de forma muito mais eficiente do que instruções de INSERIR e também armazena dados de forma mais eficaz.

Para obter mais informações sobre o uso do comando COPY, consulte Carregar dados do Amazon S3 e Carregar dados de uma tabela do Amazon DynamoDB.

Uso de um único comando COPY para carregar a partir de vários arquivos

O Amazon Redshift pode carregar automaticamente em paralelo com base em vários arquivos de dados compactados. Você pode especificar os arquivos a serem carregados usando um prefixo de objeto do Amazon S3 ou usando um arquivo manifesto.

Porém, se você usar vários comandos COPY simultâneos para carregar uma tabela de vários arquivos, o Amazon Redshift é forçado a executar um carregamento serializado. Este tipo de carregamento é muito mais lento e requer um processo de VACUUM ao final se a tabela tem uma coluna de classificação definida. Para obter mais informações sobre como usar COPY para carregar dados em paralelo, consulte Carregar dados do Amazon S3.

Carregar arquivos de dados

Os arquivos de dados de origem vêm em formatos diferentes e usam algoritmos de compactação variados. Ao carregar dados com o comando COPY, o Amazon Redshift carrega todos os arquivos referenciados pelo prefixo do bucket do Amazon S3. (O prefixo é uma string de caracteres no início do nome da chave do objeto.) Se o prefixo se referir a vários arquivos ou arquivos que podem ser divididos, o Amazon Redshift carregará os dados paralelamente, aproveitando a arquitetura MPP do Amazon Redshift. Isso divide a workload entre os nós no cluster. Por outro lado, quando você carrega dados de um arquivo que não pode ser dividido, o Amazon Redshift é forçado a executar um carregamento serializado, que é muito mais lento. As seções a seguir descrevem a forma recomendada de carregar diferentes tipos de arquivo no Amazon Redshift, dependendo do formato e da compactação.

Carregar dados de arquivos que podem ser divididos

Os seguintes arquivos podem ser divididos automaticamente quando seus dados são carregados:

- um arquivo CSV não compactado
- um arquivo em colunas (Parquet/ORC)

O Amazon Redshift divide automaticamente arquivos de 128 MB ou maiores em partes. Arquivos em colunas, especificamente Parquet e ORC, não serão divididos se tiverem menos de 128 MB. O Redshift usa fatias trabalhando em paralelo para carregar os dados. Isso fornece performance de carga rápida.

Carregar dados de arquivos que não podem ser divididos

Tipos de arquivo, como JSON ou CSV, quando compactados com outros algoritmos de compactação, como GZIP, não são divididos automaticamente. Desse modo, recomendamos dividir os dados manualmente em vários arquivos menores com tamanho semelhante, de 1 MB a 1 GB, após a compactação. Além disso, faça com que o número de arquivos seja um múltiplo do número

de fatias em seu cluster. Para obter mais informações sobre como dividir seus dados em arquivos e exemplos de como carregar dados com o comando COPY, consulte Carregar dados do Amazon S3.

Compactar arquivos de dados

Quando você desejar compactar arquivos de carga grandes, recomendamos que use gzip, lzop, bzip2 ou Zstandard para compactá-los e dividir os dados em vários arquivos menores.

Especifique a opção GZIP, LZOP, BZIP2 ou ZSTD com o comando COPY. Este exemplo carrega a tabela de HORA a partir de um arquivo lzop delimitado por pipe.

```
copy time
from 's3://amzn-s3-demo-bucket/data/timerows.lzo'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
lzop
delimiter '|';
```

Há casos em que não é necessário dividir arquivos de dados não compactados. Para obter mais informações sobre divisão de dados em arquivos e exemplos de como usar COPY para carregar dados, consulte Carregar dados do Amazon S3.

Verificação de arquivos de dados antes de depois de um carregamento

Antes de carregar dados do Amazon S3, primeiro verifique se o bucket do Amazon S3 contém todos os arquivos corretos e apenas esses arquivos. Para obter mais informações, consulte Verificação da presença dos arquivos corretos no bucket.

Após a conclusão da operação de carregamento, consulte a tabela de sistema STL_LOAD_COMMITS para certificar-se de que os arquivos esperados foram carregados. Para obter mais informações, consulte Como verificar se os dados foram carregados corretamente.

Uso de uma inserção de múltiplas linhas

Se um comando COPY não for uma opção e você precisar de inserções SQL, use uma inserção de múltiplas linhas sempre que possível. A compactação de dados é ineficiente quando você adiciona dados apenas uma linha ou algumas linhas por vez.

As inserções de múltiplas linhas melhora a performance ao agrupar uma série de inserções. O exemplo a seguir insere três linhas em uma tabela de quatro colunas usando uma única instrução de INSERT. Esta ainda é uma pequena inserção, exibida simplesmente para ilustrar a sintaxe de uma inserção de múltiplas linhas.

```
insert into category_stage values
(default, default, default, default),
(20, default, 'Country', default),
(21, 'Concerts', 'Rock', default);
```

Para obter mais detalhes e exemplos, consulte INSERT.

Uso de uma inserção em massa

Use uma operação de inserção em massa com uma cláusula SELECT para inserção de dados de alta performance.

Use os comandos INSERT e CREATE TABLE AS quando você precisar mover dados ou um subconjunto de dados de uma tabela para outra.

Por exemplo, a seguinte instrução INSERT seleciona todas as linhas da tabela CATEGORY e as insere na tabela CATEGORY_STAGE.

```
insert into category_stage
(select * from category);
```

O exemplo a seguir cria CATEGORY_STAGE como uma cópia de CATEGORIA e insere todas as linhas de CATEGORIA em CATEGORY_STAGE.

```
create table category_stage as
select * from category;
```

Carregamento de dados por ordem de chave de classificação

Carregue seus dados por ordem de chave de classificação para evitar a necessidade de vácuo.

Se cada lote de novos dados seguir as linhas existentes na tabela, e a operação COPY não for grande o suficiente para acionar determinadas otimizações de carga, seus dados serão adequadamente armazenados por ordem de classificação e você não precisará executar um vacuum. Você não precisa pré-classificar as linhas em cada carregamento, pois COPY classifica cada lote de dados de entrada durante o carregamento.

Por exemplo, suponha que você carregue dados diariamente com base nas atividades do dia atual. Se sua chave de classificação for uma coluna de carimbo de data/hora e a operação COPY

não for grande o suficiente para acionar determinadas otimizações de carga, seus dados serão armazenados em ordem de classificação. Essa ordem ocorre, pois os dados do dia atual são sempre anexados no final dos dados do dia anterior. Para obter mais informações, consulte Carregar os dados por ordem de chave de classificação. Para obter mais informações sobre operações de vacuum, consulte “Vacuum de tabelas”.

Carregamento de dados em blocos sequenciais

Se você precisa adicionar uma grande quantidade de dados, carregue os dados em blocos sequenciais de acordo com a ordem de classificação e elimine a necessidade de vácuo.

Por exemplo, suponha que você precise carregar uma tabela com eventos de janeiro de 2017 a dezembro de 2017. Supondo que cada mês está em um arquivo, carregue as linhas para janeiro, fevereiro e assim por diante. Sua tabela será completamente classificada quando seu carregamento for concluído e você não precisará executar um vacuum. Para obter mais informações, consulte Uso de tabelas de séries temporais.

Ao carregar conjuntos de dados muito grandes, o espaço necessário para classificar pode exceder o espaço total disponível. Ao carregar dados em blocos menores, você usará muito menos espaço intermediário de classificação durante cada carregamento. Além disso, o carregamento de blocos menores facilita a reinicialização se o comando COPY falhar e for revertido.

Uso de tabelas de séries temporais

Se seus dados têm um período de retenção fixo, você pode organizá-los como uma sequência de tabelas de séries temporais. Nessa sequência, cada tabela é idêntica, mas contém dados para diferentes períodos.

É possível remover dados antigos com facilidade simplesmente executando um comando DROP TABLE nas tabelas correspondentes. Essa abordagem é muito mais rápida do que o processo de executar um comando DELETE em grande escala e evita que você precise executar um processo VACUUM subsequente para recuperar espaço. Para ocultar o fato de que os dados estão armazenados em diferentes tabelas, você pode criar uma exibição UNION ALL. Ao excluir dados antigos, refina a exibição UNION ALL para remover as tabelas descartadas. Da mesma forma, à medida que carrega novos períodos de tempo em novas tabelas, adicione as novas tabelas à exibição. Para orientar o otimizador a ignorar a verificação nas tabelas que não correspondem ao filtro de consulta, sua definição de exibição filtra por intervalo de datas que corresponde a cada tabela.

Evite ter muitas tabelas na exibição UNION ALL. Cada tabela adicional adiciona um curto tempo de processamento à consulta. As tabelas não precisam usar o mesmo período. Por exemplo, você pode ter tabelas para períodos diferentes, como diariamente, mensalmente ou anualmente.

Se você usa tabelas de séries temporais com uma coluna de time stamp para a chave de classificação, você efetivamente carrega seus dados na ordem da chave de classificação. Isso elimina a necessidade de realizar vacuum para reclassificar os dados. Para obter mais informações, consulte Carregar os dados por ordem de chave de classificação.

Agendamento em torno de janelas de manutenção

Se uma manutenção programada ocorrer enquanto uma consulta estiver sendo executada, a consulta será encerrada e revertida e será necessário reiniciá-la. Programe operações de longa execução, tais como grandes carregamentos de dados ou operação VACUUM, de forma a evitar as janelas de manutenção. Você também pode minimizar o risco e facilitar reinicializações necessárias executando carregamentos de dados em incrementos menores e gerenciando o tamanho de suas operações VACUUM. Para obter mais informações, consulte Carregamento de dados em blocos sequenciais e Vacuum de tabelas.

Práticas recomendadas do Amazon Redshift para criar consultas

Para maximizar a performance das consultas, siga estas recomendações ao criá-las.

- Projete tabelas de acordo com as melhores práticas para oferecer uma base sólida para a performance da consulta. Para obter mais informações, consulte Práticas recomendadas do Amazon Redshift para projetar tabelas.
- Evite usar `select *`. Inclua apenas as colunas que você especificamente precisa.
- Projete consultas para minimizar a gravação temporária em disco. Quando uma consulta precisa de mais memória do que a disponível, o Amazon Redshift grava resultados intermediários em disco, o que retarda a consulta e pode afetar outras consultas. Reduza a gravação temporária em disco filtrando antecipadamente, selecionando menos colunas, pré-agregando e dividindo grandes consultas em etapas com tabelas temporárias. Para obter mais informações, consulte Aprimoramento do desempenho das consultas.
- Use um Expressão condicional CASE para executar agregações complexas em vez de selecionar várias vezes a partir da mesma tabela.
- Não use junções cruzadas a menos que absolutamente necessário. Essas junções sem uma condição de junção resultam no produto Cartesiano de duas tabelas. Uniões cruzadas são

normalmente executadas como uniões de loops aninhados, que são os tipos mais lentos de união possíveis.

- Use subconsultas em casos em que uma tabela na consulta é usada apenas para condições de predicado e a subconsulta retorna um pequeno número de linhas (menor que aproximadamente 200). O seguinte exemplo usa uma subconsulta para evitar a junção da tabela LISTING.

```
select sum(sales.qtysold)
from sales
where salesid in (select listid from listing where listtime > '2008-12-26');
```

- Use predicados para restringir o conjunto de dados tanto quanto possível.
- No predicado, use os operadores menos caros possíveis. Os operadores Condição de comparação são preferíveis a operadores LIKE. Operadores LIKE ainda são preferíveis a SIMILAR TO ou Operadores POSIX.
- Evite usar funções em predicados de consulta. Seu uso pode aumentar o custo da consulta ao exigir grandes números de linhas para resolver as etapas intermediárias da consulta.
- Se possível, use uma cláusula WHERE para restringir o conjunto de dados. O planejador de consulta pode, então, usar a ordem das linhas para ajudar a determinar quais registros correspondem aos critérios, portanto ele pode ignorar a varredura de um grande número blocos de disco. Sem isso, o mecanismo de execução da consulta deve fazer a varredura completa das colunas participantes.
- Adicione predicados para filtrar tabelas que participam em junções, mesmo se os predicados aplicam os mesmos filtros. A consulta retorna o mesmo conjunto de resultados, mas o Amazon Redshift é capaz de filtrar as tabelas de junção antes da etapa de verificação e pode, então, pular de forma eficiente os blocos de verificação dessas tabelas. Filtros redundantes não são necessários se você filtrar em uma coluna que é usada na condição de junção.

  Por exemplo, suponha que você deseje juntar `SALES` e `LISTING` para localizar tíquetes de venda nos tíquetes listados após dezembro, agrupados por vendedor. Ambas as tabelas são classificadas por data. A consulta a seguir junta as tabelas em sua chave comum e filtra valores `listing.listtime` maiores que 1º de dezembro.

```
select listing.sellerid, sum(sales.qtysold)
from sales, listing
where sales.salesid = listing.listid
and listing.listtime > '2008-12-01'
group by 1 order by 1;
```

A cláusula WHERE não inclui um predicado para `sales.saletime`, portanto o mecanismo de execução é forçado a varrer toda a tabela SALES. Se você sabe que o filtro resultará em menos linhas participando da junção, adicione este filtro também. O exemplo a seguir reduz o tempo de execução significativamente.

```
select listing.sellerid, sum(sales.qtysold)
from sales, listing
where sales.salesid = listing.listid
and listing.listtime > '2008-12-01'
and sales.saletime > '2008-12-01'
group by 1 order by 1;
```

- Use chaves de classificação na cláusula GROUP BY para que o planejador de consulta possa usar uma agregação mais eficiente. Uma consulta pode se qualificar para agregação de uma fase quando sua lista GROUP BY contiver apenas colunas de chave de classificação, onde uma delas também seja a chave de distribuição. As colunas de chave de classificação na lista GROUP BY devem incluir a primeira chave de classificação, seguida pelas outras chaves de classificação que você deseja usar por ordem de chave de classificação. Por exemplo, é válido usar a primeira chave de classificação, a primeira e segunda chaves de classificação, a primeira, segunda e terceira chaves de classificação e assim por diante. É inválido usar a primeira e a terceira chave de classificação.

  Você pode confirmar o uso da agregação de uma fase executando o comando EXPLAIN e procurando por `XN GroupAggregate` na etapa de agregação da consulta.

- Se você usar as duas cláusulas, GROUP BY e ORDER BY, verifique se as colunas estão na mesma ordem em ambas. Ou seja, use a abordagem logo a seguir.

  ```
  group by a, b, c
  order by a, b, c
  ```

  Não use a abordagem a seguir.

  ```
  group by b, c, a
  order by a, b, c
  ```

Trabalhar com recomendações do Amazon Redshift Advisor

Para ajudá-lo a melhorar a performance e diminuir os custos operacionais para seu cluster Amazon Redshift, o Amazon Redshift Advisor oferece recomendações específicas sobre mudanças a serem feitas. O Advisor desenvolve as recomendações personalizadas analisando métricas de performance e de uso para o cluster. Essas recomendações personalizadas relacionam-se às operações e configurações do cluster. Para ajudar você a priorizar as otimizações, o Advisor classifica as recomendações por ordem de impacto.

Como o Advisor funciona

O Advisor baseia suas recomendações em observações sobre estatísticas de performance ou dados de operações. O Advisor gera observações executando testes em clusters/grupos de trabalhos para determinar se um valor de teste está dentro de um intervalo especificado. Se o resultado do teste estiver fora desse intervalo, o Advisor gera uma observação para o cluster. Ao mesmo tempo, o Advisor cria uma recomendação sobre como fazer com que o valor observado retorne para o intervalo de melhores práticas.

Para arquiteturas de vários clusters que usam o Amazon Redshift Data Sharing, o Advisor agora fornece otimização aprimorada ao analisar padrões de workload em todos os clusters e grupos de trabalho da data mesh, incluindo clusters e grupos de trabalho em diferentes regiões. Quando você compartilha tabelas entre clusters/grupos de trabalho de produtor e consumidor, o Advisor coleta automaticamente os padrões de consulta de todos os endpoints de consumidor no data mesh, a menos que sejam explicitamente negados, e os associa a workloads de produtor para gerar recomendações mais eficazes. Isso significa que as otimizações de tabela, inclusive de chaves de classificação, de chaves de distribuição e de compactação, baseiam-se em como os dados estão realmente sendo usados em toda a organização, não apenas em um único cluster. O Advisor também oferece suporte ao Amazon Redshift sem servidor, mantendo automaticamente a continuidade da otimização nos ciclos de pausa e retomada.

Por exemplo, suponha que um data warehouse contenha tabelas com chaves de distribuição abaixo do ideal que causem distorção de dados nos nós de computação. Nesse caso, o Advisor recomenda automaticamente a redistribuição de tabelas usando o parâmetro DISTKEY para especificar uma coluna que distribua os dados uniformemente. Neste outro exemplo, suponha que o Advisor observe que o cluster tem tabelas sem chaves de classificação ou com definições de chave de classificação ineficientes que diminuam drasticamente o desempenho das consultas. Nesse caso, o Advisor oferece recomendações automaticamente para colunas de chave de classificação apropriadas com base em seus padrões de consulta para melhorar a filtragem de dados e reduzir a E/S de disco.

Otimizar arquiteturas de compartilhamento de dados

Quando você usa o compartilhamento de dados do Amazon Redshift para distribuir workloads em vários clusters/grupos de trabalho, o Advisor ajuda a otimizar o desempenho em todo o data mesh. Ele analisa automaticamente como as tabelas compartilhadas estão sendo consultadas em todos os clusters/grupos de trabalho de consumidor. Essa análise inclui entender quais colunas são filtradas com frequência, quais tabelas geralmente são unidas e como os dados estão sendo verificados. Ao considerar o panorama completo de uso de dados, o Advisor gera recomendações que melhoram o desempenho de todos os usuários de dados compartilhados.

Ao otimizar tabelas com base em padrões de uso em toda a organização, em vez de em um único cluster, é possível:

- Tomar decisões de otimização orientadas por dados com base nos padrões de acesso aos dados em todos os clusters/grupos de trabalho na malha.
- Reduzir os custos de armazenamento por meio de estratégias de compactação mais eficazes.
- Melhorar a utilização de recursos em toda o data mesh.

Regiões do Amazon Redshift em que o Advisor é oferecido

O recurso Amazon Redshift Advisor está disponível apenas nas seguintes regiões da AWS:

- Região Leste dos EUA (Norte da Virgínia) (us-east-1)
- Região Leste dos EUA (Ohio) (us-east-2)
- Região Oeste dos EUA (Norte da Califórnia) us-west-1
- Região Oeste dos EUA (Oregon) us-west-2
- Região da África (Cidade do Cabo) (af-south-1)
- Região da Ásia-Pacífico (Hong Kong) (ap-east-1)
- Região Ásia-Pacífico (Haiderabade) (ap-south-2)
- Região da Ásia-Pacífico (Jakarta) (ap-southeast-3)
- Região da Ásia-Pacífico (Melbourne) (ap-southeast-4)
- Região Ásia-Pacífico (Malásia) (ap-southeast-5)
- Região da Ásia-Pacífico (Mumbai) (ap-south-1)
- Região da Ásia-Pacífico (Osaka) (ap-northeast-3)

- Região da Ásia-Pacífico (Seul) (ap-northeast-2)
- Região da Ásia-Pacífico (Singapura) (ap-southeast-1)
- Região da Ásia-Pacífico (Sydney) (ap-southeast-2)
- Região da Ásia-Pacífico (Tóquio) (ap-northeast-1)
- Região do Canadá (Central) (ca-central-1)
- Região Oeste do Canadá (Calgary) (ca-west-1)
- Região da China (Pequim) (cn-north-1)
- Região da China (Ningxia) (cn-northwest-1)
- Região da Europa (Frankfurt) (eu-central-1)
- Região da Europa (Irlanda) (eu-west-1)
- Região da Europa (Londres) (eu-west-2)
- Região da Europa (Milão) (eu-south-1)
- Região da Europa (Paris) (eu-west-3)
- Região Europa (Espanha) (eu-south-2)
- Região da Europa (Estocolmo) (eu-north-1)
- Região Europa (Zurique) (eu-central-2)
- Região de Israel (Tel Aviv) (il-central-1)
- Região do Oriente Médio (Bahrein) (me-south-1)
- Região do Oriente Médio (EAU) (me-central-1)
- Região da América do Sul (São Paulo) (sa-east-1)

Tópicos

- Visualizar recomendações do Amazon Redshift Advisor
- Recomendações do Amazon Redshift Advisor

Visualizar recomendações do Amazon Redshift Advisor

Você pode acessar as recomendações do Amazon Redshift Advisor usando o console do Amazon Redshift, a API do Amazon Redshift ou a AWS CLI. Para acessar as recomendações, você deve ter a permissão `redshift:ListRecommendations` anexada ao seu perfil ou identidade do IAM.

Visualizar recomendações do Amazon Redshift Advisor no console provisionado do Amazon Redshift

Você pode visualizar as recomendações do Amazon Redshift Advisor no Console de gerenciamento da AWS.

Como visualizar as recomendações do Amazon Redshift Advisor para clusters do Amazon Redshift no console

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. No menu de navegação, escolha Advisor (Conselheiro).
3. Expanda cada recomendação ver mais detalhes. Nessa página, é possível classificar e agrupar as recomendações.

Visualizar recomendações do Amazon Redshift Advisor usando operações da API do Amazon Redshift

Você pode listar as recomendações do Amazon Redshift Advisor para clusters do Amazon Redshift usando a API do Amazon Redshift. Normalmente, você desenvolve uma aplicação na linguagem de programação de sua escolha para chamar a API `redshift:ListRecommendations` usando um SDK da AWS. Para obter mais informações, consulte ListRecommendations na Referência da API do Amazon Redshift.

Visualizar recomendações do Amazon Redshift Advisor usando operações da AWS Command Line Interface

Você pode listar as recomendações do Amazon Redshift Advisor para clusters do Amazon Redshift usando a AWS Command Line Interface. Para obter mais informações, consulte list-recommendations na Referência de comandos da AWS CLI.

Recomendações do Amazon Redshift Advisor

O Amazon Redshift Advisor oferece recomendações sobre como otimizar seu cluster Amazon Redshift para aumentar a performance e economizar nos custos operacionais. É possível encontrar explicações para cada recomendação no console, conforme descrito anteriormente. É possível encontrar mais detalhes sobre essas recomendações nas seções a seguir.

Tópicos

- Compactar objetos de arquivo do Amazon S3 carregados por COPY
- Isolar vários bancos de dados ativos
- Realocar a memória do Workload Management (WLM)
- Ignorar a análise de compactação durante o comando COPY
- Dividir objetos do Amazon S3 carregados por COPY
- Atualizar estatísticas da tabela
- Habilitar a aceleração de consultas breves
- Alterar chaves de distribuição em tabelas
- Alterar as chaves de classificação em tabelas
- Alterar codificações de compactação em colunas
- Recomendações de tipo de dados

Compactar objetos de arquivo do Amazon S3 carregados por COPY

O comando COPY tira proveito da arquitetura de processamento paralelo maciço (MPP) no Amazon Redshift para ler e carregar dados em paralelo. Ele pode ler arquivos do Amazon S3, tabelas do DynamoDB e saída de texto de um ou mais hosts remotos.

Ao carregar grandes quantidades de dados, é altamente recomendável usar o comando COPY para carregar arquivos de dados compactados do S3. A compactação de grandes conjuntos de dados economiza tempo no upload dos arquivos para o Amazon S3. COPY também pode agilizar o processo de carregamento ao descompactar os arquivos à medida que são lidos.

Análise

Comandos COPY de execução prolongada, que carregam grandes conjuntos de dados descompactados, geralmente têm a oportunidade de melhorar consideravelmente a performance. A análise do Advisor identifica comandos COPY que carregam grandes conjuntos de dados descompactados. Nesse caso, o Advisor gera uma recomendação para implementar a compactação nos arquivos de origem no Amazon S3.

Recomendação

Verifique se cada COPY carrega uma quantidade significativa de dados ou seja executado por um período significativo ingira objetos de dados compactados do Amazon S3. É possível identificar

os comandos COPY que carregam grandes conjuntos de dados descompactados do Amazon S3 executando o comando SQL a seguir como um superusuário.

```
SELECT
    wq.userid, query, exec_start_time AS starttime, COUNT(*) num_files,
    ROUND(MAX(wq.total_exec_time/1000000.0),2) execution_secs,
    ROUND(SUM(transfer_size)/(1024.0*1024.0),2) total_mb,
    SUBSTRING(querytxt,1,60) copy_sql
FROM stl_s3client s
JOIN stl_query q USING (query)
JOIN stl_wlm_query wq USING (query)
WHERE s.userid>1 AND http_method = 'GET'
    AND POSITION('COPY ANALYZE' IN querytxt) = 0
    AND aborted = 0 AND final_state='Completed'
GROUP BY 1, 2, 3, 7
HAVING SUM(transfer_size) = SUM(data_size)
AND SUM(transfer_size)/(1024*1024) >= 5
ORDER BY 6 DESC, 5 DESC;
```

Se os dados preparados permanecerem no Amazon S3 depois de carregá-los, o que é comum em arquiteturas de data lake, armazenar esses dados em um formato compactado poderá reduzir os custos de armazenamento.

Dicas de implementação

- O tamanho ideal do objeto é de 1 a 128 MB após a compressão.
- É possível compactar arquivos com o formato gzip, lzop ou bzip2.

Isolar vários bancos de dados ativos

Como prática recomendada, recomendamos isolar os bancos de dados no Amazon Redshift uns dos outros. As consultas são executadas em um banco de dado específico e não podem acessar dados de qualquer outro banco de dados no cluster. No entanto, as consultas executadas em todos os bancos de dados de um cluster compartilham o mesmo espaço de armazenamento os recursos de computação do cluster subjacente. Quando um único cluster contém vários bancos de dados ativos, seus workloads geralmente não são relacionados.

Análise

A análise do Advisor analisa todos os bancos de dados do cluster para workloads ativos executados ao mesmo tempo. Se houver workloads ativos em execução ao mesmo tempo, o Advisor gera uma recomendação para considerar a migração de bancos de dados para clusters separados do Amazon Redshift.

Recomendação

Considere mover cada banco de dados consultado ativamente para um cluster dedicado separado. Usar um cluster separado pode reduzir contenção de recursos e melhorar a performance de consulta. Isso é possível porque você pode definir o tamanho de cada cluster de acordo com as necessidades de armazenamento, custo e desempenho de cada workload. Além disso, as workloads não relacionadas geralmente se beneficiam de diferentes configurações de gerenciamento da workload.

Para identificar quais bancos de dados são usados ativamente, é possível executar este comando SQL como um superusuário.

```
SELECT database,
  COUNT(*) as num_queries,
  AVG(DATEDIFF(sec,starttime,endtime)) avg_duration,
  MIN(starttime) as oldest_ts,
  MAX(endtime) as latest_ts
FROM stl_query
WHERE userid > 1
GROUP BY database;
```

Dicas de implementação

- Como um usuário deve se conectar a cada banco de dados especificamente, e as consultas podem acessar somente um único banco de dados, mover bancos de dados para clusters separados tem um impacto mínimo para os usuários.
- Uma opção para mover um banco de dados é realizar as seguintes etapas:
  1. Restaurar temporariamente um snapshot do cluster atual para um cluster do mesmo tamanho.
  2. Excluir todos os bancos de dados do novo cluster, exceto o banco de dados de destino a ser movido.
  3. Redimensionar o cluster para uma contagem e um tipo de nó apropriados para o workload do banco de dados.

Realocar a memória do Workload Management (WLM)

O Amazon Redshift roteia consultas de usuário para Implementar o WLM manual para processamento. O gerenciamento de workload (WLM) define como essas consultas são roteadas para as filas. O Amazon Redshift aloca a cada fila uma parte da memória disponível do cluster. A memória de uma fila é dividida entre os slots de consulta da fila.

Quando uma fila é configurada com mais slot do que o necessário para o workload, a memória alocada para esses slots não utilizados é subutilizada. Reduzir os slots configurados para corresponder aos requisitos do workload de pico redistribui a memória subutilizada para slots ativos, e pode resultar na melhoria da performance de consulta.

Análise

A análise do Advisor analisa os requisitos de simultaneidade do workload para identificar filas de consulta com slots não utilizados. O Advisor gera uma recomendação para reduzir o número de slots em uma fila quando encontrar o seguinte:

- Uma fila com slots completamente inativos durante toda a análise.
- Uma fila com mais de quatro slots que tiveram pelo menos dois slots inativos durante toda a análise.

Recomendação

Reduzir os slots configurados para corresponder aos requisitos do workload de pico redistribui memória subutilizada para slots ativos. Considere reduzir a contagem de slots configurados para filas em que os slots nunca foram totalmente utilizados. Para identificar essas filas, é possível comparar os requisitos de slot por hora de pico para cada fila executando o seguinte comando SQL como um superusuário.

```
WITH
 generate_dt_series AS (select sysdate - (n * interval '5 second') as dt from (select
 row_number() over () as n from stl_scan limit 17280)),
 apex AS (
     SELECT iq.dt, iq.service_class, iq.num_query_tasks, count(iq.slot_count) as
 service_class_queries, sum(iq.slot_count) as service_class_slots
     FROM
         (select gds.dt, wq.service_class, wscc.num_query_tasks, wq.slot_count
```

```
        FROM stl_wlm_query wq
        JOIN stv_wlm_service_class_config wscc ON (wscc.service_class =
 wq.service_class AND wscc.service_class > 5)
        JOIN generate_dt_series gds ON (wq.service_class_start_time <= gds.dt AND
 wq.service_class_end_time > gds.dt)
        WHERE wq.userid > 1 AND wq.service_class > 5) iq
    GROUP BY iq.dt, iq.service_class, iq.num_query_tasks),
    maxes as (SELECT apex.service_class, trunc(apex.dt) as d, date_part(h,apex.dt) as
 dt_h, max(service_class_slots) max_service_class_slots
                    from apex group by apex.service_class, apex.dt,
 date_part(h,apex.dt))
SELECT apex.service_class - 5 AS queue, apex.service_class, apex.num_query_tasks AS
 max_wlm_concurrency, maxes.d AS day, maxes.dt_h || ':00 - ' || maxes.dt_h || ':59' as
 hour, MAX(apex.service_class_slots) as max_service_class_slots
FROM apex
JOIN maxes ON (apex.service_class = maxes.service_class AND apex.service_class_slots =
 maxes.max_service_class_slots)
GROUP BY  apex.service_class, apex.num_query_tasks, maxes.d, maxes.dt_h
ORDER BY apex.service_class, maxes.d, maxes.dt_h;
```

A coluna `max_service_class_slots` representa o número máximo de slots da consulta de WLM na fila de consulta para aquela hora. Se existirem filas subutilizadas, implemente a otimização de redução de slots modificando um grupo de parâmetros, conforme descrito no Guia de gerenciamento de clusters do Amazon Redshift.

Dicas de implementação

- Se o workload for altamente variável em volume, verifique se a análise capturou um período utilização de pico. Caso contrário, execute o SQL anterior repetidamente para monitorar os requisitos de simultaneidade de pico.
- Para obter mais detalhes sobre a interpretação dos resultados das consultas do código SQL anterior, consulte o script wlm_apex_hourly.sql script no GitHub.

Ignorar a análise de compactação durante o comando COPY

Quando você carrega dados em uma tabela vazia com codificação de compactação declarada com o comando COPY, o Amazon Redshift aplica compactação de armazenamento. Essa otimização garante que os dados no cluster sejam armazenados com eficiência, mesmo quando carregados por usuários finais. A análise necessária para a compactação pode levar um tempo significativo.

Análise

A análise do Advisor verifica operações COPY que foram atrasadas por análise de compactação automática. A análise determina as codificações de compactação com uma amostragem de dados durante o carregamento. Essa amostragem é semelhante àquela realizada pelo comando ANALYZE COMPRESSION.

Quando você carrega dados como parte de um processo estruturado, como em um lote noturno de extração, transformação e carregamento (ETL), é possível definir a compactação antecipadamente. Também é possível otimizar as definições de tabela para ignorar permanentemente essa fase sem nenhum impacto negativo.

Recomendação

Para melhorar a agilidade do comando COPY ignorando a fase de análise de compactação, implemente uma destas duas opções:

- Use o parâmetro da coluna `ENCODE` ao criar qualquer tabela carregada usando o comando COPY.
- Desative a compactação por completo fornecendo o parâmetro `COMPUPDATE OFF` no comando COPY.

A melhor solução geralmente é usar a codificação de coluna durante a criação da tabela, pois essa abordagem também mantém o benefício de armazenar dados compactados em disco. É possível usar o comando ANALYZE COMPRESSION para sugerir codificações de compactação, mas é necessário recriar a tabela para aplicar essas codificações. Para automatizar esse processo, você pode usar o AWSColumnEncodingUtility, encontrado no GitHub.

Para identificar as operações COPY que acionam análise de compactação automática, execute o seguinte comando SQL.

```
  WITH xids AS (
    SELECT xid FROM stl_query WHERE userid>1 AND aborted=0
    AND querytxt = 'analyze compression phase 1' GROUP BY xid
    INTERSECT SELECT xid FROM stl_commit_stats WHERE node=-1)
SELECT a.userid, a.query, a.xid, a.starttime, b.complyze_sec,
    a.copy_sec, a.copy_sql
FROM (SELECT q.userid, q.query, q.xid, date_trunc('s',q.starttime)
    starttime, substring(querytxt,1,100) as copy_sql,
```

```
    ROUND(datediff(ms,starttime,endtime)::numeric / 1000.0, 2) copy_sec
    FROM stl_query q JOIN xids USING (xid)
    WHERE (querytxt ilike 'copy %from%' OR querytxt ilike '% copy %from%')
    AND querytxt not like 'COPY ANALYZE %') a
LEFT JOIN (SELECT xid,
    ROUND(sum(datediff(ms,starttime,endtime))::numeric / 1000.0,2) complyze_sec
    FROM stl_query q JOIN xids USING (xid)
    WHERE (querytxt like 'COPY ANALYZE %'
    OR querytxt like 'analyze compression phase %')
    GROUP BY xid ) b ON a.xid = b.xid
WHERE b.complyze_sec IS NOT NULL ORDER BY a.copy_sql, a.starttime;
```

Dicas de implementação

- Verifique se todas as tabelas de tamanho significativo criadas durante seus processos de ETL, como tabelas de preparação e tabelas temporárias, declaram uma codificação de compactação para todas as colunas, exceto a primeira chave de classificação.
- Estime o tamanho esperado de vida útil da tabela que está sendo carregada para cada um dos comandos COPY identificados pelos comandos SQL anteriores. Se tiver certeza de que a tabela permanecerá muito pequena, desative a compactação por completo com o parâmetro `COMPUPDATE OFF`. Caso contrário, crie a tabela com a compactação explícita antes de carregá-la com o comando COPY.

Dividir objetos do Amazon S3 carregados por COPY

O comando COPY tira proveito da arquitetura de processamento paralelo massivo (MPP) no Amazon Redshift para ler e carregar dados de arquivos no Amazon S3. O comando COPY carrega os dados em paralelo a partir de vários arquivos, dividindo o workload entre os nós em seu cluster. Para atingir a taxa de transferência ideal, é altamente recomendável que você divida seus dados em vários arquivos para aproveitar o processamento paralelo.

Análise

A análise do Advisor identifica os comandos COPY que carregam grandes conjuntos de dados contidos em um pequeno número de arquivos preparados no Amazon S3. Comandos COPY de execução prolongada, que carregam grandes conjuntos de dados de poucos arquivos, geralmente têm a oportunidade de melhorar consideravelmente a performance. Quando o Advisor identifica que esses comandos COPY estão levando uma quantidade significativa de tempo, ele cria uma

recomendação para aumentar o paralelismo dividindo os dados em arquivos adicionais no Amazon S3.

Recomendação

Nesse caso, recomendamos as seguintes ações, listada em ordem de prioridade:

1. Otimize os comandos COPY que carregam menos arquivos que o número de nós do cluster.
2. Otimize os comandos COPY que carregam menos arquivos que o número de fatias do cluster.
3. Otimize os comandos COPY em que o número de arquivos não seja um múltiplo do número de fatias do cluster.

Certos comandos COPY carregam uma quantidade significativa de dados ou são executados por um período significativo. Para esses comandos, recomendamos que você carregue uma série de objetos de dados do Amazon S3 que seja equivalente a um múltiplo do número de fatias do cluster. Para identificar quantos objetos do S3 cada comando COPY carregou, execute o seguinte código SQL como um superusuário.

```
SELECT
    query, COUNT(*) num_files,
    ROUND(MAX(wq.total_exec_time/1000000.0),2) execution_secs,
    ROUND(SUM(transfer_size)/(1024.0*1024.0),2) total_mb,
    SUBSTRING(querytxt,1,60) copy_sql
FROM stl_s3client s
JOIN stl_query q USING (query)
JOIN stl_wlm_query wq USING (query)
WHERE s.userid>1 AND http_method = 'GET'
    AND POSITION('COPY ANALYZE' IN querytxt) = 0
    AND aborted = 0 AND final_state='Completed'
GROUP BY query, querytxt
HAVING (SUM(transfer_size)/(1024*1024))/COUNT(*) >= 2
ORDER BY CASE
WHEN COUNT(*) < (SELECT max(node)+1 FROM stv_slices) THEN 1
WHEN COUNT(*) < (SELECT COUNT(*) FROM stv_slices WHERE node=0) THEN 2
ELSE 2+((COUNT(*) % (SELECT COUNT(*) FROM stv_slices))/(SELECT COUNT(*)::DECIMAL FROM
 stv_slices))
END, (SUM(transfer_size)/(1024.0*1024.0))/COUNT(*) DESC;
```

Dicas de implementação

- O número de fatias em um nó depende do tamanho do nó do cluster. Para obter mais informações sobre o número de fatias nos vários tipos de nó, consulte “Clusters e nós no Amazon Redshift” no Guia de gerenciamento de clusters do Amazon Redshift.
- Você pode carregar vários arquivos especificando um prefixo comum ou prefixo de chave para o conjunto ou listando explicitamente os arquivos em um arquivo manifesto. Para obter mais informações o carregamento de arquivos, consulte Carregar dados de arquivos compactados e não compactados.
- O Amazon Redshift não leva em conta o tamanho do arquivo ao dividir o workload. Divida seus arquivos de dados de carregamento para que os arquivos tenham o mesmo tamanho aproximado, entre 1 MB e 1 GB após a compactação.

Atualizar estatísticas da tabela

O Amazon Redshift usa um otimizador de consulta baseado em custo para escolher o plano de execução ideal para consultas. As estimativas de custos são baseadas em estatísticas da tabela obtidas usando o comando ANALYZE. Quando as estatísticas estão desatualizadas ou ausentes, o banco de dados escolhe um plano menos eficiente para a execução de consulta, principalmente para consultas complexas. Manter estatísticas atuais ajuda que a execução de consultas complexas ocorra no mínimo tempo possível.

Análise

A análise do Advisor monitora tabelas cujas estatísticas estão desatualizadas ou ausentes. Ela revisa os metadados de acesso da tabela associados a consultas complexas. Se as tabelas que são geralmente acessadas com padrões complexos não tiverem estatísticas, o Advisor criará uma recomendação crucial para executar ANALYZE. Se as tabelas que são geralmente acessadas com padrões complexos tiverem estatísticas desatualizadas, o Advisor criará uma recomendação sugerida para executar ANALYZE.

Recomendação

Sempre que o conteúdo da tabela for alterado significativamente, atualize as estatísticas com ANALYZE. Recomendamos executar ANALYZE sempre que um número significativo de novas linhas de dados for carregado para uma tabela existente com os comandos COPY ou INSERT. Também recomendamos executar ANALYZE sempre que um número significativo de linhas for modificado com comandos UPDATE ou DELETE. Para identificar as tabelas com estatísticas ausentes ou

atualizadas, execute o seguinte comando SQL como um superusuário. Os resultados são ordenados da tabela maior para a menor.

Para identificar as tabelas com estatísticas ausentes ou atualizadas, execute o seguinte comando SQL como um superusuário. Os resultados são ordenados da tabela maior para a menor.

```
SELECT
   ti.schema||'.'||ti."table" tablename,
   ti.size table_size_mb,
   ti.stats_off statistics_accuracy
 FROM svv_table_info ti
 WHERE ti.stats_off > 5.00
 ORDER BY ti.size DESC;
```

Dicas de implementação

O limite ANALYZE padrão é 10 por cento. Esse padrão significa que o comando ANALYZE ignorará uma determinada tabela se menos de 10 por cento das linhas que ela contém tiverem sofrido alterações desde o último ANALYZE. Como resultado, você pode escolher executar comandos ANALYZE ao final de cada processo de ETL. Optar por essa abordagem significa que o comando ANALYZE será ignorado frequentemente, mas também garante que ele seja executado quando for necessário.

As estatísticas do comando ANALYZE têm maior impacto para colunas usadas em junções (por exemplo, `JOIN tbl_a ON col_b`) ou como predicados (por exemplo, `WHERE col_b = 'xyz'`). Por padrão, o comando ANALYZE coleta estatísticas para todas as colunas na tabela especificada. Se necessário, é possível reduzir o tempo necessário para executar o comando ANALYZE executando-o somente para as colunas onde tem o maior impacto. É possível executar o seguinte comando SQL para identificar colunas usadas como predicados. Você também pode permitir que o Amazon Redshift escolha quais colunas analisar especificando `ANALYZE PREDICATE COLUMNS`.

```
WITH predicate_column_info as (
SELECT ns.nspname AS schema_name, c.relname AS table_name, a.attnum as col_num,
 a.attname as col_name,
        CASE
            WHEN 10002 = s.stakind1 THEN array_to_string(stavalues1, '||')
            WHEN 10002 = s.stakind2 THEN array_to_string(stavalues2, '||')
            WHEN 10002 = s.stakind3 THEN array_to_string(stavalues3, '||')
            WHEN 10002 = s.stakind4 THEN array_to_string(stavalues4, '||')
```

```
            ELSE NULL::varchar
        END AS pred_ts
   FROM pg_statistic s
   JOIN pg_class c ON c.oid = s.starelid
   JOIN pg_namespace ns ON c.relnamespace = ns.oid
   JOIN pg_attribute a ON c.oid = a.attrelid AND a.attnum = s.staattnum)
SELECT schema_name, table_name, col_num, col_name,
       pred_ts NOT LIKE '2000-01-01%' AS is_predicate,
       CASE WHEN pred_ts NOT LIKE '2000-01-01%' THEN (split_part(pred_ts,
 '||',1))::timestamp ELSE NULL::timestamp END as first_predicate_use,
       CASE WHEN pred_ts NOT LIKE '%||2000-01-01%' THEN (split_part(pred_ts,
 '||',2))::timestamp ELSE NULL::timestamp END as last_analyze
FROM predicate_column_info;
```

Para obter mais informações, consulte Análise de tabelas.

Habilitar a aceleração de consultas breves

A aceleração de consultas breves (SQA) prioriza as consultas de curta execução sobre as consultas de execução demorada. A SQA executa consultas breves em um espaço dedicado, de maneira que as consultas SQA não sejam forçadas a esperar em filas atrás de consultas mais demoradas. O SQA apenas prioriza consultas que são de execução curta e estão em uma fila definida pelo usuário. Com a SQA, consultas breves são iniciadas com mais rapidez e os usuários veem os resultados mais cedo.

Se você ativar a SQA, poderá reduzir ou eliminar as filas de gerenciamento de workload (WLM) que são dedicadas à execução de consultas breves. Além disso, as consultas demoradas não precisam disputar slots em uma fila com as consultas breves, e assim você pode configurar suas filas do WLM usando menos slots de consulta. Quando você usa um nível de simultaneidade menor, a taxa de transferência de consultas aumenta e a performance geral do sistema melhora na maioria dos workloads. Para obter mais informações, consulte Aceleração de consulta breve.

Análise

O Advisor verifica os padrões de workload e relata o número de consultas recentes em que a SQA reduziria a latência e o tempo de fila diário de consultas qualificadas para SQA.

Recomendação

Modificar a configuração de WLM para ativar a SQA. O Amazon Redshift usa um algoritmo de Machine Learning para analisar cada consulta qualificada. As previsões serão melhores, pois a SQA

aprenderá com os padrões das suas consulta. Para obter mais informações, consulte Configurar filas do Workload Management.

Ao ativar a SQA, o WLM define o tempo máximo do ambiente de tempo de execução de consultas breves como dinâmico, por padrão. Recomendamos manter a configuração dinâmica para o tempo de execução máximo de SQA.

Dicas de implementação

Para conferir se a SQA está ativada, execute a consulta a seguir. Se a consulta retornar uma linha, a SQA está ativada.

```
select * from stv_wlm_service_class_config
where service_class = 14;
```

Para obter mais informações, consulte Monitoramento da SQA.

Alterar chaves de distribuição em tabelas

O Amazon Redshift distribui as linhas da tabela em todo o cluster de acordo com o estilo de distribuição da tabela. As tabelas com distribuição de CHAVE precisam de uma coluna como a chave de distribuição (DISTKEY). Uma linha da tabela é atribuída a uma fatia do nó de um cluster com base no valor da coluna DISTKEY.

Uma DISTKEY apropriada coloca uma quantidade semelhante de linhas e cada fatia do nó e costuma ser mencionada em condições de união. Uma união otimizada ocorre quando as tabelas são unidas nas colunas DISTKEY, acelerando a performance da consulta.

Análise

O Advisor analisa o workload do cluster para identificar a chave de distribuição mais apropriada para as tabelas que podem se beneficiar significativamente de um estilo de distribuição de CHAVE.

Recomendação

O Advisor fornece instruções ALTER TABLE que alteram o DISTSTYLE e a DISTKEY de uma tabela com base em sua análise. Para obter um benefício de performance significativo, garanta a implementação de todas as instruções SQL em um grupo de recomendação.

A redistribuição de uma tabela grande com ALTER TABLE consome recursos do cluster e requer bloqueios de tabela temporários em vários momentos. Implemente cada grupo de recomendação quando o outro workload do cluster for leve. Para ver mais detalhes sobre como otimizar as

propriedades de distribuição da tabela, consulte o Manual de design de tabela avançada do Amazon Redshift Engineering: estilos de distribuição e chaves de distribuição.

Para obter mais informações sobre ALTER DISTSYLE e DISTKEY, consulte ALTER TABLE.

> **Note**
>
> Se você não vir uma recomendação, não significa necessariamente que os estilos de distribuição atuais sejam os mais apropriados. O Advisor não fornece recomendações quando não há dados suficientes ou o benefício esperado da redistribuição é pequeno.
> As recomendações do Advisor se aplicam a uma tabela específica e não necessariamente se aplicam a uma tabela que contém uma coluna com o mesmo nome. As tabelas que têm um nome de coluna em comum podem ter características diferentes para essas colunas, a não ser que os dados dentro das tabelas sejam os mesmos.
> Se você vir recomendações para preparar tabelas que são criadas ou enviadas por tarefas ETL, modifique seus processos ETL para usar as chaves de distribuição recomendadas pelo Advisor.

Alterar as chaves de classificação em tabelas

O Amazon Redshift classifica as linhas da tabela de acordo com a chave de classificação da tabela. A classificação de linhas da tabela tem como base os valores da coluna da chave de classificação.

Classificar uma tabela em uma chave de classificação apropriada pode acelerar a performance de consultas, especialmente aquelas com predicados restritos por intervalos, exigindo que menos blocos de tabela sejam lidos do disco.

Análise

O Advisor analisa a workload do cluster ao longo de vários dias para identificar uma chave de classificação que beneficie as suas tabelas.

Recomendação

O Advisor fornece dois grupos de instruções ALTER TABLE que alteram a chave de classificação de uma tabela com base na análise dele:

- Instruções que alteram uma tabela que atualmente não tem uma chave de classificação para adicionar uma chave de classificação COMPOUND.

- Instruções que alteram uma chave de classificação de INTERLEAVED para COMPOUND ou nenhuma chave de classificação.

  Utilizar chaves de classificação composta reduz consideravelmente a sobrecarga de manutenção. Tabelas com chaves de classificação compostas não precisam das caras operações VACUUM REINDEX que são necessárias para tipos intercalados. Na prática, as chaves de classificação compostas são mais eficazes do que as chaves de classificação intercaladas para a grande maioria dos workloads do Amazon Redshift. No entanto, se a tabela for pequena, é mais eficiente não ter uma chave de classificação para evitar sobrecarga de armazenamento de chaves de classificação.

Quando uma tabela grande é classificada com a instrução ALTER TABLE, são consumidos recursos de cluster e são necessários bloqueios de tabela em vários momentos. Implemente cada recomendação quando o workload de um cluster for moderado. Mais detalhes sobre como otimizar as configurações de chave de classificação de tabela podem ser encontrados no Manual de design de tabela avançada do Amazon Redshift Engineering: chaves de classificação compostas e intercaladas.

Para obter mais informações sobre a ALTER SORTKEY, consulte ALTER TABLE.

> **Note**
>
> Se você não vir uma recomendação para uma tabela, não significa necessariamente que a configuração atual seja a melhor. O Advisor não fornece recomendações quando não há dados suficientes ou o benefício esperado da classificação é pequeno.
> As recomendações do Advisor se aplicam a uma tabela específica e não necessariamente se aplicam a uma tabela que contém uma coluna com o mesmo nome ou tipo de dados. As tabelas que compartilham nomes de coluna podem ter recomendações diferentes com base nos dados nas tabelas e no workload.

Alterar codificações de compactação em colunas

A compactação é uma operação em nível de coluna que reduz o tamanho dos dados quando são armazenados. A compactação é usada no Amazon Redshift para economizar espaço de armazenamento e melhorar a performance da consulta, reduzindo a quantidade de E/S de disco. Recomendamos uma codificação de compactação ideal para cada coluna com base em seu tipo de

dados e em padrões de consulta. Com a compactação ideal, as consultas podem ser executadas de forma mais eficiente e o banco de dados pode ocupar espaço mínimo de armazenamento.

Análise

O Advisor realiza a análise do workload e do esquema de banco de dados do cluster continuamente para identificar a codificação de compactação ideal para cada coluna da tabela.

Recomendação

O Advisor fornece instruções ALTER TABLE que alteram a codificação de compactação de colunas específicas, com base na análise dele.

Alterando codificações de compactação de coluna com ALTER TABLE consome recursos do cluster e requer bloqueios de tabela em vários momentos. É melhor implementar recomendações quando o workload do cluster for leve.

Para referência, o Exemplos de ALTER TABLE mostra várias instruções que alteram a codificação de uma coluna.

> **Note**
>
> O Advisor não fornece recomendações quando não há dados suficientes ou o benefício esperado de alterar a codificação é pequeno.

Recomendações de tipo de dados

O Amazon Redshift tem uma biblioteca de tipos de dados SQL para vários casos de uso. Entre eles, incluem-se tipos inteiros, como `INT`, e tipos para armazenar personagens, como `VARCHAR`. O Redshift armazena tipos de modo otimizado para garantir acesso rápido e boa performance nas consultas. Além disso, o Redshift fornece funções para tipos específicos, que você pode usar para formatar ou executar cálculos usando resultados de consultas.

Análise

O Advisor realiza a análise da workload e do esquema de banco de dados do cluster continuamente para identificar as colunas que podem se beneficiar significativamente de uma alteração de tipo de dado.

Recomendação

O Advisor fornece uma instrução `ALTER TABLE` que adiciona uma nova coluna com o tipo de dado sugerido. Uma instrução complementar `UPDATE` copia os dados da coluna existente para a nova coluna. Depois de criar a nova coluna e carregar os dados, altere suas consultas e scripts de ingestão para acessar a nova coluna. Depois, aproveite os recursos e as funções especializadas para o novo tipo de dado, encontrados em Referência de funções SQL.

Copiar os dados existentes para a nova coluna pode levar algum tempo. Recomendamos que você implemente cada recomendação do Advisor quando a workload do cluster for leve. Consulte a lista de tipos de dados disponíveis em Tipos de dados.

O Advisor não fornece recomendações quando não há dados suficientes ou quando o benefício esperado da alteração de tipo de dado é pequeno.

Tutoriais para o Amazon Redshift

Siga as etapas desses tutoriais para saber mais sobre os recursos do Amazon Redshift:

Tutorial: Carregar dados do Amazon S3	Neste tutorial, você acompanhará o processo completo para carregar dados de arquivos de dados presentes em um bucket do S3 em tabelas de banco de dados do Amazon Redshift.
Tutorial: Consultar dados aninhados com o Amazon Redshift Spectrum	Neste tutorial, o Redshift Spectrum é usado para consultar dados aninhados. Com o Redshift Spectrum, é possível consultar dados nos formatos de arquivo Parquet, ORC, JSON ou Ion.
Tutorial: Configuração de filas de gerenciamento do workload (WLM) manual	Neste tutorial, você configurará o Amazon Redshift para usar filas do gerenciamento de workloads (WLM) manual. O Amazon Redshift usa filas do WLM para gerenciar como os recursos devem ser divididos para executar consultas simultâneas. Você deve configurar o Amazon Redshift para usar o WLM manual se precisar usar várias filas do WLM.
Tutorial: Uso de funções SQL espaciais com Amazon Redshift	Neste tutorial, você consultará dados usando funções espaciais. As funções espaciais são usadas para consultar dados geométricos e geográficos.
Tutoriais para o Amazon Redshift ML	Nestes tutoriais, você criará e usará modelos de machine learning.
Tutorial: Criar funções e consultar com o RBAC	Neste tutorial, você usará o controle de acesso por perfil (RBAC) para gerenciar permissões em um banco de dados criado por você. Com o RBAC, você pode criar perfis com permissõe

	s específicas, como somente leitura ou leitura e gravação. Em seguida, você pode atribuir esses perfis aos usuários para conceder a eles as permissões especificadas.

Otimização automática de banco de dados

O Amazon Redshift hospeda um conjunto de recursos automatizados, denominados coletivamente como operações autônomas, que aprimoram o desempenho, reduzem a manutenção manual e otimizam o uso de recursos. As operações autônomas utilizam machine learning e processos em segundo plano para gerenciar operações do banco de dados com eficiência, automatizando várias tarefas de manutenção de rotina para reduzir a carga de trabalho dos administradores de banco de dados.

A seguinte tabela detalha os recursos autônomos do Amazon Redshift:

Recurso autônomo	Descrição
Classificação automática de vacuum	O Amazon Redshift reorganiza automatic amente os dados das tabelas com base nos padrões de consulta observados para obter uma ordem de classificação ideal. Esse recurso está habilitado por padrão para tabelas com chaves de classificação especificadas. Para obter mais informações, consulte Classificação automática de tabela.
Exclusão de vacuum automática	O Amazon Redshift executa operações de limpeza automaticamente para recuperar espaço das linhas excluídas e classificar dados. Para ter mais informações sobre operações automáticas de vacuum, consulte Exclusão de vacuum automática.
Otimização automática de tabelas	O Amazon Redshift monitora o desempenh o das consultas e os metadados de tabela para determinar automaticamente as melhores chaves de classificação e distribuição para tabelas e escolhe o tipo de compactação aplicado a uma coluna de valores de dados à medida que são adicionados dados a uma tabela. Para obter mais informações, consulte

Recurso autônomo	Descrição
	Otimização automática de tabelas e Codificações de compactação.
Análise automática	O Amazon Redshift analisa automaticamente as tabelas à medida que os dados dentro delas mudam, garantindo que o planejador de consultas tenha informações atualizadas para planos de execução ideais. Para ter mais informações sobre operações de análise automáticas, consulte Análise automática.
Visões materializadas automatizadas	O Amazon Redshift cria e atualiza visões materializadas automaticamente com base nos padrões de consulta observados. Isso reduz a necessidade de os usuários criarem ou atualizarem manualmente as visões para se beneficiarem de respostas mais rápidas às consultas. Para ter informações sobre visões materializadas, consulte Visões materializadas no Amazon Redshift.

Esses recursos autônomos estão habilitados por padrão e são executados automaticamente em segundo plano durante períodos de baixo tráfego para otimizar o desempenho do cluster. Se desejar, consulte Valores de parâmetros padrão no Guia de gerenciamento do Amazon Redshift para configurar recursos automáticos.

Para clusters ou grupos de trabalho com tráfego alto e contínuo, recomendamos habilitar recursos computacionais adicionais para realizar a otimização contínua. Para obter mais informações, consulte Alocação de recursos extras de computação para otimização automática de banco de dados.

Tópicos

- Alocação de recursos extras de computação para otimização automática de banco de dados
- Faturamento por operações autônomas
- Métricas de uso de operações autônomas

Alocação de recursos extras de computação para otimização automática de banco de dados

As operações autônomas melhoram o desempenho das consultas no cluster e ajudam a gerenciar os custos e a otimização do armazenamento. É possível optar por alocar recursos computacionais adicionais para garantir que os recursos de autogerenciamento sejam executados de forma consistente, mesmo durante períodos de tráfego alto, para obter os benefícios da otimização. Observe que as operações autônomas executadas com esses recursos extras são faturáveis. Para ter mais informações sobre o faturamento de operações autônomas, consulte Faturamento por operações autônomas e Preços do Amazon Redshift.

Se recursos extras de computação para operações autônomas forem desabilitados, o Amazon Redshift suspenderá temporariamente essas operações durante períodos de alta carga do sistema a fim de minimizar o impacto em workloads simultâneas, até que haja recursos suficientes para executá-las sem afetar negativamente as consultas dos usuários, o que pode afetar o desempenho.

Gerenciar recursos extras de computação para operações autônomas

Por padrão, recursos extras de computação para operações autônomas estão desabilitados tanto para clusters provisionados quanto para grupos de trabalho sem servidor, com exceção dos grupos de trabalho sem servidor com ajuste de escala e otimização orientados por IA. Para grupos de trabalho sem servidor com ajuste de escala e otimização orientados por IA, o Amazon Redshift sem servidor habilita e gerencia automaticamente recursos de computação adicionais para operações autônomas, com base em seu perfil de relação preço/performance.

Você pode optar por habilitar recursos extras de computação durante ou após a criação do cluster ou do grupo de trabalho.

Gerenciar recursos extras de computação para operações autônomas em clusters provisionados usando o console do Amazon Redshift

Ao criar um cluster, é possível habilitar recursos extras de computação para operações autônomas escolhendo Habilitar computação extra no painel Configuração de operações autônomas. Para fazer isso:

- Selecione Criar cluster.
- No painel Configuração de operações autônomas, escolha Habilitar computação extra.
- Você criará um cluster com computação extra habilitada.

Para ter mais informações, consulte Criar um cluster no Guia de gerenciamento de clusters do Amazon Redshift.

Para habilitar recursos extras de computação depois de criar um cluster:

- Acesse o cluster no console do Amazon Redshift.
- Na página de detalhes do cluster, selecione Editar configuração de operações autônomas no menu suspenso Ações.
- Escolha Habilitar computação extra.

Gerenciar operações autônomas de computação extra em clusters provisionados usando a AWS CLI

Para criar um cluster provisionado com recursos extras de computação para operações autônomas, use o comando `create-cluster` com a opção `extra-compute-for-automatic-optimization`.

```
aws redshift create-cluster --cluster-identifier <value> \
--node-type <value> \
--cluster-type <value> \
--master-username <value> \
--master-user-password <value> \
--extra-compute-for-automatic-optimization \
--region <value>
```

Para habilitar recursos extras de computação para operações autônomas em um cluster provisionado existente, use o comando `modify-cluster` com a opção `extra-compute-for-automatic-optimization`.

```
aws redshift modify-cluster --cluster-identifier <value> \
--extra-compute-for-automatic-optimization \
--region <value>
```

Para desabilitar recursos extras de computação para operações autônomas em um cluster provisionado existente, use o comando `modify-cluster` com a opção `no-extra-compute-for-automatic-optimization`.

```
aws redshift modify-cluster --cluster-identifier <value> \
--no-extra-compute-for-automatic-optimization \
--region <value>
```

Gerenciar operações autônomas de computação extra para grupos de trabalho sem servidor usando o console do Amazon Redshift

Ao criar um grupo de trabalho, é possível habilitar recursos extras de computação para operações autônomas escolhendo Habilitar computação extra no painel Configuração de operações autônomas.

- Escolha Create workgroup (Criar grupo de trabalho).
- Na Etapa 1, no painel Configuração de operações autônomas, escolha Habilitar computação extra.
- Você criará um grupo de trabalho sem servidor com computação extra habilitada.

> **Note**
>
> A opção Habilitar computação extra pode não estar disponível se você tiver selecionado uma meta de relação preço/performance. Isso ocorre porque o Amazon Redshift sem servidor habilita e gerencia automaticamente recursos extras de computação para operações autônomas em grupos de trabalho com otimização e ajuste de escala orientados por IA com base em seu perfil de relação preço/performance.

Para habilitar recursos extras de computação depois de criar um grupo de trabalho:

- Acesse o grupo de trabalho no console do Amazon Redshift sem servidor.
- Na página de detalhes do grupo de trabalho, selecione Editar configuração de operações autônomas no menu suspenso Ações.
- Escolha Habilitar computação extra.

> **Note**
>
> A opção Editar configuração de operações autônomas poderá estar desativada se o grupo de trabalho tiver uma meta de relação preço/performance definida. Isso ocorre porque o Amazon Redshift sem servidor habilita e gerencia automaticamente recursos extras de

computação para operações autônomas em grupos de trabalho com otimização e ajuste de escala orientados por IA com base em seu perfil de relação preço/performance.

Gerenciar operações autônomas de computação extra para grupos de trabalho sem servidor usando a AWS CLI

Para criar um grupo de trabalho sem servidor com recursos extras de computação alocados para operações autônomas, use o comando `create-workgroup` com a opção `extra-compute-for-automatic-optimization`.

```
aws redshift-serverless create-workgroup --base-capacity <value> \
--namespace-name <value> \
--workgroup-name <value> \
--extra-compute-for-automatic-optimization \
--region <value>
```

Para habilitar recursos extras de computação para operações autônomas em um grupo de trabalho sem servidor existente, use o comando `update-workgroup` com a opção `--extra-compute-for-automatic-optimization`.

```
aws redshift-serverless update-workgroup \
--workgroup-name <value> \
--extra-compute-for-automatic-optimization \
--region <value>
```

Para desabilitar recursos extras de computação para operações autônomas em um grupo de trabalho sem servidor existente, use o comando `update-workgroup` com a opção `--no-extra-compute-for-automatic-optimization`.

```
aws redshift-serverless update-workgroup \
--workgroup-name <value> \
--no-extra-compute-for-automatic-optimization \
--region <value>
```

Controle de custos de operações autônomas que usam recursos extras de computação

Controle de custos de operações autônomas de computação extra em clusters provisionados

Por meio do console do Amazon Redshift, é possível definir limites de uso em recursos extras de computação usados para operações autônomas em clusters provisionados. Os recursos extras de computação usados para operações autônomas não são contabilizados em relação a nenhum limite que você possa ter definido para escalabilidade simultânea e vice-versa.

Para criar um limite de uso de computação extra por meio do console:

- Acesse a página de detalhes do cluster e selecione a guia Manutenção no menu de navegação do cluster.
- Role para baixo e escolha Criar limites de uso. A página Gerenciar limites de uso será aberta.
- Na página Gerenciar limites de uso, localize a seção intitulada Limite de uso de computação extra para otimização automática e selecione Adicionar limite.

Para criar um limite de uso de computação extra usando a AWS CLI, use o comando `create-usage-limit` com o tipo de recurso `extra-compute-for-automatic-optimization`. O parâmetro `--limit-type` deve ser definido como `time` para esse tipo de recurso.

```
aws redshift create-usage-limit \
--cluster-identifier <value> \
--feature-type extra-compute-for-automatic-optimization \
--limit-type time \
--period <value> \
--amount <value> \
--breach-action <value>
```

Os parâmetros `--period` e `--breach-action` aceitam os mesmos valores dos outros limites de uso. Para ter mais informações, consulte Definir um limite de uso no Guia de gerenciamento de clusters do Amazon Redshift.

Controle de custos de operações autônomas de computação extra em grupos de trabalho sem servidor

O Amazon Redshift sem servidor oferece recursos para definir limites de uso para grupos de trabalho sem servidor. Consulte Definir limites de uso, inclusive limites de RPU no Guia de gerenciamento do Amazon Redshift. O uso de operações autônomas de computação extra no Amazon Redshift sem servidor está incluído no uso geral sem servidor, assim como as consultas de usuário.

Por meio do console do Amazon Redshift sem servidor, é possível ver quantos segundos de RPU o grupo de trabalho sem servidor está usando para operações autônomas de computação extra. Para fazer isso:

- Acesse o console do Amazon Redshift sem servidor.
- Selecione Monitoramento de recursos na barra lateral.
- Selecione um grupo de trabalho.
- Role para baixo até o grafo Segundos cobrados de computação extra para otimizações automáticas.

Para ter mais informações, consulte Console do Amazon Redshift sem servidor no Guia de gerenciamento do Amazon Redshift.

Também é possível verificar a visualização do sistema SYS_SERVERLESS_USAGE para encontrar o `charged_extra_compute_for_automatic_optimization_seconds`, que registra os segundos acumulados de unidade de computação (RPU) cobrados para otimizações automáticas.

Faturamento por operações autônomas

Por padrão, o Amazon Redshift não cobrará pelos recursos usados para operações autônomas. Se você optar por alocar recursos extras de computação para ter operações autônomas mais consistente, o Amazon Redshift cobrará somente pelas operações autônomas que realmente usarem recursos adicionais, como clusters de escalabilidade simultânea (ou RPUs adicionais), que são alocados somente quando os clusters principais ou as RPUs básicas são totalmente utilizados ao executar workloads de usuário. Para obter mais informações, consulte Alocação de recursos extras de computação para otimização automática de banco de dados e Escalabilidade simultânea..

Faturamento de operações autônomas para clusters provisionados

Em clusters provisionados, o faturamento de operações autônomas segue a lógica abaixo:

Tipo de operação autônoma	Comportamento do faturamento
O Amazon Redshift executa uma operação autônoma no cluster principal.	O Amazon Redshift não cobra pela operação autônoma.
O Amazon Redshift executa uma operação autônoma no cluster de escalabilidade simultânea.	O Amazon Redshift cobra pela operação autônoma.

Para obter mais informações, consulte Preços do Amazon Redshift.

Faturamento de operações autônomas para grupos de trabalho sem servidor

Em grupos de trabalho sem servidor, o faturamento de operações autônomas segue a lógica abaixo:

Tipo de operação autônoma	Comportamento do faturamento
O Amazon Redshift executa uma consulta do usuário e, enquanto isso, conclui uma operação autônoma.	O Amazon Redshift não cobra pela operação autônoma.
O Amazon Redshift precisa alocar capacidade computacional adicional para executar operações autônomas durante períodos de alta carga do sistema.	O Amazon Redshift cobra pela capacidade computacional adicional usada na execução de operações autônomas.
O Amazon Redshift começa a executar uma operação autônoma enquanto nenhuma consulta do usuário está sendo executada. O Amazon Redshift só fará isso se você tiver habilitado recursos extras de computação para operações autônomas.	O Amazon Redshift cobra pela operação autônoma.

Você só recebe cobrança uma vez pela sobreposição de atividades de workloads de usuário ou de operações autônomas. Por exemplo, suponha que o Amazon Redshift inicie uma operação

autônoma enquanto nenhuma consulta de usuário está sendo executada e uma consulta de usuário comece durante essa operação. Nesse caso, o faturamento é aplicado uma vez para o intervalo durante o qual o grupo de trabalho permanece ativo.

Para ter mais informações, consulte Faturamento do Amazon Redshift sem servidor no Guia de gerenciamento do Amazon Redshift.

Métricas de uso de operações autônomas

Se você habilitou recursos extras de computação para otimizações automáticas, é possível ver informações sobre o uso faturado de operações autônomas usando o console do Amazon Redshift. Para ter mais informações sobre métricas de operações autônomas no console para grupos de trabalho sem servidor, consulte a seção de gerenciamento de recursos no console do Amazon Redshift sem servidor no Guia de gerenciamento do Amazon Redshift.

As seguintes tabelas de sistema contêm métricas de uso sobre operações autônomas:

- SYS_EXTRA_COMPUTE_FOR_AUTOMATIC_OPTIMIZATION: use SYS_EXTRA_COMPUTE_FOR_AUTOMATIC_OPTIMIZATION para ver por quanto tempo o Amazon Redshift executa operações autônomas de clusters provisionados em um determinado período de uso.
- SYS_AUTOMATIC_OPTIMIZATION: use SYS_AUTOMATIC_OPTIMIZATION para ver informações detalhadas sobre operações autônomas de clusters provisionados e grupos de trabalho sem servidor.
- SYS_SERVERLESS_USAGE: use SYS_SERVERLESS_USAGE para ver por quanto tempo o Amazon Redshift executa operações autônomas de grupos de trabalho sem servidor em um determinado período de uso.

Otimização automática de tabelas

Otimização automática de tabelas é um recurso de auto-ajuste que otimiza automaticamente o design de tabelas aplicando chaves de classificação e distribuição sem a necessidade de intervenção do administrador. Usando a automação para ajustar o design de tabelas, você pode começar a usar e obter a performance mais rápida sem investir tempo para ajustar manualmente e implementar otimizações de tabelas.

A otimização automática de tabelas observa continuamente como as consultas interagem com tabelas. Ele usa métodos avançados de inteligência artificial para escolher chaves de classificação e distribuição para otimizar a performance do workload do cluster. Se o Amazon Redshift determinar que a aplicação de uma chave melhora a performance do cluster, as tabelas serão alteradas automaticamente em poucas horas a partir do momento em que o cluster foi criado, com impacto mínimo nas consultas.

Para aproveitar essa automação, um administrador do Amazon Redshift cria uma nova tabela ou altera uma tabela existente para habilitá-la a usar a otimização automática. Tabelas existentes com um estilo de distribuição ou chave de classificação de `AUTO` já estão habilitados para automação. Quando você executa consultas nessas tabelas, o Amazon Redshift determina se uma chave de classificação ou uma chave de distribuição melhorará a performance. Em caso afirmativo, o Amazon Redshift modificará automaticamente a tabela sem exigir a intervenção do administrador. Se um número mínimo de consultas for executado, as otimizações serão aplicadas dentro de horas após o cluster ser iniciado.

Se o Amazon Redshift determinar que uma chave de distribuição melhora a performance das consultas, tabelas onde o estilo de distribuição é `AUTO` pode ter seu estilo de distribuição alterado para `KEY`.

Tópicos

- Habilitar, desabilitar e monitorar a otimização automática de tabelas
- Gerenciar exclusões de workload do Autonomics
- Compactação de colunas para reduzir o tamanho dos dados armazenados
- Distribuição de dados para otimização de consultas
- Chaves de classificação
- Restrições de tabela

Habilitar, desabilitar e monitorar a otimização automática de tabelas

Por padrão, as tabelas criadas sem definir explicitamente chaves de classificação ou chaves de distribuição são definidas como AUTO. No momento da criação da tabela, você também pode definir explicitamente uma chave de classificação ou distribuição manualmente. Se você definir a chave de classificação ou distribuição, a tabela não será gerenciada automaticamente.

Habilitar a otimização automática de tabelas

Para permitir que uma tabela existente seja otimizada automaticamente, use as opções de instrução ALTER para alterar a tabela para AUTO. Você pode optar por definir a automação para chaves de classificação, mas não para chaves de distribuição (e vice-versa). Se você executar uma instrução ALTER para converter uma tabela para ser uma tabela automatizada, chaves de classificação existentes e estilos de distribuição serão preservados.

```
ALTER TABLE table_name ALTER SORTKEY AUTO;
```

```
ALTER TABLE table_name ALTER DISTSTYLE AUTO;
```

Para obter mais informações, consulte ALTER TABLE.

Inicialmente, uma tabela não tem chave de distribuição ou chave de classificação. O estilo de distribuição é definido como EVEN ou ALLdependendo do tamanho da tabela. À medida que a tabela cresce em tamanho, o Amazon Redshift aplica as chaves de distribuição e as chaves de classificação ideais. As otimizações são aplicadas dentro de horas após um número mínimo de consultas serem executadas. Ao determinar otimizações de chave de classificação, o Amazon Redshift tenta otimizar os blocos de dados lidos do disco durante uma varredura de tabela. Ao determinar otimizações de estilo de distribuição, o Amazon Redshift tenta otimizar o número de bytes transferidos entre nós de cluster.

Remover a otimização automática de tabelas de uma tabela

Você pode remover uma tabela da otimização automática. A remoção de uma tabela da automação envolve a seleção de uma chave de classificação ou estilo de distribuição. Para alterar o estilo de distribuição, especifique um estilo de distribuição específico.

```
ALTER TABLE table_name ALTER DISTSTYLE EVEN;
```

```
ALTER TABLE table_name ALTER DISTSTYLE ALL;
```

```
ALTER TABLE table_name ALTER DISTSTYLE KEY DISTKEY c1;
```

Para alterar uma chave de classificação, você pode definir uma chave de classificação ou escolher nenhuma.

```
ALTER TABLE table_name ALTER SORTKEY(c1, c2);
```

```
ALTER TABLE table_name ALTER SORTKEY NONE;
```

Monitorar a otimização automática de tabelas

A visualização do sistema SVV_ALTER_TABLE_RECOMMENDATIONS registra as recomendações atuais do Amazon Redshift Advisor para tabelas. Essa exibição mostra recomendações para todas as tabelas, aquelas que são definidas para otimização automática e aquelas que não são.

Para visualizar se uma tabela está definida para otimização automática, consulte a visualização do sistema SVV_TABLE_INFO. As entradas aparecem somente para tabelas visíveis no banco de dados da sessão atual. Recomendações são inseridas na exibição duas vezes por dia, começando em horas a partir do momento em que o cluster foi criado. Após uma recomendação estar disponível, ela começa dentro de uma hora. Depois que uma recomendação for aplicada (pelo Amazon Redshift ou por você), ela não aparecerá mais na visualização.

A visualização do sistema SVL_AUTO_WORKER_ACTION mostra um log de auditoria de todas as ações realizadas pelo Amazon Redshift e o estado anterior da tabela.

A visualização do sistema SVV_TABLE_INFO lista todas as tabelas no sistema, juntamente com uma coluna para indicar se a chave de classificação e o estilo de distribuição da tabela estão definidos como AUTO.

Para obter mais informações sobre essas visualizações de sistema, consulte Monitoramento do sistema (somente provisionado).

Gerenciar exclusões de workload do Autonomics

Você pode impedir que endpoints provisionados específicos ou grupos de trabalho sem servidor influenciem decisões autônomas, como chave de distribuição e chave de classificação, por meio de um recurso de negação de lista. Isso permite que você controle quais workloads o Amazon Redshift considera ao tomar decisões de otimização para seus dados do Redshift Managed Storage (RMS).

Usar a lista de negação

Você pode gerenciar a lista de negação por meio da seção sobre o Autonomics no console do Amazon Redshift:

1. Adicionar ou remover itens

   Adicione endpoints provisionados específicos ou grupos de trabalho sem servidor à lista de negação e remova-os quando necessário.

2. Visualizar e pesquisar

   Visualize todos os itens da lista de negação e pesquise endpoints ou grupos de trabalho específicos nessa lista.

A lista de negação é particularmente útil quando você administra marketplaces de dados, compartilha dados com usuários externos ou tem várias unidades de negócios, nas quais você deseja evitar que determinados padrões de uso influenciem as decisões de otimização adequadas à sua workload. Por exemplo, se o grupo de trabalho A executar a workload A e o grupo B executar a workload B em uma tabela compartilhada T, a chave de classificação de T será determinada pelas workloads A e B. Se você quiser que somente a workload A influencie a decisão da chave de classificação, adicione o grupo de trabalho B à lista de negação do endpoint ou grupo de trabalho que possui a tabela T. Por padrão, o Amazon Redshift Autonomics considera os padrões de consulta do produtor e de todos clusters/grupos de trabalho de consumidor, a menos que sejam especificamente excluídos por meio da lista de negação.

> **Note**
>
> Você pode negar recursos em diferentes regiões da AWS na mesma conta. O bloqueio por lista de negação entre contas não é aceito.

Gerenciar recursos da lista de negação no console do Amazon Redshift

No console do Amazon Redshift Serverless, conclua as seguintes etapas:

1. Escolha clusters ou grupos de trabalho sem servidor.
2. Navegue até a página de detalhes de um cluster ou grupo de trabalho específico.
3. Escolha Autonomics na seção de guias.
4. Na guia Autonomics, você pode visualizar e gerenciar sua lista de negação.
5. Para gerenciar a lista de negação entre regiões, escolha a região da AWS correspondente.

Adicionar recursos de lista de negação

1. Navegue até a guia Autonomics do cluster ou do grupo de trabalho selecionado, selecione a região da AWS e escolha Adicionar recursos.
2. Selecione um ou mais clusters provisionados ou grupos de trabalho sem servidor para adicionar à lista de negação e escolha Adicionar.
3. A tabela mostra a lista de recursos da lista de negação.

Remover recursos de lista de negação

1. Navegue até a guia Autonomics do cluster ou do grupo de trabalho selecionado e escolha a região da AWS.
2. Escolha o cluster ou o grupo de trabalho que você deseja excluir da lista e selecione Remover.
3. A caixa de diálogo de confirmação é aberta. Selecione Remover para confirmar.

Compactação de colunas para reduzir o tamanho dos dados armazenados

A compactação é uma operação em nível de coluna que reduz o tamanho dos dados quando eles são armazenados. A compactação economiza espaço de armazenamento e reduz o tamanho dos dados que são lidos a partir do armazenamento, que reduz a quantidade de E/S de disco e, portanto, melhora a performance da consulta.

ENCODE AUTO é o padrão para tabelas. Quando a tabela é definida como ENCODE AUTO, o Amazon Redshift gerencia automaticamente a codificação de compactação para todas as colunas da tabela. Para obter mais informações, consulte CRIAR TABELA e ALTER TABLE.

Porém, se você especificar a codificação de compactação para qualquer coluna da tabela, a tabela não será mais definida como ENCODE AUTO. O Amazon Redshift não gerencia mais automaticamente a codificação de compactação para todas as colunas da tabela.

Você pode aplicar um tipo de compactação ou codificação para as colunas em uma tabela manualmente quando você cria a tabela. Ou pode usar o comando COPY para analisar e aplicar compactação automaticamente. Para obter mais informações, consulte Permitir que COPY selecione as codificações de compactação. Para obter detalhes sobre a aplicação de compactação automática, consulte Carregamento de tabelas com compactação automática.

> **Note**
>
> Recomendamos o uso do comando COPY para aplicar a compactação automática.

Você pode optar por aplicar codificações de compactação manualmente se a nova tabela compartilhar as mesmas características de dados que outra tabela. Ou pode fazê-lo se ao testar você descobrir que as codificações de compactação aplicadas durante a compactação automática não são a melhor opção para seus dados. Se você escolher aplicar as codificações de compactação manualmente, poderá executar o comando ANALYZE COMPRESSION em uma tabela já povoada e usar os resultados para escolher as codificações de compactação.

Para aplicar a compactação manualmente, especifique as codificações de compactação para colunas individuais como parte da instrução CREATE TABLE. A sintaxe é a seguinte.

```
CREATE TABLE table_name (column_name
data_type ENCODE encoding-type)[, ...]
```

Aqui, encoding-type é retirado da tabela de palavras-chave na seção a seguir.

Por exemplo, a seguinte instrução cria uma tabela de duas colunas, PRODUCT. Quando os dados são carregados na tabela, a coluna PRODUCT_ID não é compactada, mas a coluna de PRODUCT_NAME é compactada usando a codificação do dicionário de bytes (BYTEDICT).

```
create table product(
```

```
product_id int encode raw,
product_name char(20) encode bytedict);
```

Você pode especificar a codificação para uma coluna quando ela é adicionada a uma tabela usando o comando ALTER TABLE.

```
ALTER TABLE table-name ADD [ COLUMN ] column_name column_type ENCODE encoding-type
```

Tópicos

- Codificações de compactação
- Teste de codificações de compactação

Codificações de compactação

Uma codificação de compactação especifica o tipo de compactação que é aplicado a uma coluna de valores de dados conforme as linhas são adicionadas a uma tabela.

ENCODE AUTO é o padrão para tabelas. Quando a tabela é definida como ENCODE AUTO, o Amazon Redshift gerencia automaticamente a codificação de compactação para todas as colunas da tabela. Para obter mais informações, consulte CRIAR TABELA e ALTER TABLE.

Porém, se você especificar a codificação de compactação para qualquer coluna da tabela, a tabela não será mais definida como ENCODE AUTO. O Amazon Redshift não gerencia mais automaticamente a codificação de compactação para todas as colunas da tabela.

Quando você usa CREATE TABLE, ENCODE AUTO é desativada ao especificar a codificação de compactação para qualquer coluna da tabela. Se ENCODE AUTO estiver desativada, o Amazon Redshift atribuirá automaticamente uma codificação de compactação a colunas para as quais você não especificar um tipo ENCODE, da seguinte maneira:

- Colunas que são definidas como chaves de classificação são designadas a compactação RAW.
- Colunas que são definidas como tipos de dados BOOLEAN, REAL ou DOUBLE PRECISION recebem a compactação RAW.
- As colunas definidas como tipos de dados SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIMESTAMP ou TIMESTAMPTZ recebem a compactação AZ64.
- As colunas definidas como tipos de dados CHAR ou VARCHAR recebem a compactação LZO.

Você pode alterar a codificação de uma tabela depois de criá-la usando ALTER TABLE. Se você desabilitar ENCODE AUTO usando ALTER TABLE, o Amazon Redshift deixará de gerenciar automaticamente as codificações de compactação de suas colunas. Todas as colunas manterão os tipos de codificação de compactação que tinham quando você desativou ENCODE AUTO até que sejam alteradas ou até que ENCODE AUTO seja reativada.

O Amazon Redshift permite as seguintes codificações de compactação:

Raw

A codificação raw é a codificação padrão para as colunas que são designadas como chaves de classificação e colunas definidas como tipo de dados BOOLEAN, REAL ou DOUBLE PRECISION. Com a codificação raw, os dados são armazenados em formato bruto, descompactado.

AZ64

AZ64 é um algoritmo de codificação de compactação proprietário projetado pela Amazon para atingir uma alta taxa de compactação e processamento de consulta aprimorado. No seu núcleo, o algoritmo AZ64 compacta grupos menores de valores de dados e usa instruções SIMD (Single Instruction, Multiple Data – instrução única, vários dados) para processamento paralelo. Use o AZ64 para obter uma economia significativa de armazenamento e alta performance para tipos de dados numéricos, de data e hora.

É possível usar o AZ64 como a codificação de compactação ao definir colunas com as instruções CREATE TABLE e ALTER TABLE com os seguintes tipos de dados:

- SMALLINT
- INTEGER
- BIGINT
- DECIMAL
- DATE
- TIMESTAMP
- TIMESTAMPTZ

Byte-dictionary

Na codificação do dicionário de bytes, um dicionário separado de valores exclusivos é criado para cada bloco de valores de coluna em disco. (Um bloco de disco do Amazon Redshift ocupa 1 MB.)

O dicionário contém até 256 valores de um byte que são armazenados como índices para os valores de dados originais. Se mais do que 256 valores forem armazenados em um único bloco, os valores adicionais serão gravados no bloco em formato bruto, descompactado. O processo é repetido para cada bloco de disco.

Essa codificação é muito eficaz em colunas de strings de baixa cardinalidade. Esta codificação é ótima quando o domínio de dados de uma coluna é menor do que 256 valores exclusivos.

Para colunas com o tipo de dado string (CHAR e VARCHAR) codificado com BYTEDICT, o Amazon Redshift executa varreduras vetorizadas e avaliações de predicados que operam diretamente sobre dados compactados. Essas varreduras usam instruções SIMD (instrução única, vários dados) específicas do hardware para processamento paralelo. Isso acelera significativamente a varredura de colunas de strings. A codificação do dicionário de bytes é especialmente eficiente em termos de espaço se uma coluna CHAR/VARCHAR contém strings de caracteres longas.

Suponha que uma tabela tenha uma coluna COUNTRY com um tipo de dados CHAR(30). Conforme os dados são carregados, o Amazon Redshift cria o dicionário e preenche a coluna COUNTRY com o valor do índice. O dicionário contém os valores exclusivos indexados e a tabela em si contém somente as subscrições de um byte dos valores correspondentes.

> **Note**
>
> Os espaços em branco são armazenados para colunas de caracteres de comprimento fixo. Portanto, em uma coluna CHAR(30), cada valor compactado economiza 29 bytes de armazenamento quando você usa a codificação do dicionário de bytes.

A tabela a seguir representa o dicionário para a coluna COUNTRY.

Valor de dado exclusivo	Índice do dicionário	Tamanho (tamanho fixo, 30 bytes por valor)
England	0	30
United States of America	1	30
Venezuela	2	30

Valor de dado exclusivo	Índice do dicionário	Tamanho (tamanho fixo, 30 bytes por valor)
Sri Lanka	3	30
Argentina	4	30
Japan	5	30
Total		180

A tabela a seguir representa os valores na coluna COUNTRY.

Valor original dos dados	Tamanho original (tamanho fixo, 30 bytes por valor)	Valor compactado (índice)	Novo tamanho (bytes)
England	30	0	1
England	30	0	1
United States of America	30	1	1
United States of America	30	1	1
Venezuela	30	2	1
Sri Lanka	30	3	1
Argentina	30	4	1
Japan	30	5	1
Sri Lanka	30	3	1
Argentina	30	4	1
Total	300		10

O tamanho total compactado neste exemplo é calculado da seguinte forma: 6 entradas diferentes são armazenadas no dicionário (6 x 30 = 180) e a tabela contém 10 valores de 1 byte compactados, totalizando 190 bytes.

Delta

As codificações delta são muito úteis para colunas de data e hora.

A codificação delta compacta dados registrando a diferença entre os valores que se seguem na coluna. Essa diferença é registrada em um dicionário separado para cada bloco de valores da coluna em disco. (Um bloco de disco do Amazon Redshift ocupa 1 MB.) Por exemplo, suponha que a coluna contém 10 inteiros em sequência de 1 a 10. Os primeiros são armazenados como um inteiro de 4 bytes (mais um sinalizador de 1 byte). Os próximos nove são armazenados como um byte com o valor 1, indicando que é um maior do que o valor anterior.

A codificação delta vem em duas variações:

- DELTA registra as diferenças como valores de 1 byte (inteiros 8 bits)
- DELTA32K registra as diferenças como valores de 2 bytes (inteiros 16 bits)

Se a maioria dos valores na coluna podem ser compactados usando um único byte, a variação de 1 byte é muito eficaz. No entanto, se os deltas forem maiores, essa codificação, na pior das hipóteses, é um pouco menos eficaz do que o armazenamento de dados descompactados. Uma lógica similar se aplica à versão de 16 bits.

Se a diferença entre dois valores exceder o intervalo de 1 byte (DELTA) ou o intervalo de 2 bytes (DELTA32K), o valor original total é armazenado com um sinalizador de 1 byte. O intervalo de 1 byte é de -127 a 127 e o intervalo de 2 bytes é de -32K a 32K.

A tabela a seguir mostra como uma codificação delta funciona para uma coluna numérica:

Valor original dos dados	Tamanho original (bytes)	Diferença (delta)	Valor compactado	Tamanho compactado (bytes)
1	4		1	1+4 (sinalizador + valor real)
5	4	4	4	1

Valor original dos dados	Tamanho original (bytes)	Diferença (delta)	Valor compactado	Tamanho compactado (bytes)
50	4	45	45	1
200	4	150	150	1+4 (sinalizador + valor real)
185	4	-15	-15	1
220	4	35	35	1
221	4	1	1	1
Totais	28			15

LZO

A codificação LZO fornece uma taxa de compactação bastante elevada com boa performance. A codificação LZO funciona particularmente bem para as colunas CHAR e VARCHAR que armazenam strings de caracteres muito longas. Elas são especialmente boas para texto de formato livre, tais como descrições de produto, comentários de usuários ou strings JSON.

Mostly

Codificações mostly são úteis quando o tipo de dados de uma coluna é maior do que a maioria dos valores armazenados exige. Ao especificar uma codificação mostly para esse tipo de coluna, você pode compactar a maioria dos valores na coluna para um tamanho menor de armazenamento padrão. Os valores restantes que não podem ser compactados são armazenados na forma bruta. Por exemplo, você pode compactar uma coluna de 16 bits, tal como uma coluna INT2, para um armazenamento de 8 bits.

Geralmente, as codificações mostly funcionam com os seguintes tipos de dados:

- SMALLINT/INT2 (16 bits)
- INTEGER/INT (32 bits)
- BIGINT/INT8 (64 bits)
- DECIMAL/NUMERIC (64 bits)

Escolha a variação apropriada da codificação mostly para atender ao tamanho do tipo de dado para a coluna. Por exemplo, aplique MOSTLY8 a uma coluna que seja definida como uma coluna de inteiros de 16 bits. A aplicação de MOSTLY16 a uma coluna com um tipo de dados de 16 bits ou MOSTLY32 a uma coluna com um tipo de dados de 32 bits não é autorizada.

As codificações mostly podem ser menos eficazes do que nenhuma compactação quando um número relativamente alto de valores na coluna não pode ser compactado. Antes de aplicar uma dessas codificações a uma coluna, execute uma verificação. A maioria dos valores que você pretende carregar agora (e prováveis futuros carregamentos) devem caber nos intervalos exibidos na seguinte tabela.

Codificação	Tamanho de armazenamento compactado	Intervalo de valores que podem ser compactados (valores fora do intervalo são armazenados na forma bruta)
MOSTLY8	1 byte (8 bits)	-128 a 127
MOSTLY16	2 bytes (16 bits)	-32768 a 32767
MOSTLY32	4 bytes (32 bits)	-2147483648 a +2147483647

> **Note**
>
> Para valores decimais, ignore o ponto decimal para determinar se o valor se enquadra no intervalo. Por exemplo, 1.234,56 é tratado como 123.456 e pode ser compactado em uma coluna MOSTLY32.

Por exemplo, a coluna VENUEID na tabela VENUE é definida como uma coluna de inteiros brutos, o que significa que seus valores consomem 4 bytes de armazenamento. Contudo, o atual intervalo de valores na coluna é de **`0`** a **`309`**. Portanto, recriar e recarregar essa tabela com a codificação MOSTLY16 para VENUEID reduziria o armazenamento de cada valor nessa coluna para 2 bytes.

Se os valores de VENUEID referenciados em outra tabela estavam sobretudo no intervalo de 0 a 127, pode fazer sentido codificar essa coluna de chave estrangeira como MOSTLY8. Antes de fazer a escolha, execute várias consultas nos dados da tabela de referência para descobrir se os valores caem principalmente no intervalo de 8 bits, 16 bits ou 32 bits.

A tabela a seguir mostra os tamanhos compactados para valores numéricos específicos quando as codificações MOSTLY8, MOSTLY16 e MOSTLY32 são usadas:

Valor original	Tamanho original INT ou BIGINT (bytes)	Tamanho compactado MOSTLY8 (bytes)	Tamanho compactado MOSTLY16 (bytes)	Tamanho compactado MOSTLY32 (bytes)
1	4	1	2	4
10	4	1	2	4
100	4	1	2	4
1000	4	O mesmo tamanho de dados brutos	2	4
10000	4		2	4
20000	4		2	4
40000	8		O mesmo tamanho de dados brutos	4
100000	8			4
2000000000	8			4

Run length

A execução de codificação de comprimento substitui um valor que é repetido consecutivamente por um token que consiste no valor e uma contagem do número de ocorrências consecutivas (a duração da execução). Um dicionário separado de valores exclusivos é criado para cada bloco de valores de coluna em disco. (Um bloco de disco do Amazon Redshift ocupa 1 MB.) Esta codificação é mais apropriada para uma tabela na qual os valores de dados são frequentemente repetidos consecutivamente, por exemplo, quando a tabela é classificada por esses valores.

Por exemplo, suponha que uma coluna em uma tabela de grande dimensão tenha um domínio previsivelmente pequeno, como uma coluna COLOR com menos de 10 valores possíveis. Esses valores provavelmente cairão em longas sequências em toda a tabela, mesmo que os dados não sejam classificados.

Não recomendamos aplicar a codificação de comprimento de execução em qualquer coluna designada como uma chave de classificação. Varreduras de intervalo restrito têm melhor performance quando os blocos contêm números semelhantes de linhas. Se as colunas de chave de classificação forem mais altamente compactadas do que outras colunas na mesma consulta, varreduras restritas por intervalo poderão apresentar má performance.

A tabela a seguir usa o exemplo da coluna COLOR para mostrar como funciona a codificação de comprimento de execução.

Valor original dos dados	Tamanho original (bytes)	Valor compactado (token)	Tamanho compactado (bytes)
Blue	4	{2,Azul}	5
Blue	4		0
Green	5	{3,Verde}	6
Green	5		0
Green	5		0
Blue	4	{1,Blue}	5
Yellow	6	{4,Yellow}	7
Yellow	6		0
Yellow	6		0
Yellow	6		0
Total	51		23

Text255 and Text32k

As codificações text255 e text32k são úteis para a compactação de colunas VARCHAR nas quais as mesmas palavras se repetem com frequência. Um dicionário separado de palavras exclusivas é criado para cada bloco de valores de coluna em disco. (Um bloco de disco do Amazon Redshift ocupa 1 MB.) O dicionário contém as primeiras 245 palavras exclusivas

na coluna. Essas palavras são substituídas em disco por um valor de índice de um byte que representa um dos 245 valores e todas as palavras que não estão representadas no dicionário são armazenadas descompactadas. O processo é repetido para cada bloco de disco de 1 MB. Se as palavras indexadas ocorrerem com frequência na coluna, a coluna produzirá uma alta taxa de compactação.

Para a codificação text32k, o princípio é o mesmo, mas o dicionário para cada bloco não captura um número específico de palavras. Em vez disso, o dicionário indexa cada palavra exclusiva que ele encontra até que as entradas combinadas atinjam o comprimento de 32K, menos alguma sobrecarga. Os valores dos índices são armazenados em dois bytes.

Por exemplo, considere a coluna VENUENAME na tabela VENUE. Palavras como **`Arena`**, **`Center`** e **`Theatre`** são repetidas nesta coluna e provavelmente estão entre as primeiras 245 palavras encontradas em cada bloco se a compactação text255 é aplicada. Em caso afirmativo, esta coluna se beneficia da compactação. Isso ocorre porque sempre que essas palavras aparecem, elas ocupam apenas 1 byte de armazenamento (em vez de 5, 6 ou 7 bytes, respectivamente).

ZSTD

A codificação Zstandard (ZSTD) fornece uma alta taxa de compactação com performance muito boa ao longo de diversos conjuntos de dados. ZSTD funciona particularmente bem com colunas CHAR e VARCHAR que armazenam uma grande variedade de strings longas e curtas, tais como descrições de produto, comentários de usuários, logs e strings JSON. Embora seja possível que alguns algoritmos, como a codificação Delta ou Mostly, usem mais espaço de armazenamento do que a opção sem compactação, é improvável que a codificação ZSTD aumente o uso de disco.

ZSTD é compatível com os tipos de dados SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE PRECISION, BOOLEAN, CHAR, VARCHAR, DATE, TIMESTAMP e TIMESTAMPTZ.

A tabela a seguir identifica as codificações de compactação compatíveis e os tipos de dados compatíveis com a codificação.

Tipo de codificação	Palavra chave em CREATE TABLE e ALTER TABLE	Tipos de dados
Bruto (sem compactação)	RAW	Todos

Tipo de codificação	Palavra chave em CREATE TABLE e ALTER TABLE	Tipos de dados
AZ64	AZ64	SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIMESTAMP, TIMESTAMPTZ
Dicionário de bytes	BYTEDICT	SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE PRECISION, CHAR, VARCHAR, DATE, TIMESTAMP, TIMESTAMP TZ
Delta	DELTA DELTA32K	SMALLINT, INT, BIGINT, DATE, TIMESTAMP, DECIMAL INT, BIGINT, DATE, TIMESTAMP, DECIMAL
LZO	LZO	SMALLINT, INTEGER, BIGINT, DECIMAL, CHAR, VARCHAR, DATE, TIMESTAMP, TIMESTAMP TZ, SUPER
Mostlyn	MOSTLY8 MOSTLY16 MOSTLY32	SMALLINT, INT, BIGINT, DECIMAL INT, BIGINT, DECIMAL BIGINT, DECIMAL
Run-length	RUNLENGTH	SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE PRECISION, BOOLEAN, CHAR, VARCHAR, DATE, TIMESTAMP, TIMESTAMPTZ
Texto	TEXT255 TEXT32K	Somente VARCHAR Somente VARCHAR

Tipo de codificação	Palavra chave em CREATE TABLE e ALTER TABLE	Tipos de dados
Zstandard	ZSTD	SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE PRECISION, BOOLEAN, CHAR, VARCHAR, DATE, TIMESTAMP, TIMESTAMPTZ, SUPER

Teste de codificações de compactação

Se você decidir especificar codificações de coluna manualmente, talvez queira testar diferentes codificações com seus dados.

> **Note**
>
> Recomendamos que você use o comando COPY para carregar dados sempre que possível, permitindo que o comando COPY escolha as codificações ideais com base em seus dados. Como alternativa, você pode usar o comando ANALYZE COMPRESSION para visualizar as codificações sugeridas para os dados existentes. Para obter detalhes sobre a aplicação de compactação automática, consulte Carregamento de tabelas com compactação automática.

Para executar um teste significativo de compactação de dados, é necessário ter um grande número de linhas. Para este exemplo, crie uma tabela e insira linhas usando uma instrução que seleciona dados de duas tabelas: VENUE e LISTING. Omita a cláusula WHERE que normalmente faria a junção das duas tabelas. O resultado é que cada linha da tabela VENUE é unida a todas as linhas da tabela LISTING, para um total de mais de 32 milhões de linhas. Isso é conhecido como uma junção cartesiana e não é normalmente recomendado. No entanto, para esta finalidade, é um método conveniente para criar muitas linhas. Se você tiver uma tabela existente com dados que deseja testar, poderá ignorar esta etapa.

Depois de criar uma tabela com dados de exemplo, crie uma tabela com sete colunas. Cada um tem uma codificação de compactação diferente: raw, bytedict, lzo, run length, text255, text32k e zstd. Preencha cada coluna com exatamente os mesmos dados executando um comando INSERT que seleciona os dados da primeira tabela.

Para testar codificações de compactação, faça o seguinte:

1. (Opcional) Primeiro, use uma junção cartesiana para criar uma tabela com um grande número de linhas. Ignore esta etapa se você deseja testar uma tabela existente.

```
create table cartesian_venue(
venueid smallint not null distkey sortkey,
venuename varchar(100),
venuecity varchar(30),
venuestate char(2),
venueseats integer);

insert into cartesian_venue
select venueid, venuename, venuecity, venuestate, venueseats
from venue, listing;
```

2. Em seguida, crie uma tabela com as codificações que deseja comparar.

```
create table encodingvenue (
venueraw varchar(100) encode raw,
venuebytedict varchar(100) encode bytedict,
venuelzo varchar(100) encode lzo,
venuerunlength varchar(100) encode runlength,
venuetext255 varchar(100) encode text255,
venuetext32k varchar(100) encode text32k,
venuezstd varchar(100) encode zstd);
```

3. Insira os mesmos dados em todas as colunas usando uma instrução INSERT com uma cláusula SELECT.

```
insert into encodingvenue
select venuename as venueraw, venuename as venuebytedict, venuename as venuelzo,
 venuename as venuerunlength, venuename as  venuetext32k, venuename as  venuetext255,
 venuename as venuezstd
from cartesian_venue;
```

4. Verifique o número de linhas na nova tabela.

```
select count(*) from encodingvenue

  count
----------
```

```
 38884394
(1 row)
```

5. Consulte a tabela de sistema STV_BLOCKLIST para comparar o número de blocos de disco de 1 MB usados por cada coluna.

   A função de agregação MAX retorna o número de bloco mais alto para cada coluna. A tabela STV_BLOCKLIST inclui detalhes para as três colunas geradas pelo sistema. Este exemplo usa `col < 6` na cláusula WHERE para excluir as colunas geradas pelo sistema.

```
select col, max(blocknum)
from stv_blocklist b, stv_tbl_perm p
where (b.tbl=p.id) and name ='encodingvenue'
and col < 7
group by name, col
order by col;
```

   A consulta retorna os seguintes resultados. As colunas são numeradas a partir de zero. Dependendo de como seu cluster está configurado, o resultado pode ter números diferentes, mas os tamanhos relativos devem ser similares. Você pode ver que a codificação BYTEDICT na segunda coluna produziu os melhores resultados para este conjunto de dados. Essa abordagem tem uma taxa de compactação de melhor que 20:1. As codificações LZO e ZSTD também produziram excelentes resultados. É evidente que diferentes conjuntos de dados produzem resultados diferentes. Quando uma coluna contém strings de texto mais longas, a codificação LZO frequentemente produz os melhores resultados de compactação.

```
 col | max
-----+-----
   0 | 203
   1 |  10
   2 |  22
   3 | 204
   4 |  56
   5 |  72
   6 |  20
(7 rows)
```

Se você tiver dados em uma tabela existente, será possível usar o comando ANALYZE COMPRESSION para visualizar as codificações sugeridas para a tabela. Por exemplo, o

seguinte exemplo mostra a codificação recomendada para uma cópia da tabela VENUE, CARTESIAN_VENUE, que contém 38 milhões de linhas. Observe que ANALYZE COMPRESSION recomenda a codificação LZO para a coluna VENUENAME. ANALYZE COMPRESSION escolhe a compactação ideal com base em diversos fatores, incluindo a porcentagem de redução. Neste caso específico, BYTEDICT fornece a melhor compactação, mas LZO também produz uma compactação maior que 90 por cento.

```
analyze compression cartesian_venue;

Table          | Column     | Encoding | Est_reduction_pct
---------------+------------+----------+------------------
reallybigvenue | venueid    | lzo      | 97.54
reallybigvenue | venuename  | lzo      | 91.71
reallybigvenue | venuecity  | lzo      | 96.01
reallybigvenue | venuestate | lzo      | 97.68
reallybigvenue | venueseats | lzo      | 98.21
```

Exemplo

O seguinte exemplo cria uma tabela CUSTOMER que tem colunas com vários tipos de dados. Esta instrução CREATE TABLE mostra uma das muitas combinações possíveis de codificações para essas colunas.

```
create table customer(
custkey int encode delta,
custname varchar(30) encode raw,
gender varchar(7) encode text255,
address varchar(200) encode text255,
city varchar(30) encode text255,
state char(2) encode raw,
zipcode char(5) encode bytedict,
start_date date encode delta32k);
```

A tabela a seguir mostra as codificações de coluna que foram escolhidas para a tabela CUSTOMER, fornecendo uma explicação para as escolhas:

Coluna	Tipo de dados	Codificação	Explicação
CUSTKEY	int	delta	CUSTKEY consiste em valores inteiros

Coluna	Tipo de dados	Codificação	Explicação
			consecutivos exclusivos. Como as diferenças são de um byte, DELTA é uma boa escolha.
CUSTNAME	varchar(30)	bruto	CUSTNAME tem um grande domínio com poucos valores repetidos. Qualquer codificação de compactação seria provavelmente ineficaz.
GENDER	varchar(7)	text255	GENDER é um domínio muito pequeno com muitos valores repetidos . Text255 funciona perfeitamente com colunas VARCHAR nas quais as mesmas palavras se repetem.

Coluna	Tipo de dados	Codificação	Explicação
ADDRESS	varchar(200)	text255	ADDRESS é um grande domínio, mas contém muitas palavras repetidas, como rua, avenida, Norte, Sul etc. Text 255 e text 32k são úteis para a compactação de colunas VARCHAR nas quais as mesmas palavras se repetem. O tamanho de coluna é curto, portanto text255 é uma boa escolha.
CITY	varchar(30)	text255	CITY é um grande domínio, com alguns valores repetidos . Determinados nomes de cidade são usados muito mais comumente do que outros. Text255 é uma boa escolha pelos mesmos motivos que ADDRESS.

Coluna	Tipo de dados	Codificação	Explicação
STATE	char(2)	bruto	Nos Estados Unidos, STATE é um domínio preciso de 50 valores de dois caracteres. A codificação Bytedict produziria alguma compactação, mas como o tamanho da coluna é de somente dois caracteres, a compactação pode não compensar o custo da descompac tação dos dados.
ZIPCODE	char(5)	bytedict	ZIPCODE é um domínio conhecido de pouco mais que 50.000 valores exclusivos. Determina dos códigos postais ocorrem muito mais comumente do que outros. A codificação bytedict é muito eficaz quando uma coluna contém um número limitado de valores exclusivos.

Coluna	Tipo de dados	Codificação	Explicação
START_DATE	date	delta32k	Codificações delta são muito úteis para colunas de data e hora, especialmente se as linhas são carregadas em ordem de data.

Distribuição de dados para otimização de consultas

Quando você carrega dados em uma tabela, o Amazon Redshift distribui as linhas da tabela para cada um dos nós de computação de acordo com o estilo de distribuição da tabela. Quando você executa uma consulta, o otimizador de consulta redistribui as linhas aos nós de computação conforme necessário para executar junções e agregações. O objetivo de escolher um estilo de distribuição de tabela é minimizar o impacto da etapa de redistribuição ao localizar os dados onde eles precisam estar antes que a consulta seja executada.

> **Note**
>
> Esta seção apresentará os princípios de distribuição de dados em um banco de dados do Amazon Redshift. Recomendamos que você crie suas tabelas com `DISTSTYLE AUTO`. Se você fizer isso, o Amazon Redshift usará a otimização automática de tabela para escolher o estilo de distribuição de dados. Para obter mais informações, consulte Otimização automática de tabelas. O restante desta seção fornece detalhes sobre estilos de distribuição.

Tópicos

- Conceitos de distribuição de dados
- Estilos de distribuição
- Visualização dos estilos de distribuição
- Avaliação dos padrões de consulta
- Designação de estilos de distribuição
- Avaliação do plano de consulta

- Exemplo de plano de consulta
- Exemplos de distribuição

Conceitos de distribuição de dados

Seguem-se alguns conceitos de distribuição de dados para o Amazon Redshift.

Nós e fatias

Um cluster do Amazon Redshift é um conjunto de nós. Cada nó no cluster tem seu próprio sistema operacional, memória dedicada e armazenamento de disco dedicado. Um dos nós é o nó líder, que gerencia a distribuição de dados e tarefas de processamento de consultas aos nós de computação. Os nós de computação fornecem recursos para realizar essas tarefas.

O armazenamento de disco para um nó de computação é dividido em um número de fatias. O número de fatias por nó depende do tamanho do nó do cluster. Todos os nós participam da execução de consultas paralelas, trabalhando nos dados que são distribuídos da maneira mais uniforme possível pelas fatias. Para obter mais informações sobre o número de fatias que cada tamanho de nó possui, consulte “Clusters e nós no Amazon Redshift” no Guia de gerenciamento de clusters do Amazon Redshift.

Redistribuição de dados

Quando você carrega dados em uma tabela, o Amazon Redshift distribui as linhas da tabela para cada uma das fatias de nó de acordo com o estilo de distribuição da tabela. Como parte de um plano de consulta, o otimizador determina onde os blocos de dados precisam estar localizados para melhor executar a consulta. Então, os dados são movidos fisicamente, ou redistribuídos, enquanto a consulta é executada. A redistribuição pode envolver o envio de linhas específicas aos nós para junção ou a transmissão de uma tabela inteira para todos os nós.

A redistribuição de dados pode representar uma parcela substancial do custo de um plano de consulta e o tráfego de rede que ela gera pode afetar outras operações de banco de dados e retardar a performance geral do sistema. Na medida em que você antecipa onde melhor localizar os dados inicialmente, é possível minimizar o impacto da redistribuição de dados.

Metas da distribuição de dados

Quando você carrega dados em uma tabela, o Amazon Redshift distribui as linhas da tabela para os nós de computação e fatias de acordo com o estilo de distribuição que você escolheu ao criar a tabela. A distribuição de dados tem duas metas principais:

- Para distribuir uniformemente o workload entre os nós no cluster. A distribuição desigual, ou desvio de distribuição de dados, força alguns nós a trabalhar mais do que outros, o que afeta a performance da consulta.
- Para minimizar a movimentação de dados conforme uma consulta é executada. Se as linhas que participam de junções ou agregações já estão colocadas nos nós com suas linhas de junção em outras tabelas, o otimizador não precisa redistribuir tantos dados quando as consultas são executadas.

A estratégia de distribuição escolhida para seu banco de dados tem consequências importantes para a performance da consulta, requisitos de armazenamento, carregamento de dados e manutenção. Com a escolha do melhor estilo de distribuição para cada tabela, você pode equilibrar sua distribuição de dados e melhorar a performance geral do sistema.

Estilos de distribuição

Ao criar uma tabela, é possível determinar um destes quatro estilos de distribuição; AUTO, EVEN, KEY ou ALL.

Se você não especificar um estilo de distribuição, o Amazon Redshift usa distribuição AUTO.

Distribuição AUTO

Com a distribuição AUTO, o Amazon Redshift atribui um estilo de distribuição ideal com base no tamanho dos dados da tabela. Por exemplo, se o estilo de distribuição AUTO for especificado, o Amazon Redshift inicialmente vai atribuir o estilo de distribuição ALL a uma tabela pequena. Quando a tabela crescer, o Amazon Redshift poderá alterar o estilo de distribuição para KEY, escolhendo a chave primária (ou uma coluna da chave primária composta) como chave de distribuição. Se a tabela crescer e nenhuma das colunas for adequada para ser a chave de distribuição, o Amazon Redshift vai alterar o estilo de distribuição para EVEN. A mudança no estilo de distribuição ocorre em segundo plano com impacto mínimo nas consultas do usuário.

Para visualizar ações que o Amazon Redshift executou automaticamente para alterar uma chave de distribuição de tabela, consulte SVL_AUTO_WORKER_ACTION. Para exibir as recomendações atuais sobre a alteração de uma chave de distribuição de tabela, consulte SVV_ALTER_TABLE_RECOMMENDATIONS.

Para visualizar o estilo de distribuição aplicado a uma tabela, consulte a exibição de catálogo de sistema PG_CLASS_INFO. Para obter mais informações, consulte Visualização dos estilos de

distribuição. Se você não especificar um estilo de distribuição com a instrução CREATE TABLE, o Amazon Redshift aplicará a distribuição AUTO.

Distribuição EVEN

O nó de liderança distribui as linhas ao longo das fatias de modo round-robin, independente dos valores de qualquer coluna específica. A distribuição EVEN é apropriada quando uma tabela não participa de junções. Também é apropriado quando não há uma opção clara entre a distribuição KEY e ALL.

Distribuição KEY

As linhas são distribuídas de acordo com os valores em uma coluna. O nó líder coloca os valores correspondentes na mesma fatia do nó. Se você distribuir um par de tabelas nas chaves de união, o nó líder coloca as linhas nas fatias de acordo com os valores nas colunas de união. Desta forma, os valores correspondentes das colunas comuns são armazenados fisicamente juntos.

Distribuição ALL

Uma cópia de toda a tabela é distribuída para cada nó. Onde a distribuição EVEN ou a distribuição KEY coloca apenas uma porção das linhas da tabela em cada nó, a distribuição ALL garante que todas as linhas sejam dispostas para cada junção na qual a tabela participa.

A distribuição ALL multiplica o armazenamento exigido pelo número de nós no cluster e, portanto, ela demora muito mais tempo para carregar, atualizar ou inserir dados em várias tabelas. A distribuição ALL é apropriada somente para tabelas relativamente lentas; ou seja, tabelas que não são atualizadas frequentemente ou extensivamente. Como o custo de redistribuir tabelas pequenas durante uma consulta é baixo, não há um benefício significativo em definir tabelas de pequena dimensão como DISTSTYLE ALL.

> **Note**
>
> Quando você tiver especificado um estilo de distribuição para uma coluna, o Amazon Redshift processa a distribuição em nível do cluster. O Amazon Redshift não exige ou é compatível com o conceito de particionamento de dados em objetos do banco de dados. Você não precisa criar espaços de tabela ou definir esquemas de particionamento para tabelas.

Em alguns cenários, você pode alterar o estilo de distribuição de uma tabela depois de criada. Para obter mais informações, consulte ALTER TABLE. Nos cenários em que não é possível mudar o estilo de distribuição de uma tabela depois da sua criação, você pode criar novamente a tabela e preenchê-la com uma cópia profunda. Para obter mais informações, consulte Execução de uma cópia profunda

Visualização dos estilos de distribuição

Para exibir o estilo de distribuição de uma tabela, consulte a exibição PG_CLASS_INFO ou a exibição SVV_TABLE_INFO.

A coluna RELEFFECTIVEDISTSTYLE em PG_CLASS_INFO indica o estilo de distribuição atual da tabela. Se a tabela usar distribuição automática, RELEFFECTIVEDISTSTYLE será 10, 11 ou 12, o que indica se o estilo de distribuição efetivo é AUTO (ALL) ou AUTO (EVEN) ou AUTO (KEY). Se a tabela usar distribuição automática, o estilo de distribuição poderá mostrar inicialmente AUTO (ALL) e mudar para AUTO (EVEN) ou AUTO (KEY) quando a tabela crescer.

A tabela a seguir fornece o estilo de distribuição para cada valor na coluna RELEFFECTIVEDISTSTYLE:

RELEFFECTIVEDISTSTYLE	Estilo de distribuição atual
0	EVEN
1	KEY
8	ALL
10	AUTO (ALL)
11	AUTO (EVEN)
12	AUTO (KEY)

A coluna DISTSTYLE em SVV_TABLE_INFO indica o estilo de distribuição atual da tabela. Se a tabela usar distribuição automática, DISTSTYLE será AUTO (ALL) ou AUTO (EVEN) ou AUTO (KEY).

O exemplo a seguir cria quatro tabelas usando os três estilos de distribuição e distribuição automática e, em seguida, consulta SVV_TABLE_INFO para visualizar os estilos de distribuição.

```
create table public.dist_key (col1 int)
diststyle key distkey (col1);

insert into public.dist_key values (1);

create table public.dist_even (col1 int)
diststyle even;

insert into public.dist_even values (1);

create table public.dist_all (col1 int)
diststyle all;

insert into public.dist_all values (1);

create table public.dist_auto (col1 int);

insert into public.dist_auto values (1);

select "schema", "table", diststyle from SVV_TABLE_INFO
where "table" like 'dist%';

        schema   |    table        | diststyle
     ------------+-----------------+------------
      public     | dist_key        | KEY(col1)
      public     | dist_even       | EVEN
      public     | dist_all        | ALL
      public     | dist_auto       | AUTO(ALL)
```

Avaliação dos padrões de consulta

A escolha de estilos de distribuição é apenas um aspecto do design do banco de dados. Considere os estilos de distribuição somente no contexto do sistema completo, equilibrando a distribuição com outros fatores importantes, como o tamanho do cluster, os métodos de codificação de compactação, as chaves de classificação e as restrições da tabela.

Teste seu sistema com dados que sejam o mais próximo possível de dados reais.

Para fazer boas escolhas para estilos de distribuição, você precisa entender os padrões de consulta para sua aplicação do Amazon Redshift. Identifique as consultas mais caras em seu sistema e baseie seu projeto inicial de banco de dados nas demandas dessas consultas. Os fatores que determinam o custo total de uma consulta incluem o tempo necessário para execução da consulta e a quantidade de recursos de computação consumida. Outros fatores que determinam o custo da consulta são a frequência de execução da consulta e o quanto ela interrompe outras consultas e operações do banco de dados.

Identifique as tabelas que são usadas pelas consultas mais caras e avalie seu perfil no runtime da consulta. Considere como as tabelas são associadas e agregadas.

Use as diretrizes nesta seção para escolher um estilo de distribuição para cada tabela. Depois de fazer isso, crie as tabelas e carregue-as com dados que sejam o mais próximo possível dos dados reais. Em seguida, teste as tabelas para os tipos de consultas que você espera usar. Você pode avaliar os planos de explicação da consulta para identificar oportunidades de ajuste. Compare os tempos de carregamento, o espaço de armazenamento e os runtimes da consulta para equilibrar os requisitos globais do seu sistema.

Designação de estilos de distribuição

As considerações e recomendações para designação de estilos de distribuição neste seção usam um esquema estrela como exemplo. O design do seu banco de dados pode ser baseado em um esquema em estrela, alguma variante de um esquema em estrela ou um esquema totalmente diferente. O Amazon Redshift foi projetado para funcionar de maneira eficaz com qualquer projeto de esquema que você escolher. Os princípios nesta seção podem ser aplicados a qualquer esquema de design.

1. Especifique a chave primária e chaves estrangeiras para todas as suas tabelas.

   O Amazon Redshift não impõe restrições de chave primária e chave estrangeira, mas o otimizador de consulta as usa ao gerar planos de consulta. Se você definir chaves primárias e chaves estrangeiras, seu aplicativo deverá manter a validade das chaves.
2. Distribua a tabela de fatos e sua maior tabela de dimensões em suas colunas comuns.

   Escolha a maior dimensão com base no tamanho do conjunto de dados que participa da junção mais comum, não apenas no tamanho da tabela. Se uma tabela é geralmente filtrada usando uma cláusula WHERE, somente uma porção de suas linhas participam da junção. Tal tabela tem menor impacto na redistribuição do que uma tabela menor que contribua mais dados. Designe a chave primária da tabela de dimensões e a chave estrangeira correspondente da tabela de fatos

como DISTKEY. Se várias tabelas usarem a mesma chave de distribuição, elas também serão colocadas com a tabela de fatos. Sua tabela de fatos pode ter apenas uma chave de distribuição. Quaisquer tabelas que se juntam em outra chave não são colocadas com a tabela de fatos.

3. Designe as chaves de distribuição para as outras tabelas de dimensões.

   Distribua as tabelas em suas chaves primárias ou em suas chaves estrangeiras, dependendo de como elas geralmente se juntam com outras tabelas.

4. Avalie a necessidade de alterar algumas das tabelas de dimensões para usar a distribuição ALL.

   Se uma tabela de dimensão não pode ser disposta com a tabela de fato ou outras tabelas de junção importantes, você pode melhorar a performance da consulta significativamente ao distribuir toda a tabela para todos os nós. O uso de distribuição ALL multiplica as exigências de espaço de armazenamento e aumenta os tempos de carregamento e operações de manutenção, portanto você deve considerar todos os fatores antes de optar pela distribuição ALL. A seção a seguir explica como identificar candidatos para a distribuição ALL ao avaliar o plano EXPLAIN.

5. Use a distribuição AUTO para as tabelas restantes.

   Se uma tabela for amplamente desnormalizada e não participar de junções, ou se você não tiver uma opção clara de outro estilo de distribuição, use a distribuição AUTO.

Para permitir que o Amazon Redshift escolha o estilo de distribuição apropriado, não especifique explicitamente um estilo de distribuição.

Avaliação do plano de consulta

Você pode usar planos de consulta para identificar candidatos para otimização do estilo de distribuição.

Após tomar as decisões iniciais de design, crie suas tabelas, preencha-as com dados e teste-as. Use um conjunto de dados de teste que seja o mais próximo possível dos dados reais. Meça o tempo de carregamento para usar como linha de base para comparações.

Avalie as consultas que representam as consultas mais caras que você espera executar, especificamente as consultas que usam junções e agregações. Compare os runtimes para várias opções de design. Ao comparar runtimes, não conte a primeira vez que a consulta é executada, porque o primeiro runtime inclui o tempo de compilação.

DS_DIST_NONE

Nenhuma redistribuição é necessária, pois as fatias correspondentes são arranjadas nos nós de computação. Normalmente, você tem apenas uma etapa DS_DIST_NONE, a junção entre a tabela de fatos e uma tabela de dimensão.

DS_DIST_ALL_NONE

Nenhuma redistribuição é necessária, pois a tabela de junção interna usou DISTSTYLE ALL. A tabela completa está localizada em todos os nós.

DS_DIST_INNER

A tabela interna é redistribuída.

DS_DIST_OUTER

A tabela externa é redistribuída.

DS_BCAST_INNER

Uma cópia de toda a tabela interna é transmitida a todos os nós de computação.

DS_DIST_ALL_INNER

Toda a tabela interna é redistribuída a uma única fatia, pois a tabela externa usa DISTSTYLE ALL.

DS_DIST_BOTH

Ambas as tabelas são redistribuídas.

DS_DIST_ERR

Quando a tabela não tem um estilo de distribuição selecionado.

DS_DIST_NONE e DS_DIST_ALL_NONE são bons. Eles indicam que nenhuma distribuição foi necessária para esta etapa, pois todas as junções estão dispostas.

DS_DIST_INNER significa que a etapa provavelmente tem um custo relativamente alto porque a tabela interna está sendo redistribuída para os nós. DS_DIST_INNER indica que a tabela externa já está adequadamente distribuída na chave de junção. Defina a chave de distribuição da tabela interna como a chave de junção para converter isso para DS_DIST_NONE. Em alguns casos, a distribuição da tabela interna na chave de junção não é possível porque a tabela externa não está distribuída na chave de junção. Se esse for o caso, avalie se deseja usar a distribuição ALL para a tabela interna.

Se a tabela não for atualizada com frequência ou extensivamente e for grande o suficiente para suportar um alto custo de redistribuição, altere o estilo de distribuição para ALL e teste novamente. A distribuição ALL resulta em maior tempos de carregamento, portanto ao testar novamente, inclua o tempo de carregamento em seus fatores de avaliação.

DS_DIST_ALL_INNER não é bom. Isso significa que toda a tabela interna é redistribuída em uma única fatia, pois a tabela externa usa DISTSTYLE ALL, de forma que uma cópia de toda a tabela externa está localizada em cada nó. Isso resulta em runtime em série ineficaz da junção em um único nó, em vez de tirar proveito do runtime paralelo usando todos os nós. DISTSTYLE ALL é destinado ao uso somente para a tabela de junção interna. Em vez disso, especifique uma chave de distribuição ou use a distribuição uniforme para a tabela externa.

DS_BCAST_INNER e DS_DIST_BOTH não são bons. Geralmente, essas redistribuições ocorrem pois as tabelas não são associadas em suas chaves de distribuição. Se a tabela de fatos ainda não tiver uma chave de distribuição, especifique a coluna de junção como a chave de distribuição para ambas as tabelas. Se a tabela de fatos já tiver uma chave de distribuição em outra coluna, avalie se alterar a chave de distribuição para colocar essa junção melhora a performance geral. Se alterar a chave de distribuição da tabela externa não for uma escolha ideal, você pode obter a colocação especificando DISTSTYLE ALL para a tabela interna.

O seguinte exemplo mostra uma porção de um plano de consulta com os rótulos DS_BCAST_INNER e DS_DIST_NONE.

```
->  XN Hash Join DS_BCAST_INNER  (cost=112.50..3272334142.59 rows=170771 width=84)
        Hash Cond: ("outer".venueid = "inner".venueid)
        ->  XN Hash Join DS_BCAST_INNER  (cost=109.98..3167290276.71 rows=172456
 width=47)
              Hash Cond: ("outer".eventid = "inner".eventid)
              ->  XN Merge Join DS_DIST_NONE  (cost=0.00..6286.47 rows=172456 width=30)
                    Merge Cond: ("outer".listid = "inner".listid)
                    ->  XN Seq Scan on listing  (cost=0.00..1924.97 rows=192497
 width=14)
                    ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=24)
```

Após a alteração das tabelas de dimensões para uso de DISTSTYLE ALL, o plano de consulta para a mesma consulta exibe DS_DIST_ALL_NONE em vez de DS_BCAST_INNER. Além disso, há uma alteração drástica no custo relativo para as etapas da junção. O custo total é `14142.59` comparado ao `3272334142.59` da consulta anterior.

```
->  XN Hash Join DS_DIST_ALL_NONE  (cost=112.50..14142.59 rows=170771 width=84)
```

```
        Hash Cond: ("outer".venueid = "inner".venueid)
        ->  XN Hash Join DS_DIST_ALL_NONE  (cost=109.98..10276.71 rows=172456 width=47)
              Hash Cond: ("outer".eventid = "inner".eventid)
              ->  XN Merge Join DS_DIST_NONE  (cost=0.00..6286.47 rows=172456 width=30)
                    Merge Cond: ("outer".listid = "inner".listid)
                    ->  XN Seq Scan on listing  (cost=0.00..1924.97 rows=192497
 width=14)
                    ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=24)
```

Exemplo de plano de consulta

Este exemplo mostra como avaliar um plano de consulta para encontrar oportunidades de otimização da distribuição.

Execute a seguinte consulta com um comando EXPLAIN para produzir um plano de consulta.

```
explain
select lastname, catname, venuename, venuecity, venuestate, eventname,
month, sum(pricepaid) as buyercost, max(totalprice) as maxtotalprice
from category join event on category.catid = event.catid
join venue on venue.venueid = event.venueid
join sales on sales.eventid = event.eventid
join listing on sales.listid = listing.listid
join date on sales.dateid = date.dateid
join users on users.userid = sales.buyerid
group by lastname, catname, venuename, venuecity, venuestate, eventname, month
having sum(pricepaid)>9999
order by catname, buyercost desc;
```

No banco de dados TICKIT, SALES é uma tabela de fatos e LISTING é a maior dimensão. Para posicionar as tabelas, SALES é distribuída em LISTID, que é a chave estrangeira para LISTING, e LISTING é distribuída em sua chave primária, LISTID. O exemplo a seguir mostra os comandos CREATE TABLE para SALES e LISTING.

```
create table sales(
 salesid integer not null,
 listid integer not null distkey,
 sellerid integer not null,
 buyerid integer not null,
 eventid integer not null encode mostly16,
 dateid smallint not null,
```

```
 qtysold smallint not null encode mostly8,
 pricepaid decimal(8,2) encode delta32k,
 commission decimal(8,2) encode delta32k,
 saletime timestamp,
 primary key(salesid),
 foreign key(listid) references listing(listid),
 foreign key(sellerid) references users(userid),
 foreign key(buyerid) references users(userid),
 foreign key(dateid) references date(dateid))
       sortkey(listid,sellerid);

create table listing(
 listid integer not null distkey sortkey,
 sellerid integer not null,
 eventid integer not null encode mostly16,
 dateid smallint not null,
 numtickets smallint not null encode mostly8,
 priceperticket decimal(8,2) encode bytedict,
 totalprice decimal(8,2) encode mostly32,
 listtime timestamp,
 primary key(listid),
 foreign key(sellerid) references users(userid),
 foreign key(eventid) references event(eventid),
 foreign key(dateid) references date(dateid));
```

No seguinte plano de consulta, a etapa merge join para a junção em SALES e LISTING mostra DS_DIST_NONE, que indica que nenhuma redistribuição é necessária para a etapa. No entanto, mais acima no plano de consulta, outras junções internas exibem DS_BCAST_INNER, que indica que a tabela interna é transmitida como parte da execução da consulta. Como somente um par de tabelas pode ser posicionado usando a distribuição de chaves, cinco tabelas precisam ser retransmitidas.

```
QUERY PLAN
XN Merge  (cost=1015345167117.54..1015345167544.46 rows=1000 width=103)
  Merge Key: category.catname, sum(sales.pricepaid)
  ->  XN Network  (cost=1015345167117.54..1015345167544.46 rows=170771 width=103)
        Send to leader
        ->  XN Sort  (cost=1015345167117.54..1015345167544.46 rows=170771 width=103)
              Sort Key: category.catname, sum(sales.pricepaid)
              ->  XN HashAggregate  (cost=15345150568.37..15345152276.08 rows=170771
 width=103)
                    Filter: (sum(pricepaid) > 9999.00)
```

```
                     ->  XN Hash Join DS_BCAST_INNER  (cost=742.08..15345146299.10
rows=170771 width=103)
                           Hash Cond: ("outer".catid = "inner".catid)
                           ->  XN Hash Join DS_BCAST_INNER
(cost=741.94..15342942456.61 rows=170771 width=97)
                                 Hash Cond: ("outer".dateid = "inner".dateid)
                                 ->  XN Hash Join DS_BCAST_INNER
(cost=737.38..15269938609.81 rows=170766 width=90)
                                       Hash Cond: ("outer".buyerid = "inner".userid)
                                       ->  XN Hash Join DS_BCAST_INNER
(cost=112.50..3272334142.59 rows=170771 width=84)
                                             Hash Cond: ("outer".venueid =
"inner".venueid)
                                             ->  XN Hash Join DS_BCAST_INNER
(cost=109.98..3167290276.71 rows=172456 width=47)
                                                   Hash Cond: ("outer".eventid =
"inner".eventid)
                                                   ->  XN Merge Join DS_DIST_NONE
(cost=0.00..6286.47 rows=172456 width=30)
                                                         Merge Cond: ("outer".listid =
"inner".listid)
                                                         ->  XN Seq Scan on listing
(cost=0.00..1924.97 rows=192497 width=14)
                                                         ->  XN Seq Scan on sales
(cost=0.00..1724.56 rows=172456 width=24)
                                                   ->  XN Hash  (cost=87.98..87.98
rows=8798 width=25)
                                                         ->  XN Seq Scan on event
(cost=0.00..87.98 rows=8798 width=25)
                                             ->  XN Hash  (cost=2.02..2.02 rows=202
width=41)
                                                   ->  XN Seq Scan on venue
(cost=0.00..2.02 rows=202 width=41)
                                       ->  XN Hash  (cost=499.90..499.90 rows=49990
width=14)
                                             ->  XN Seq Scan on users
(cost=0.00..499.90 rows=49990 width=14)
                                 ->  XN Hash  (cost=3.65..3.65 rows=365 width=11)
                                       ->  XN Seq Scan on date  (cost=0.00..3.65
rows=365 width=11)
                           ->  XN Hash  (cost=0.11..0.11 rows=11 width=10)
                                 ->  XN Seq Scan on category  (cost=0.00..0.11 rows=11
width=10)
```

Uma solução é alterar as tabelas para que tenham DISTSTYLE ALL.

```
ALTER TABLE users ALTER DISTSTYLE ALL;
ALTER TABLE venue ALTER DISTSTYLE ALL;
ALTER TABLE category ALTER DISTSTYLE ALL;
ALTER TABLE date ALTER DISTSTYLE ALL;
ALTER TABLE event ALTER DISTSTYLE ALL;
```

Execute a mesma consulta com EXPLAIN novamente e examine o novo plano de consulta. As junções agora exibem DS_DIST_ALL_NONE, indicando que nenhuma redistribuição é necessária, pois os dados foram distribuídos para todos os nós usando DISTSTYLE ALL.

```
QUERY PLAN
XN Merge  (cost=1000000047117.54..1000000047544.46 rows=1000 width=103)
  Merge Key: category.catname, sum(sales.pricepaid)
  ->  XN Network  (cost=1000000047117.54..1000000047544.46 rows=170771 width=103)
        Send to leader
        ->  XN Sort  (cost=1000000047117.54..1000000047544.46 rows=170771 width=103)
              Sort Key: category.catname, sum(sales.pricepaid)
              ->  XN HashAggregate  (cost=30568.37..32276.08 rows=170771 width=103)
                    Filter: (sum(pricepaid) > 9999.00)
                    ->  XN Hash Join DS_DIST_ALL_NONE  (cost=742.08..26299.10
 rows=170771 width=103)
                          Hash Cond: ("outer".buyerid = "inner".userid)
                          ->  XN Hash Join DS_DIST_ALL_NONE  (cost=117.20..21831.99
 rows=170766 width=97)
                                Hash Cond: ("outer".dateid = "inner".dateid)
                                ->  XN Hash Join DS_DIST_ALL_NONE
 (cost=112.64..17985.08 rows=170771 width=90)
                                      Hash Cond: ("outer".catid = "inner".catid)
                                      ->  XN Hash Join DS_DIST_ALL_NONE
 (cost=112.50..14142.59 rows=170771 width=84)
                                            Hash Cond: ("outer".venueid =
 "inner".venueid)
                                            ->  XN Hash Join DS_DIST_ALL_NONE
 (cost=109.98..10276.71 rows=172456 width=47)
                                                  Hash Cond: ("outer".eventid =
 "inner".eventid)
                                                  ->  XN Merge Join DS_DIST_NONE
 (cost=0.00..6286.47 rows=172456 width=30)
                                                        Merge Cond: ("outer".listid =
 "inner".listid)
```

```
                                                        ->  XN Seq Scan on listing
 (cost=0.00..1924.97 rows=192497 width=14)
                                                        ->  XN Seq Scan on sales
 (cost=0.00..1724.56 rows=172456 width=24)
                                                  ->  XN Hash  (cost=87.98..87.98
 rows=8798 width=25)
                                                        ->  XN Seq Scan on event
 (cost=0.00..87.98 rows=8798 width=25)
                                              ->  XN Hash  (cost=2.02..2.02 rows=202
 width=41)
                                                    ->  XN Seq Scan on venue
 (cost=0.00..2.02 rows=202 width=41)
                                        ->  XN Hash  (cost=0.11..0.11 rows=11 width=10)
                                              ->  XN Seq Scan on category
 (cost=0.00..0.11 rows=11 width=10)
                                  ->  XN Hash  (cost=3.65..3.65 rows=365 width=11)
                                        ->  XN Seq Scan on date  (cost=0.00..3.65
 rows=365 width=11)
                            ->  XN Hash  (cost=499.90..499.90 rows=49990 width=14)
                                  ->  XN Seq Scan on users  (cost=0.00..499.90 rows=49990
 width=14)
```

Exemplos de distribuição

Os seguintes exemplos mostram como os dados são distribuídos de acordo com as opções que você define na instrução CREATE TABLE.

Exemplos de DISTKEY

Observe o esquema da tabela USERS no banco de dados TICKIT. USERID é definida como a coluna SORTKEY e a coluna DISTKEY:

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'users';

    column     |          type          | encoding | distkey | sortkey
---------------+------------------------+----------+---------+---------
 userid        | integer                | none     | t       |       1
 username      | character(8)           | none     | f       |       0
 firstname     | character varying(30)  | text32k  | f       |       0

...
```

USERID é uma boa escolha para a coluna de distribuição dessa tabela. Se você consultar a exibição de sistema SVV_DISKUSAGE, verá que a tabela é distribuída de forma bastante uniforme. Os números de coluna são baseados em zero, portanto USERID é a coluna 0.

```
select slice, col, num_values as rows, minvalue, maxvalue
from svv_diskusage
where name='users' and col=0 and rows>0
order by slice, col;

slice| col | rows  | minvalue | maxvalue
-----+-----+-------+----------+----------
0    | 0   | 12496 | 4        | 49987
1    | 0   | 12498 | 1        | 49988
2    | 0   | 12497 | 2        | 49989
3    | 0   | 12499 | 3        | 49990
(4 rows)
```

A tabela contém 49.990 linhas. As coluna de linhas (num_values) mostra que cada fatia contém aproximadamente o mesmo número de linhas. As colunas minvalue e maxvalue exibem o intervalo de valores em cada fatia. Cada fatia inclui quase todo o intervalo de valores, portanto, há uma boa chance de que cada fatia participe da execução de uma consulta que filtra uma variedade de IDs de usuário.

Este exemplo demonstra a distribuição em um pequeno sistema de teste. O número total de fatias é geralmente muito maior.

Se você normalmente realiza junções ou agrupamentos usando a coluna STATE, pode optar por distribuir na coluna STATE. O exemplo a seguir mostra um caso em que você cria uma nova tabela com os mesmos dados da tabela USERS, mas define DISTKEY para a coluna STATE. Neste caso, a distribuição não é igual. A fatia 0 (13.587 linhas) contém aproximadamente 30 por cento mais linhas do que a fatia 3 (10.150 linhas). Em uma tabela muito maior, essa quantidade de distorção de distribuição pode ter um impacto adverso no processamento de consultas.

```
create table userskey distkey(state) as select * from users;

select slice, col, num_values as rows, minvalue, maxvalue from svv_diskusage
where name = 'userskey' and col=0 and rows>0
order by slice, col;

slice | col | rows  | minvalue | maxvalue
------+-----+-------+----------+----------
```

```
    0 |   0 | 13587 |        5 |    49989
    1 |   0 | 11245 |        2 |    49990
    2 |   0 | 15008 |        1 |    49976
    3 |   0 | 10150 |        4 |    49986
(4 rows)
```

Exemplo de DISTSTYLE EVEN

Se você criar uma nova tabela com os mesmos dados da tabela USERS, mas definir DISTSTYLE como EVEN, as linhas sempre serão distribuídas uniformemente ao longo das fatias.

```
create table userseven diststyle even as
select * from users;

select slice, col, num_values as rows, minvalue, maxvalue from svv_diskusage
where name = 'userseven' and col=0 and rows>0
order by slice, col;

slice | col | rows  | minvalue | maxvalue
------+-----+-------+----------+----------
    0 |   0 | 12497 |        4 |    49990
    1 |   0 | 12498 |        8 |    49984
    2 |   0 | 12498 |        2 |    49988
    3 |   0 | 12497 |        1 |    49989
(4 rows)
```

Entretanto, como a distribuição não é baseada em uma coluna específica, o processamento da consulta pode ser degradado, especialmente se a tabela estiver associada à outras tabelas. A falta de distribuição em uma coluna de junção frequentemente influencia o tipo de operação de junção que pode ser executada eficientemente. Operações de junção, agregação e agrupamento são otimizadas quando ambas as tabelas são distribuídas e classificadas em suas respectivas colunas de junção.

Exemplo de DISTSTYLE ALL

Se você criar uma nova tabela com os mesmos dados da tabela USERS, mas definir DISTSTYLE como ALL, todas as linhas serão distribuídas para a primeira fatia de cada nó.

```
select slice, col, num_values as rows, minvalue, maxvalue from svv_diskusage
where name = 'usersall' and col=0 and rows > 0
order by slice, col;
```

```
slice | col | rows  | minvalue | maxvalue
------+-----+-------+----------+----------
    0 |   0 | 49990 |        4 |    49990
    2 |   0 | 49990 |        2 |    49990

(4 rows)
```

Chaves de classificação

> **Note**
>
> Recomendamos que você crie suas tabelas com `SORTKEY AUTO`. Se você fizer isso, o Amazon Redshift usará a otimização automática de tabela para escolher a chave de classificação. Para obter mais informações, consulte Otimização automática de tabelas. O restante desta seção fornece detalhes sobre a ordem de classificação.

Ao criar uma tabela, você pode definir alternativamente uma ou mais de suas colunas como chaves de classificação. Quando os dados são carregados na tabela vazia, as linhas são armazenadas em disco na ordem de classificação. As informações sobre as colunas de chave de classificação são transmitidas ao planejador de consulta, que usa essas informações para construir planos que exploram a forma como os dados são classificados. Para obter mais informações, consulte CRIAR TABELA. Para obter informações sobre práticas recomendadas ao criar uma chave de classificação, consulte Escolha a melhor chave de classificação.

A classificação permite o manuseio eficiente de predicados de alcance restrito. O Amazon Redshift armazena dados colunares em blocos de disco de 1 MB. Os valores mínimos e máximos para cada bloco são armazenados como parte dos metadados. Se a consulta usar um predicado restrito por intervalo, o processador de consulta poderá usar os valores mínimos e máximos para ignorar rapidamente um grande número de blocos durante varreduras da tabela. Por exemplo, suponha que uma tabela armazena cinco anos de dados classificados por data e uma consulta especifica um intervalo de um mês. Nesse caso, você pode remover até 98% dos blocos de disco da verificação. Se os dados não estiverem classificados, a varredura deverá incluir um número maior de blocos de disco (possivelmente todos).

Você pode especificar uma chave de classificação composta ou intercalada. Uma chave de classificação composta é mais eficiente quando os predicados de consulta usam um prefixo, que

é um subconjunto das colunas de chave de classificação ordenadas. Uma chave de classificação intercalada concede igual peso a todas as colunas na chave de classificação, portanto os predicados de consulta podem usar qualquer subconjunto de colunas que componha a chave de classificação, em qualquer ordem.

Para compreender o impacto da chave de classificação escolhida na performance da consulta, use o comando EXPLAIN. Para obter mais informações, consulte Planejamento de consulta e fluxo de trabalho de execução.

Para definir um tipo de classificação, use a palavra-chave INTERLEAVED ou COMPOUND com sua instrução CREATE TABELA ou CREATE TABLE AS. O padrão é COMPOUND. COMPOUND é recomendado quando você atualiza tabelas regularmente com as operações INSERT, UPDATE ou DELETE. Uma chave de classificação INTERLEAVED pode usar um máximo de oito colunas. Dependendo dos dados e do tamanho do cluster, VACUUM REINDEX leva muito mais tempo do que VACUUM FULL porque faz uma passagem adicional para analisar as chaves de classificação intercaladas. As operações de classificação e mesclagem podem levar mais tempo para tabelas intercaladas porque a classificação intercalada pode precisar reorganizar mais linhas do que uma classificação composta.

Para visualizar as chaves de classificação para uma tabela, consulte a exibição de sistema SVV_TABLE_INFO.

Tópicos

- Classificação do layout de dados multidimensionais
- Chave de classificação composta
- Chave de classificação intercalada

Classificação do layout de dados multidimensionais

Uma chave de classificação do layout de dados multidimensional é um tipo de chave de classificação AUTO que se baseia em predicados repetitivos encontrados em um workload. Se o workload tiver predicados repetitivos, o Amazon Redshift poderá melhorar o desempenho da verificação de tabelas colocando linhas de dados que atendam aos predicados repetitivos. Em vez de armazenar dados de uma tabela em ordem de colunas estrita, uma chave de classificação do layout de dados multidimensional armazena dados analisando predicados repetitivos exibidos em um workload. Mais de um predicado repetitivo pode ser encontrado em um workload. Dependendo do workload, esse tipo de chave de classificação pode melhorar o desempenho de muitos predicados. O Amazon

Redshift determina automaticamente se esse método de chave de classificação deve ser usado em tabelas definidas com uma chave de classificação AUTO.

Por exemplo, suponhamos que você tenha uma tabela com dados classificado na ordem de colunas. Muitos blocos de dados talvez precisem ser examinados para determinar se atendem aos predicados no workload. Porém, se os dados forem armazenados no disco em uma ordem de predicados, menos blocos precisarão ser examinados para atender à consulta. O uso de uma chave de classificação de layout de dados multidimensional é benéfico nesse caso.

Para exibir se uma consulta está usando uma chave de layout de dados multidimensional, consulte a coluna `step_attribute` da exibição SYS_QUERY_DETAIL. Quando o valor é `multi-dimensional`, o layout de dados multidimensional foi usado na consulta.

Para evitar que o Amazon Redshift use uma chave de classificação de layout de dados multidimensional, escolha uma opção da chave de classificação de tabela diferente de SORTKEY AUTO. Para obter mais informações sobre opções SORTKEY, consulte CRIAR TABELA.

Chave de classificação composta

Uma chave composta contém todas as colunas listadas na definição de chave de classificação, na ordem em que estão listadas. Uma chave de classificação composta é mais útil quando o filtro de uma consulta aplica condições, tais como filtros e junções, que usam um prefixo das chaves de classificação. Os benefícios de performance da classificação compostos diminuem quando as consultas dependerem somente de colunas de classificação secundárias, sem consultar as colunas principais. COMPOUND é o tipo de classificação padrão.

As chaves de classificação compostas podem acelerar junções, operações GROUP BY e ORDER BY e funções de janela que usam PARTITION BY e ORDER BY. Por exemplo, uma junção de mesclagem, que frequentemente é mais rápida do junções de hash, é possível quando os dados são distribuídos e pré-classificados nas colunas de junção. As chaves de classificação compostas também ajudam a melhorar a compactação.

À medida que você adiciona linhas em uma tabela classificada que já contém dados, a região não classificada aumenta, o que tem um efeito significativo na performance. O efeito é maior quando a tabela usa classificação intercalada, especialmente quando as colunas de classificação contêm dados que crescem monotonicamente, tais como colunas de data ou timestamp. Execute uma operação VACUUM regularmente, especialmente após grandes carregamentos de dados, para reclassificar e reanalisar os dados. Para obter mais informações, consulte Reduzir o tamanho da região não classificada. Após a limpeza para reclassificação dos dados, é uma boa prática executar

um comando ANALYZE para atualizar os metadados estatísticos para o planejador de consulta. Para obter mais informações, consulte Análise de tabelas.

Chave de classificação intercalada

Uma classificação intercalada concede igual peso a cada coluna ou subconjunto de colunas na chave de classificação. Se várias consultas usam colunas diferentes como filtros, é frequentemente possível melhorar a performance dessas consultas usando um estilo intercalado de classificação. Quando uma consulta usa predicados restritivos em colunas de classificação secundárias, a classificação intercalada melhora significativamente a performance da consulta em relação à classificação composta.

> **Important**
>
> Não use uma chave de classificação intercalada em colunas com atributos que aumentam monotonicamente, como colunas de identidade, datas ou time stamps.

As melhorias de performance que você ganha ao executar uma chave de classificação intercalada devem ser pesadas contra o aumento nos tempos de carregamento e limpeza.

Classificações intercaladas são mais eficazes com consultas altamente seletivas que filtram uma ou mais das colunas de chave de classificação na cláusula WHERE, por exemplo `select c_name from customer where c_region = 'ASIA'`. Os benefícios da classificação intercalada aumentam com o número de colunas classificadas que são restritas.

Uma classificação intercalada é mais eficaz com tabelas grandes. A classificação é aplicada em cada fatia. Assim, uma classificação intercalada é mais eficaz quando uma tabela é grande o suficiente para exigir vários blocos de 1 MB por fatia. Aqui, o processador de consultas pode ignorar uma proporção significativa dos blocos usando predicados restritivos. Para visualizar o número de blocos que uma tabela usa, consulte a exibição de sistema STV_BLOCKLIST.

Para classificação em uma única coluna, uma classificação intercalada pode apresentar melhor performance do que uma classificação composta se os valores da coluna tiverem um prefixo comum longo. Por exemplo, URLs normalmente começam com “http://www”. As chaves de classificação compostas usam um número limitado de caracteres do prefixo, o que resulta na duplicação de muitas chaves. As classificações intercaladas usam um esquema de compactação interna para valores de mapas de zona que lhes permite melhor distinção entre valores de coluna que possuem um prefixo comum longo.

Ao migrar clusters provisionados do Amazon Redshift para o Amazon Redshift Serverless, o Redshift converte tabelas com chaves de classificação intercaladas e DISTSTYLE KEY em chaves de classificação compostas. No entanto, tabelas com apenas chaves de classificação intercaladas permanecem inalteradas. Para obter mais informações sobre estilos de distribuição, consulte Trabalhar com estilos de distribuição de dados.

VACUUM REINDEX

À medida que você adiciona linhas a uma tabela classificada que já contém dados, a performance pode deteriorar-se com o tempo. Essa deterioração ocorre para classificações compostas e classificações intercaladas, mas tem um efeito maior em tabelas intercaladas. Um VACUUM restaura a ordem de classificação, mas a operação pode levar mais tempo para tabelas intercaladas, pois a mesclagem de novos dados intercalados pode envolver a modificação de cada bloco de dados.

Quando as tabelas são carregadas inicialmente, o Amazon Redshift analisa a distribuição dos valores nas colunas-chave de classificação e usa essas informações para intercalação ideal das colunas-chave de classificação. À medida que a tabela cresce, a distribuição dos valores nas colunas de chave de classificação pode mudar, ou desviar, especialmente com colunas de data ou timestamp. Se o desvio se tornar muito grande, a performance poderá ser afetada. Para reanalisar as chaves de classificação e restaurar a performance, execute o comando VACUUM com a palavra chave REINDEX. Como ele precisa realizar uma análise adicional dos dados, o comando VACUUM REINDEX pode demorar mais do que um comando VACUUM padrão para tabelas intercaladas. Para visualizar informações sobre desvio de distribuição de chaves e hora da última reindexação, consulte a exibição de sistema SVV_INTERLEAVED_COLUMNS.

Para obter mais informações sobre como determinar a frequência de execução do VACUUM e quando executar o VACUUM REINDEX, consulte Decidir sobre a reindexação.

Restrições de tabela

As restrições de exclusividade, chave primária e chave estrangeira são apenas informativas; elas não são impostas pelo Amazon Redshift quando você preenche uma tabela. Por exemplo, se você inserir dados em uma tabela com dependências, a inserção poderá ser bem-sucedida mesmo que viole a restrição. Apesar disso, chaves primárias e chaves estrangeiras são usadas como sugestões de planejamento e devem ser declaradas se seu processo de ETL ou algum outro processo em seu aplicativo reforçar sua integridade.

Por exemplo, o planejador de consultas usa chaves primárias e estrangeiras em determinados cálculos estatísticos. Ele faz isso para inferir singularidade e relações referenciais que afetam

técnicas de decoração de subconsulta. Ao fazer isso, ele pode ordenar um grande número de junções e remover junções redundantes.

O planejador aproveita esses relacionamentos de chave, mas assume que todas as chaves nas tabelas do Amazon Redshift são válidas quando carregadas. Se seu aplicativo permitir chaves estrangeiras ou primárias inválidas, algumas consultas podem retornar resultados incorretos. Por exemplo, uma consulta SELECT DISTINCT pode retornar linhas duplicadas se a chave primária não for exclusiva. Não defina restrições de chaves para suas tabelas se você questionar sua validez. Por outro lado, sempre declare chaves primárias e estrangeiras e restrições de exclusividade se você souber que elas são válidas.

O Amazon Redshift impõe restrições de coluna NOT NULL.

Para obter mais informações sobre restrições de tabela, consulte CRIAR TABELA. Para obter informações sobre como descartar uma tabela com dependências, consulte DESCARTAR TABELA.

Carregar dados no Amazon Redshift

Há algumas maneiras de carregar dados em um banco de dados do Amazon Redshift. Uma fonte conhecida de dados a serem carregados são os arquivos do Amazon S3. A tabela a seguir resume alguns métodos a serem usados diretamente de uma fonte do Amazon S3.

Método a ser usado	Descrição	Quando o método é necessário
COPY command	Executa uma ingestão de arquivos em lote para carregar dados dos arquivos do Amazon S3. Esse método utiliza processamento paralelo do Amazon Redshift. Para obter mais informações, consulte Carregar tabelas com o comando COPY.	Deve ser usado quando são necessários requisitos básicos de carregamento de dados para iniciar a ingestão manual de arquivos em lote. Esse método é usado principalmente com pipelines de ingestão de arquivos personalizados e de terceiros ou workloads de ingestão de arquivos únicas ou ad hoc.
COPY... Comando CREATE JOB (cópia automática)	Executa seus comandos COPY automaticamente quando um arquivo é criado em caminhos rastreados do Amazon S3. Para obter mais informações, consulte Criar uma integração de eventos do S3 para copiar automaticamente arquivos de buckets do Amazon S3.	Deve ser usado quando um pipeline de ingestão de arquivos precisa ingerir dados automaticamente quando um arquivo é criado no Amazon S3. O Amazon Redshift monitora os arquivos ingeridos para evitar a duplicação de dados. Esse método requer configuração por proprietários de buckets do Amazon S3.
Carregar diretamente de consultas de data lake	Crie tabelas externas para executar consultas de data lake em seus arquivos do Amazon S3 e, depois, execute	É possível usar qualquer um dos seguintes casos:

Método a ser usado	Descrição	Quando o método é necessário
	o comando INSERT INTO para carregar os resultados de suas consultas de data lake em tabelas locais. Para obter mais informações, consulte Tabelas externas para o Redshift Spectrum.	• Carregar diretamente do AWS Glue e abrir formatos de tabela (como Apache Iceberg, Apache Hudi ou Delta Lake). • Os arquivos de origem precisam ser ingeridos parcialmente (por exemplo, necessários para executar uma cláusula WHERE para ingerir linhas específicas). • É necessária maior flexibilidade para ingerir colunas específicas (como executar um comando SELECT) ou fazer a transformação básica de dados em tempo real (como aplicar operações básicas ou chamar UDFs nos valores do arquivo de origem).
Outros métodos a serem considerados		

Método a ser usado	Descrição	Quando o método é necessário
Ingestão de streaming	A ingestão de streaming oferece ingestão de dados de fluxo de baixa latência e alta velocidade do Amazon Kinesis Data Streams e do Amazon Managed Streaming para Apache Kafka em uma visão materializada provisionada pelo Amazon Redshift ou Amazon Redshift sem servidor. Para obter mais informações, consulte Conceitos básicos da ingestão de streaming do Amazon Kinesis Data Streams e Conceitos básicos da ingestão de streaming de origens do Apache Kafka.	Deve ser considerado para casos de uso em que os dados são primeiro transmitidos para arquivos no Amazon S3 e depois carregados diretamente do Amazon S3. Se não for necessário manter os dados no Amazon S3, muitas vezes é possível transmiti-los diretamente para o Amazon Redshift.
Executar consultas de data lake	Executar consultas diretamente de uma tabela do data lake em vez de ingerir o conteúdo da tabela em uma tabela local. Para obter mais informações, consulte Amazon Redshift Spectrum.	Deve ser usado quando o caso de uso não exige a performance de consultas de tabelas locais no Amazon Redshift.

Método a ser usado	Descrição	Quando o método é necessário
Carregamento em lote com o Editor de Consultas do Amazon Redshift v2	É possível preparar e executar as workloads de ingestão de arquivos em lote visualmente no Editor de Consultas do Amazon Redshift v2. Para ter mais informações, consulte Carregar dados do S3 no Guia de gerenciamento do Amazon Redshift.	Deve ser usado quando você deseja que o Editor de Consultas v2 prepare declarações COPY e deseja uma ferramenta visual para simplificar o processo de preparação da declaração COPY.
Carregar dados de um arquivo local com o Editor de Consultas do Amazon Redshift v2.	É possível fazer upload de arquivos diretamente da área de trabalho para as tabelas do Amazon Redshift sem a necessidade de carregar manualmente os arquivos no Amazon S3. Para ter mais informações, consulte Carregar dados de uma configuração e fluxo de trabalho de arquivo local no Guia de gerenciamento do Amazon Redshift.	Deve ser usado quando você precisa carregar rapidamente arquivos do computador local para fins de consulta única. Com esse método, o Editor de Consultas do Amazon Redshift v2 armazena temporariamente o arquivo em um bucket do Amazon S3 de propriedade do cliente e executa um comando de cópia usando esse caminho do Amazon S3.

Um comando COPY é a forma mais eficiente para carregar uma tabela. Você também pode adicionar dados às suas tabelas usando comandos INSERT, embora isso seja muito menos eficiente que o uso de COPY. O comando COPY é capaz de ler de vários arquivos de dados ou vários fluxos de dados simultaneamente. O Amazon Redshift aloca a workload aos nós do Amazon Redshift e executa as operações de carregamento em paralelo, incluindo a classificação das linhas e a distribuição dos dados entre fatias do nó.

> **Note**
>
> As tabelas externas do Amazon Redshift Spectrum são de somente leitura. Não é possível COPY ou INSERT em uma tabela externa.

Para acessar dados em outros recursos da AWS, o Amazon Redshift deve ter permissão para acessar esses recursos e para realizar as ações necessárias para acessar os dados. É possível usar o AWS Identity and Access Management (IAM) para limitar o acesso que os usuários têm aos recursos e aos dados do Amazon Redshift.

Após o carregamento inicial de seus dados, se você adicionar, modificar ou excluir uma quantia significativa de dados, você deve executar um comando VACUUM para reorganizar seus dados e recuperar espaço após exclusões. Você também deve executar um comando ANALYZE para atualizar as estatísticas da tabela.

Tópicos

- Carregar tabelas com o comando COPY
- Criar uma integração de eventos do S3 para copiar automaticamente arquivos de buckets do Amazon S3
- Carregar tabelas com comandos DML
- Execução de uma cópia profunda
- Análise de tabelas
- Vacuum de tabelas
- Gerenciamento de operações de gravação simultâneas
- Tutorial: Carregar dados do Amazon S3

Carregar tabelas com o comando COPY

O comando COPY aproveita a arquitetura de processamento massivamente paralelo (MPP) do Amazon Redshift para ler e carregar dados em paralelo de arquivos no Amazon S3, de uma tabela do DynamoDB ou de saída de texto de um ou mais hosts remotos.

Antes de conhecer todas as opções do comando COPY, recomendamos aprender as opções básicas para carregar dados do Amazon S3. O Guia de conceitos básicos do Amazon Redshift

demonstra um uso simples do comando COPY para carregar dados do Amazon S3 usando um perfil padrão do IAM. Consulte Etapa 4: Carregar dados do Amazon S3 para o Amazon Redshift para ter detalhes.

> **Note**
>
> Recomendamos veementemente o uso do comando COPY para carregar grandes quantidades de dados. O uso de instruções INSERT individuais para povoar uma tabela pode ser proibitivamente lento. Como alternativa, se seus dados já existem em outras tabelas de banco de dados do Amazon Redshift, use INSERT INTO... SELECT ou CREATE TABLE AS para aprimorar a performance. Para obter informações, consulte INSERT ou CREATE TABLE AS.

Para carregar dados de outro recurso da AWS, o Amazon Redshift deve ter permissão para acessar o recurso e realizar as ações necessárias.

Por conceder ou revogar o privilégio para carregar dados em uma tabela usando um comando COPY, conceda ou revogue o privilégio INSERT.

Seus dados precisam estar no formato adequado para serem carregados na tabela do Amazon Redshift. Esta seção apresenta as diretrizes para preparar e verificar seus dados antes do carregamento e para validar uma instrução COPY antes de executá-la.

Para proteger as informações em seus arquivos, você pode criptografar os arquivos de dados antes de carregá-los em seu bucket do Amazon S3; O COPY irá descriptografar os dados enquanto executa o carregamento. Você também pode limitar o acesso que os usuários têm aos seus dados de carregamento fornecendo credenciais de segurança temporárias aos usuários. As credenciais de segurança temporárias oferecem segurança aprimorada, pois possuem vidas úteis curtas e não podem ser reutilizadas após o vencimento.

O Amazon Redshift conta com recursos integrados ao COPY para carregar rapidamente dados descompactados e delimitados. Mas é possível compactar arquivos usando o gzip, o lzop ou o bzip2 para economizar tempo de carregamento de arquivos.

Se as seguintes palavras-chave ocorrerem na consulta COPY, a divisão automática dos dados não compactados não é compatível: ESCAPE, REMOVEQUOTES e FIXEDWIDTH. Mas a palavra-chave CSV é compatível.

Para ajudar a manter seus dados seguros em trânsito na Nuvem AWS, o Amazon Redshift usa SSL acelerado por hardware para se comunicar com o Amazon S3 ou o Amazon DynamoDB para operações de COPY, UNLOAD, backup e restauração.

Ao carregar sua tabela diretamente de uma tabela do Amazon DynamoDB, você tem a opção de controlar a quantidade do throughput provisionado do Amazon DynamoDB que você consome.

Você pode opcionalmente permitir que COPY analise seus dados de entrada e aplique automaticamente as codificações de compactação ideais em sua tabela como parte do processo de carregamento.

Tópicos

- Credenciais e permissões de acesso
- Preparação de dados de entrada
- Carregar dados do Amazon S3
- Carregar dados do Amazon EMR
- Carregamento de dados de hosts remotos
- Carregar dados de uma tabela do Amazon DynamoDB
- Como verificar se os dados foram carregados corretamente
- Validação de dados de entrada
- Carregamento de tabelas com compactação automática
- Otimização de armazenamento para tabelas estreitas
- Carregamento de valores padrão de coluna
- Solução de problemas de carregamento de dados

Credenciais e permissões de acesso

Para carregar ou descarregar dados usando outro recurso da AWS como Amazon S3, Amazon DynamoDB, Amazon EMR ou Amazon EC2, o Amazon Redshift deve ter permissão para acessar o recurso e realizar as ações necessárias para acessar os dados. Por exemplo, para carregar dados do Amazon S3, COPY deve ter acesso LIST para o bucket e acesso GET para os objetos do bucket.

Para ter autorização para acessar um recurso, o Amazon Redshift deve ser autenticado. Você pode escolher o controle de acesso com base em função ou o controle de acesso com base em chave. Esta seção apresenta uma visão geral dos dois métodos. Para obter os detalhes completos e exemplos, consulte Permissões para acessar outros recursos da AWS.

Controle de acesso com base em função

Com o controle de acesso baseado em perfil, o Amazon Redshift assume temporariamente um perfil do AWS Identity and Access Management (IAM) em seu nome. Então, com base nas autorizações concedidas para o perfil, o Amazon Redshift pode acessar os recursos da AWS necessários.

Recomendamos o uso de controle de acesso baseado em função porque ele fornece um controle mais seguro e refinado de acesso a recursos da AWS e dados confidenciais do usuário, além de proteger suas credenciais da AWS.

Para usar o controle de acesso baseado em perfil, você deve primeiro criar um perfil do IAM usando o tipo de perfil de serviço do Amazon Redshift e, depois, anexar o perfil ao seu data warehouse. A função deve ter, pelo menos, as permissões listadas em Permissões do IAM para COPY, UNLOAD e CREATE LIBRARY. Para ver as etapas de como criar um perfil do IAM e anexá-lo ao cluster, consulte "Criar uma função do IAM para permitir que o cluster do Amazon Redshift acesse serviços da AWS" no Guia de gerenciamento de clusters do Amazon Redshift.

Você pode adicionar uma função a um cluster ou visualizar as funções associadas a um cluster usando o Console de Gerenciamento do Amazon Redshift, a CLI ou a API. Para obter mais informações, consulte "Autorizar operações COPY, UNLOAD, CREATE EXTERNAL FUNCTION e CREATE EXTERNAL SCHEMA usando funções do IAM" no Guia de gerenciamento de clusters do Amazon Redshift.

Quando você cria uma função do IAM, o IAM retorna um nome de recurso da Amazon (ARN) para a função. Para executar o comando COPY usando uma função do IAM, forneça o ARN da função usando o parâmetro IAM_ROLE ou o parâmetro CREDENTIALS.

O seguinte exemplo de comando COPY usa o parâmetro IAM_ROLE com a função `MyRedshiftRole` para autenticação.

```
COPY customer FROM 's3://amzn-s3-demo-bucket/mydata'
IAM_ROLE 'arn:aws:iam::12345678901:role/MyRedshiftRole';
```

O usuário do AWS deve ter, pelo menos, as permissões listadas em Permissões do IAM para COPY, UNLOAD e CREATE LIBRARY.

Se você se conectar a uma identidade federada do IAM, também poderá carregar usando as credenciais da sua sessão atual especificando `IAM_ROLE 'SESSION'` em vez de assumir um perfil do IAM de cluster. Para obter mais informações, consulte SESSION. Para ver um exemplo do uso

de uma identidade federada, consulte Usar uma identidade federada para gerenciar o acesso do Amazon Redshift aos recursos locais e a tabelas externas do Amazon Redshift Spectrum.

Controle de acesso com base em chave

Com o controle de acesso baseado em chave, você fornece o ID da chave de acesso e a chave de acesso secreta para um usuário que está autorizado a acessar os recursos da AWS que contêm os dados.

> **Note**
>
> É altamente recomendável usar uma função do IAM para autenticação em vez de fornecer um ID da chave de acesso de texto simples e uma chave de acesso secreta. Se você escolher o controle de acesso com base em chave, nunca use as credenciais de sua conta da AWS (raiz). Sempre crie um usuário do IAM e forneça o ID da chave de acesso e a chave de acesso secreta desse usuário. Para obter as etapas para criar um usuário do IAM, consulte Criação de um usuário do IAM em sua conta da AWS.

Preparação de dados de entrada

Se seus dados de entrada não forem compatíveis com as colunas da tabela que os receberão, o comando COPY falhará.

Use as diretrizes a seguir para ajudar a garantir que seus dados de entrada sejam válidos:

- Seus dados podem conter somente caracteres UTF-8 com até quatro bytes de comprimento.
- Verifique que as strings CHAR e VARCHAR são mais longas que os comprimentos das colunas correspondentes. Strings VARCHAR são medidas em bytes, não caracteres, portanto, por exemplo, uma string de quatro caracteres chineses que ocupam quatro bytes cada requer uma coluna VARCHAR(16).
- Caracteres multibyte podem ser usados somente com colunas VARCHAR. Verifique que os caracteres multibyte não tenham mais que quatro bytes de comprimento.
- Verifique que os dados para colunas CHAR contenham somente caracteres de único byte.
- Não inclua quaisquer caracteres ou sintaxe especial para indicar o último campo em um registro. Este campo pode ser um delimitador.
- Se seus dados incluem terminais nulos, também chamados de NUL (UTF-8 0000) ou zero binário (0x000), você pode carregar esses caracteres como NULLS em colunas CHAR ou VARCHAR

usando a opção NULL AS no comando COPY: `null as '\0'` ou `null as '\000'`. Se você não utilizar NULL AS, terminais nulos farão com que seu COPY falhe.

- Se suas strings contêm caracteres especiais, tais como delimitadores e novas linhas inseridas, use a opção ESCAPE com o comando COPY.
- Verifique se todas as aspas simples e duplas correspondem adequadamente.
- Verifique que as strings de ponto flutuante estejam no formato de ponto flutuante padrão, tal como 12.123, ou no formato exponencial, tal como 1.0E4.
- Verifique que todas as strings de timestamp e data sigam as especificações para Strings DATEFORMAT e TIMEFORMAT. O formato de timestamp padrão é AAAA-MM-DD hh:mm:ss e formato de data padrão é AAAA-MM-DD.
- Para obter mais informações sobre os limites e limitações dos tipos de dados individuais, consulte Tipos de dados. Para obter mais informações sobre erros de caracteres multibyte, consulte . Erros no carregamento de caracteres multibyte

Carregar dados do Amazon S3

O comando COPY utiliza a arquitetura de processamento paralelo em massa (MPP) do Amazon Redshift para ler e carregar dados em paralelo de um ou mais arquivos em um bucket do Amazon S3. Aproveite ao máximo a vantagem do processamento paralelo dividindo seus dados em vários arquivos, nos casos em que os arquivos são compactados. (Há exceções a essa regra. Elas são detalhados em Carregar arquivos de dados.) Você também pode aproveitar ao máximo a vantagem do processamento paralelo ao configurar chaves de distribuição em suas tabelas. Para obter mais informações sobre chaves de distribuição, consulte Distribuição de dados para otimização de consultas.

Os dados são carregados em uma tabela de destino, uma linha por linha. Os campos no arquivo de dados são correspondidos com as colunas da tabela em ordem, da esquerda para a direita. Os campos nos arquivos de dados podem ser de largura fixa ou delimitados por caracteres; o delimitador padrão é um pipe (|). Por padrão, todas as colunas da tabela são carregadas, mas você pode, opcionalmente, definir uma lista de colunas separadas por vírgula. Se uma tabela não foi incluída na lista de colunas especificada no comando COPY, ela será carregada com um valor padrão. Para obter mais informações, consulte Carregamento de valores padrão de coluna.

Tópicos

- Carregar dados de arquivos compactados e não compactados
- Carregar arquivos no Amazon S3 para usar com COPY

- Usando o comando COPY para carregar do Amazon S3

Carregar dados de arquivos compactados e não compactados

Ao carregar dados compactados, recomendamos dividir os dados de cada tabela em vários arquivos. Quando você carrega dados descompactados e delimitados, o comando COPY usa processamento paralelo em massa (MPP) e intervalos de varredura para carregar dados de arquivos grandes em um bucket do Amazon S3.

Carregar dados de vários arquivos compactados

Caso você tenha dados compactados, recomendamos dividir os dados de cada tabela em vários arquivos. O comando COPY pode carregar dados a partir de vários arquivos em paralelo. Você pode carregar vários arquivos especificando um prefixo comum ou prefixo de chave para o conjunto ou listando explicitamente os arquivos em um arquivo manifesto.

Divida seus dados em arquivos de modo que o número de arquivos seja um múltiplo do número de fatias em seu cluster. Assim, o Amazon Redshift pode dividir os dados igualmente entre as fatias. O número de fatias por nó depende do tamanho do nó do cluster. Por exemplo, cada nó de computação dc2.large tem duas fatias e cada nó de computação dc2.8xlarge tem 16 fatias. Para obter mais informações sobre o número de fatias que cada tamanho de nó possui, consulte “Clusters e nós no Amazon Redshift” no Guia de gerenciamento de clusters do Amazon Redshift.

Todos os nós participam da execução de consultas paralelas, trabalhando nos dados que são distribuídos da maneira mais uniforme possível pelas fatias. Caso tenha um cluster com dois nós dc2.large, você poderá dividir os dados em quatro arquivos ou em um múltiplo de quatro. O Amazon Redshift não leva em conta o tamanho do arquivo ao dividir o workload. Assim, é necessário garantir que os arquivos tenham aproximadamente o mesmo tamanho, de 1 MB a 1 GB após a compactação.

Para usar prefixos de objeto para identificar arquivos de carregamento, nomeie cada arquivo com um prefixo comum. Por exemplo, você poderá dividir o arquivo `venue.txt` em quatro arquivos, do modo a seguir.

```
venue.txt.1
venue.txt.2
venue.txt.3
venue.txt.4
```

Ao colocar vários arquivos em uma pasta de seu bucket, você pode especificar o nome da pasta como o prefixo, e o COPY carregará todos os arquivos na pasta. Se você explicitamente listar os

arquivos a serem carregador usando um arquivo manifesto, os arquivos podem residir em buckets ou pastas diferentes.

Para obter mais informações sobre arquivos manifesto, consulte Example: COPY from Amazon S3 using a manifest.

Carregar dados de arquivos descompactados e delimitados

Ao carregar dados delimitados e descompactados, o comando COPY usa a arquitetura de processamento paralelo em massa (MPP) no Amazon Redshift. O Amazon Redshift usa automaticamente fatias que trabalham em paralelo para carregar intervalos de dados de um arquivo grande para um bucket do Amazon S3. Para que o carregamento paralelo ocorra, o arquivo deve ser delimitado. Por exemplo, delimitado por barras verticais. O carregamento de dados automático e paralelo com o comando COPY também está disponível para arquivos CSV. Você também pode utilizar o processamento paralelo configurando chaves de distribuição em suas tabelas. Para obter mais informações sobre chaves de distribuição, consulte Distribuição de dados para otimização de consultas.

O carregamento paralelo automático não é compatível quando a consulta COPY inclui uma das seguintes palavras-chave: ESCAPE, REMOVEQUOTES e FIXEDWIDTH.

Os dados dos arquivos são carregados em uma tabela de destino, uma linha por linha da tabela. Os campos no arquivo de dados são correspondidos com as colunas da tabela em ordem, da esquerda para a direita. Os campos nos arquivos de dados podem ser de largura fixa ou delimitados por caracteres; o delimitador padrão é um pipe (|). Por padrão, todas as colunas da tabela são carregadas, mas você pode, opcionalmente, definir uma lista de colunas separadas por vírgula. Se a tabela não estiver incluída na lista de colunas especificada no comando COPY, ela será carregada com um valor padrão. Para obter mais informações, consulte Carregamento de valores padrão de coluna.

Siga este processo geral para carregar dados do Amazon S3 quando os dados são descompactados e delimitados:

1. Faça upload de seus arquivos para o Amazon S3.
2. Execute um comando COPY para carregar a tabela.
3. Verifique se os dados foram carregados corretamente.

Para obter exemplos de comandos COPY, consulte Exemplos de COPY. Para obter informações sobre dados carregados no Amazon Redshift, verifique as tabelas de sistema STL_LOAD_COMMITS e STL_LOAD_ERRORS.

Para obter mais informações sobre nós e as fatias contidas em cada um deles, consulte "Clusters e nós no Amazon Redshift" no Guia de gerenciamento de clusters do Amazon Redshift.

Carregar arquivos no Amazon S3 para usar com COPY

Há algumas abordagens a serem adotadas ao carregar arquivos de texto para o Amazon S3:

- Se você tiver arquivos compactados, recomendamos dividir arquivos grandes para aproveitar a vantagem do processamento paralelo no Amazon Redshift.
- No entanto, o COPY divide automaticamente dados de arquivos grandes, descompactados e delimitados por texto para facilitar o paralelismo e distribuir efetivamente dados de arquivos grandes.

Crie um bucket do Amazon S3 para armazenar seus arquivos de dados e, em seguida, carregue os arquivos de dados para o bucket. Para obter informações sobre a criação de buckets e carregamento de arquivos, consulte Trabalhar com buckets do Amazon S3 no Guia do usuário do Amazon Simple Storage Service.

> **Important**
>
> O bucket do Amazon S3 que contém os arquivos de dados deve ser criado na mesma região da AWS do seu cluster, a menos que você use a opção REGION para especificar a região em que o bucket do Amazon S3 está localizado.

Verifique se os intervalos de IP do S3 foram adicionados à sua lista de permissões. Para saber mais sobre os intervalos de IP do S3 necessários, consulte Isolamento de rede.

Você pode criar um bucket do Amazon S3 em uma região específica selecionando a região ao criar o bucket usando o console do Amazon S3 ou especificando um endpoint ao criar o bucket usando a API ou CLI do Amazon S3.

Após o carregamento de dados, verifique se os arquivos corretos estão presentes no Amazon S3.

Tópicos

- Gerenciamento da consistência de dados
- Carregamento de dados criptografados para Amazon S3
- Verificação da presença dos arquivos corretos no bucket

Gerenciamento da consistência de dados

O Amazon S3 oferece uma forte consistência de leitura após gravação para operações COPY, UNLOAD, INSERT (tabela externa), CREATE EXTERNAL TABLE AS e Amazon Redshift Spectrum em buckets do Amazon S3 em todas as regiões da AWS. Além disso, as operações de leitura no Amazon S3 Select, lista de controle de acesso do Amazon S3, Amazon S3 Object Tags e metadados de objeto (por exemplo, objeto HEAD) são fortemente consistentes. Para obter mais informações sobre consistência de dados, consulte Modelo de consistência de dados do Amazon S3 no Guia do usuário do Amazon Simple Storage Service.

Carregamento de dados criptografados para Amazon S3

O Amazon S3 oferece suporte para criptografia do lado do servidor e criptografia do lado do cliente. Este tópico discute as diferenças entre a criptografia do lado do servidor e do lado do cliente e descreve as etapas para usar a criptografia do lado do cliente com o Amazon Redshift. A criptografia do lado do servidor é transparente para o Amazon Redshift.

Criptografia do lado do servidor

A criptografia do lado do servidor é a criptografia de dados em repouso, ou seja, o Amazon S3 criptografa seus dados à medida que os carrega e os descriptografa para você quando você os acessa. Quando você carrega tabelas usando um comando COPY, não há diferença na maneira como você carrega de objetos criptografados ou não criptografados do lado do servidor no Amazon S3. Para obter mais informações sobre a criptografia do lado do servidor, consulte Usar criptografia do lado do servidor no Guia do usuário do Amazon Simple Storage Service.

Criptografia do lado do cliente

Na criptografia no lado do cliente, seu aplicativo cliente gerencia a criptografia de seus dados, chaves de criptografia e ferramentas relacionadas. Você pode fazer upload de dados para um bucket do Amazon S3 usando criptografia do lado do cliente e, em seguida, carregar os dados usando o comando COPY com a opção ENCRYPTED e uma chave de criptografia privada para fornecer maior segurança.

Você criptografa seus dados usando criptografia de envelope. Com criptografia de envelope, seu aplicativo cuida de toda criptografia exclusivamente. Suas chaves de criptografia privadas e seus dados não criptografados nunca são enviados para a AWS, por isso é muito importante que você gerencie com segurança suas chaves de criptografia. Se você perder suas chaves de criptografia, não poderá descriptografar seus dados e não poderá recuperar suas chaves de criptografia da AWS. A criptografia de envelope alia a performance da criptografia simétrica rápida ao mesmo tempo com a maior segurança que o gerenciamento com chaves assimétricas oferece. Uma chave simétrica de uso único (a chave simétrica de envelope) é gerada por seu cliente de criptografia Amazon S3 para criptografar seus dados. Portanto, essa chave é criptografada por sua chave raiz e armazenada com seus dados no Amazon S3. Quando o Amazon Redshift acessa seus dados durante um carregamento, a chave simétrica criptografada é recuperada e descriptografada com sua chave real e, em seguida, os dados são descriptografados.

Para trabalhar com dados criptografados do lado do cliente do Amazon S3 no Amazon Redshift, siga as etapas descritas em Proteger dados usando criptografia do lado do cliente no Guia do usuário do Amazon Simple Storage Service, com os demais requisitos que você usa:

- Criptografia simétrica A classe `AmazonS3EncryptionClient` do SDK for Java da AWS usa criptografia de envelope, descrita anteriormente, que é baseada na criptografia de chave simétrica. Use esta classe para criar um cliente Amazon S3 para carregar dados criptografados do lado do cliente.
- Uma chave mestra simétrica AES de 256 bits: uma chave primária criptografa a chave de envelope. Você transmite a chave raiz à sua instância da classe `AmazonS3EncryptionClient`. Salve esta chave, pois você precisará dela para copiar dados no Amazon Redshift.
- Metadados de objeto para armazenar chave de envelope criptografada – Por padrão, o Amazon S3 armazena a chave de envelope como metadados de objeto para a classe `AmazonS3EncryptionClient`. A chave de envelope criptografada que é armazenada como metadados de objeto é usada durante o processo de descriptografia.

> **Note**
>
> Se você receber uma mensagem de erro de criptografia quando usar a API de criptografia pela primeira vez, sua versão do JDK pode ter um arquivo de políticas de jurisdição JCE (Java Cryptography Extension) que limita o tamanho máximo da chave para transformações de criptografia e descriptografia para 128 bits. Para obter informações sobre como resolver

esse problema, acesse Especificar a criptografia do lado do cliente usando o AWS SDK para Java no Guia do usuário do Amazon Simple Storage Service.

Para obter informações sobre como carregar arquivos criptografados do lado do cliente em suas tabelas do Amazon Redshift usando o comando COPY, consulte Carregar arquivos de dados criptografados do Amazon S3.

Exemplo: upload de dados criptografados no lado do cliente

Para obter um exemplo de como usar o AWS SDK para Java para carregar dados criptografados do lado do cliente, acesse Proteger dados usando criptografia do lado do cliente no Guia do usuário do Amazon Simple Storage Service.

A segunda opção mostra as escolhas que você deve fazer durante a criptografia do lado do cliente para que os dados possam ser carregados no Amazon Redshift. Especificamente, o exemplo mostra o uso de metadados de objeto para armazenar a chave de envelope criptografada e o uso de uma chave raiz simétrica AES de 256 bits.

Este exemplo fornece código de exemplo usando o AWS SDK para Java para criar uma chave raiz simétrica AES de 256 bits e salvá-la em um arquivo. Em seguida, o exemplo faz upload de um objeto no Amazon S3 usando um cliente de criptografia do S3 que inicialmente criptografa dados de amostra no lado do cliente. O exemplo também baixa objeto e verifica se os dados são os mesmos.

Verificação da presença dos arquivos corretos no bucket

Depois de carregar seus arquivos para o bucket do Amazon S3, recomendamos listar o conteúdo do bucket para verificar se todos os arquivos corretos estão presentes e se nenhum arquivo indesejado está presente. Por exemplo, se o bucket `amzn-s3-demo-bucket` tiver um arquivo chamado `venue.txt.back`, esse arquivo será carregado, talvez involuntariamente, pelo seguinte comando:

```
COPY venue FROM 's3://amzn-s3-demo-bucket/venue' … ;
```

Se você quiser controlar especificamente quais arquivos são carregados, você pode usar um arquivo manifesto para listar explicitamente os arquivos de dados. Para obter mais informações sobre o uso de um arquivo manifesto, consulte a opção copy_from_s3_manifest_file para o comando COPY e Example: COPY from Amazon S3 using a manifest nos exemplos de COPY.

Para obter mais informações sobre como listar o conteúdo do bucket, consulte Listagem de chaves de objeto no Guia do desenvolvedor do Amazon S3.

Usando o comando COPY para carregar do Amazon S3

Use o comando COPY para carregar uma tabela em paralelo a partir de arquivos de dados no Amazon S3. Você pode especificar os arquivos a serem carregados usando um prefixo de objeto do Amazon S3 ou usando um arquivo manifesto.

A sintaxe para especificar os arquivos a serem carregados usando um prefixo é a seguinte:

```
COPY <table_name> FROM 's3://<bucket_name>/<object_prefix>'
authorization;
```

O arquivo manifesto é um arquivo em formato JSON que lista os arquivos de dados a serem carregados. A sintaxe para especificar os arquivos a serem carregados usando um arquivo manifesto é a seguinte:

```
COPY <table_name> FROM 's3://<bucket_name>/<manifest_file>'
authorization
MANIFEST;
```

A tabela a ser carregada já deve existir no banco de dados. Para obter informações sobre como criar uma tabela, consulte CRIAR TABELA na referência do SQL.

Os valores para autorização fornecem a autorização da AWS de que o Amazon Redshift precisa para acessar objetos do Amazon S3. Para obter informações sobre as permissões necessárias, consulte Permissões do IAM para COPY, UNLOAD e CREATE LIBRARY. O método preferido de autenticação é especificar o parâmetro IAM_ROLE e fornecer o nome de recurso da Amazon (ARN) para uma função do IAM com as permissões necessárias. Para obter mais informações, consulte Controle de acesso com base em função .

Para autenticação usando o parâmetro IAM_ROLE, substitua *<aws-account-id>* e *<role-name>* conforme exibido na sintaxe a seguir.

```
IAM_ROLE 'arn:aws:iam::<aws-account-id>:role/<role-name>'
```

O seguinte exemplo mostra a autenticação usando uma função do IAM.

```
COPY customer
FROM 's3://amzn-s3-demo-bucket/mydata'
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Para obter mais informações sobre outras opções de autorização, consulte Parâmetros de autorização

Se você quiser validar seus dados sem realmente carregar a tabela, use a opção NOLOAD com o comando COPY.

O seguinte exemplo mostra as primeiras linhas de dados delimitados por pipe em um arquivo chamado `venue.txt`.

```
1|Toyota Park|Bridgeview|IL|0
2|Columbus Crew Stadium|Columbus|OH|0
3|RFK Stadium|Washington|DC|0
```

Antes de enviar o arquivo para o Amazon S3, divida o arquivo em vários arquivos para que o comando COPY possa carregá-lo usando o processamento paralelo. O número de arquivos deve ser um múltiplo do número de fatias em seu cluster. Divida seus arquivos de dados de carregamento para que os arquivos tenham o mesmo tamanho aproximado, entre 1 MB e 1 GB após a compactação. Para obter mais informações, consulte Carregar dados de arquivos compactados e não compactados.

Por exemplo, o arquivo `venue.txt` pode ser dividido em quatro arquivos, da seguinte forma:

```
venue.txt.1
venue.txt.2
venue.txt.3
venue.txt.4
```

O seguinte comando COPY carrega a tabela VENUE usando os dados delimitados por pipe nos arquivos de dados com o prefixo “local” no bucket do Amazon S3 `amzn-s3-demo-bucket`.

> **Note**
>
> O bucket `amzn-s3-demo-bucket` do Amazon S3 nos exemplos a seguir não existe. Para uma amostra dos comandos COPY que usam dados reais em um bucket existente do Amazon S3, consulte Carregar dados de amostra.

```
COPY venue FROM 's3://amzn-s3-demo-bucket/venue'
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
```

```
DELIMITER '|';
```

Se nenhum objeto Amazon S3 com o prefixo “local” existir, o carregamento falhará.

Tópicos

- Uso de um manifesto para especificar arquivos de dados
- Carregar arquivos de dados compactados do Amazon S3
- Carregar dados de largura fixa do Amazon S3
- Carregar dados multibyte do Amazon S3
- Carregar arquivos de dados criptografados do Amazon S3

Uso de um manifesto para especificar arquivos de dados

Você pode usar um manifesto para garantir que o comando COPY carregue todos os arquivos necessários, e somente os arquivos necessários, para um carregamento de dados. Você pode usar um manifesto para carregar vários arquivos de buckets diferentes ou arquivos que não compartilham o mesmo prefixo. Em vez de fornecer um caminho de objeto para o comando COPY, você fornece o nome de um arquivo de texto em formato JSON que lista explicitamente os arquivos a serem carregados. O URL no manifesto deve especificar o nome de bucket e o caminho de objeto completo para o arquivo, e não apenas um prefixo.

Para obter mais informações sobre arquivos manifesto, consulte o exemplo de COPY Usar um manifesto para especificar arquivos de dados.

O seguinte exemplo mostra o JSON para carregar arquivos de diferentes buckets e com nomes de arquivos que começam com carimbos de data.

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket1/2013-10-04-custdata", "mandatory":true},
    {"url":"s3://amzn-s3-demo-bucket1/2013-10-05-custdata", "mandatory":true},
    {"url":"s3://amzn-s3-demo-bucket2/2013-10-04-custdata", "mandatory":true},
    {"url":"s3://amzn-s3-demo-bucket2/2013-10-05-custdata", "mandatory":true}
  ]
}
```

O sinalizador opcional `mandatory` especifica se COPY deve retornar um erro se o arquivo não for localizado. O padrão de `mandatory` é `false`. Independentemente de qualquer configuração obrigatória, COPY será encerrado se nenhum arquivo for encontrado.

O exemplo a seguir executa o comando COPY com o manifesto do exemplo anterior, chamado `cust.manifest`.

```
COPY customer
FROM 's3://amzn-s3-demo-bucket/cust.manifest'
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
MANIFEST;
```

Uso de um manifesto criado por UNLOAD

Um manifesto criado por uma operação UNLOAD usando o parâmetro MANIFEST pode ter chaves que não são necessárias para a operação COPY. Por exemplo, o manifesto UNLOAD a seguir inclui uma chave `meta` que é necessária para uma tabela externa do Amazon Redshift Spectrum e para o carregamento de arquivos de dados em um formato de arquivo ORC ou `Parquet`. A chave `meta` contém uma chave `content_length` com um valor que é o tamanho real do arquivo em bytes. A operação COPY requer somente a chave `url` uma chave `mandatory` opcional.

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket/unload/manifest_0000_part_00", "meta":
 { "content_length": 5956875 }},
    {"url":"s3://amzn-s3-demo-bucket/unload/unload/manifest_0001_part_00", "meta":
 { "content_length": 5997091 }}
 ]
}
```

Para obter mais informações sobre arquivos manifesto, consulte Example: COPY from Amazon S3 using a manifest.

Carregar arquivos de dados compactados do Amazon S3

Para carregar arquivos de dados compactados usando gzip, lzop, ou bzip2, inclua a opção correspondente: GZIP, LZOP ou BZIP2.

Por exemplo, o seguinte comando carrega a partir de arquivos que foram compactados usando lzop.

```
COPY customer FROM 's3://amzn-s3-demo-bucket/customer.lzo'
```

```
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
DELIMITER '|' LZOP;
```

> **Note**
>
> Quando você compacta um arquivo de dados com compactação lzop e usa a opção --filter, não é possível usar o comando COPY.

Carregar dados de largura fixa do Amazon S3

Arquivos de dados de largura fixa têm comprimentos uniformes para cada coluna de dados. Cada campo em um arquivo de dados de largura fixa tem exatamente o mesmo comprimento e posição. Para dados de caracteres (CHAR e VARCHAR) em um arquivo de dados de largura fixa, você deve incluir espaços em branco iniciais ou finais como espaços reservados para manter a largura uniforme. Para números inteiros, você deve usar zeros iniciais como espaços reservados. Um arquivo de dados de largura fixa não tem qualquer delimitador para separar colunas.

Para carregar um arquivo de dados de largura fixa em uma tabela existente, USE o parâmetro FIXEDWIDTH no comando COPY. As especificações de sua tabela devem corresponder ao valor de fixedwidth_spec para que os dados sejam carregados corretamente.

Para carregar dados de largura fixa de um arquivo para uma tabela, emita o seguinte comando:

```
COPY table_name FROM 's3://amzn-s3-demo-bucket/prefix'
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
FIXEDWIDTH 'fixedwidth_spec';
```

O parâmetro fixedwidth_spec é uma string que contém um identificador para cada coluna e a largura de cada coluna, separados por um sinal de dois pontos. Os pares de **`column:width`** são delimitados por vírgulas. O identificador pode ser o que você escolher: números, letras ou uma combinação dos dois. O identificador não tem qualquer relação com a própria tabela, portanto a especificação deve conter as colunas na mesma ordem que a tabela.

Os dois exemplos a seguir mostram a mesma especificação, a primeira usando identificadores numéricos e a segunda usando identificadores de string:

```
'0:3,1:25,2:12,3:2,4:6'
```

```
'venueid:3,venuename:25,venuecity:12,venuestate:2,venueseats:6'
```

O seguinte exemplo exibe dados de amostra de largura fixa que poderiam ser carregados na tabela VENUE usando as especificações precedentes:

```
1  Toyota Park               Bridgeview  IL0
2  Columbus Crew Stadium     Columbus    OH0
3  RFK Stadium               Washington  DC0
4  CommunityAmerica Ballpark Kansas City KS0
5  Gillette Stadium          Foxborough  MA68756
```

O seguinte comando COPY carrega este conjunto de dados na tabela VENUE:

```
COPY venue
FROM 's3://amzn-s3-demo-bucket/data/venue_fw.txt'
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
FIXEDWIDTH 'venueid:3,venuename:25,venuecity:12,venuestate:2,venueseats:6';
```

Carregar dados multibyte do Amazon S3

Se seus dados incluem caracteres multibyte não ASCII (tal como caracteres chineses ou cirílicos), você deve carregar os dados em colunas VARCHAR. O tipo de dados VARCHAR é compatível com caracteres UTF-8 de quatro bytes, mas o tipo de dados CHAR aceita apenas caracteres ASCII de único byte. Você não pode carregar caracteres de cinco bytes ou mais nas tabelas do Amazon Redshift. Para obter mais informações sobre CHAR e VARCHAR, consulte Tipos de dados.

Para verificar qual codificação um arquivo de entrada usa, use o comando *file* do Linux:

```
$ file ordersdata.txt
ordersdata.txt: ASCII English text
$ file uni_ordersdata.dat
uni_ordersdata.dat: UTF-8 Unicode text
```

Carregar arquivos de dados criptografados do Amazon S3

Você pode usar o comando COPY para carregar arquivos de dados que foram carregados no Amazon S3 usando criptografia do lado do servidor, criptografia do lado do cliente ou ambos.

O comando COPY é compatível com os seguintes tipos de criptografia do Amazon S3:

- Criptografia do lado do servidor com chaves gerenciadas pelo Amazon S3 (SSE-S3)
- Criptografia do lado do servidor com o AWS KMS keys (SSE-KMS)
- Criptografia do lado do cliente usando uma chave raiz simétrica do lado do cliente

O comando COPY não é compatível com os seguintes tipos de criptografia do Amazon S3:

- Criptografia do lado do servidor com chaves fornecidas pelo cliente (SSE-C)
- Criptografia do lado do cliente usando uma AWS KMS key
- Criptografia do lado do cliente usando uma chave raiz assimétrica fornecida pelo cliente

Para obter mais informações sobre a criptografia do Amazon S3, consulte Proteger dados usando criptografia do lado do servidor e Proteger dados usando criptografia do lado do cliente no Guia do usuário do Amazon Simple Storage Service.

O comando UNLOAD criptografa arquivos automaticamente usando SSE-S3. Você também pode descarregar usando SSE-KMS ou criptografia do lado do cliente com uma chave simétrica gerenciada pelo cliente. Para obter mais informações, consulte . Descarregamento de arquivos de dados criptografados

O comando COPY automaticamente reconhece e carrega arquivos de dados criptografados usando SSE-S3 e SSE-KMS. Você pode carregar arquivos criptografados usando uma chave raiz simétrica do lado do cliente, especificando a opção ENCRYPTED e fornecendo a chave-valor. Para obter mais informações, consulte Carregamento de dados criptografados para Amazon S3.

Para carregar arquivos de dados criptografados do lado do cliente, forneça a chave-valor raiz usando o parâmetro MASTER_SYMMETRIC_KEY e inclua a opção ENCRYPTED.

```
COPY customer FROM 's3://amzn-s3-demo-bucket/encrypted/customer'
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
MASTER_SYMMETRIC_KEY '<root_key>'
ENCRYPTED
DELIMITER '|';
```

Para carregar arquivos de dados criptografados que estão compactados por gzip, lzop ou bzip2, inclua a opção GZIP, LZOP ou BZIP2 com a chave-valor raiz e a opção ENCRYPTED.

```
COPY customer FROM 's3://amzn-s3-demo-bucket/encrypted/customer'
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
```

```
MASTER_SYMMETRIC_KEY '<root_key>'
ENCRYPTED
DELIMITER '|'
GZIP;
```

Carregar dados do Amazon EMR

Você pode usar o comando COPY para carregar dados em paralelo de um cluster do Amazon EMR configurado para gravar arquivos de texto no Sistema de Arquivos Distribuído do Hadoop (HDFS) do cluster na forma de arquivos de largura fixa, arquivos delimitados por caracteres, arquivos CSV ou com formato JSON.

Processo de carregamento de dados do Amazon EMR

Esta seção o orienta no processo de carregamento de dados de um cluster Amazon EMR. As seções a seguir fornecem os detalhes de que você precisa para realizar cada etapa.

- Etapa 1: Configurar permissões do IAM

  Os usuários que criam o cluster do Amazon EMR e executam o comando COPY do Amazon Redshift devem ter as permissões necessárias.
- Etapa 2: Criar um cluster do Amazon EMR

  Configure o cluster para enviar arquivos de texto para o Hadoop Distributed File System (HDFS). Você precisará do ID do cluster do Amazon EMR e do DNS público principal do cluster (o endpoint da instância do Amazon EC2 que hospeda o cluster).
- Etapa 3: Recuperar a chave pública do cluster do Amazon Redshift e os endereços IP do nó do cluster

  A chave pública permite que os nós de cluster do Amazon Redshift estabeleçam conexões SSH com os hosts. Você usará o endereço IP de cada nó do cluster para configurar os grupos de segurança do host para permitir o acesso de seu cluster Amazon Redshift usando esses endereços IP.
- Etapa 4: Adicionar a chave pública do cluster do Amazon Redshift a cada arquivo de chaves autorizadas do host do Amazon EC2

  Você adiciona a chave pública do cluster do Amazon Redshift ao arquivo de chaves autorizadas do host para que o host reconheça o cluster do Amazon Redshift e aceite a conexão SSH.
- Etapa 5: Configurar os hosts para aceitar todos os endereços IP do cluster do Amazon Redshift

Modifique os grupos de segurança da instância do Amazon EMR para adicionar regras de entrada para aceitar os endereços IP do Amazon Redshift.

- Etapa 6: Executar o comando COPY para carregar os dados

  De um banco de dados do Amazon Redshift, execute o comando COPY para carregar os dados em uma tabela do Amazon Redshift.

Etapa 1: Configurar permissões do IAM

Os usuários que criam o cluster do Amazon EMR e executam o comando COPY do Amazon Redshift devem ter as permissões necessárias.

Para configurar permissões do IAM

1. Adicione as permissões a seguir para o usuário que criará o cluster do Amazon EMR.

   ```
   ec2:DescribeSecurityGroups
   ec2:RevokeSecurityGroupIngress
   ec2:AuthorizeSecurityGroupIngress
   redshift:DescribeClusters
   ```

2. Adicione a seguinte permissão para o usuário ou perfil do IAM que executará o comando COPY.

   ```
   elasticmapreduce:ListInstances
   ```

3. Adicione a permissão a seguir à função do IAM do cluster do Amazon EMR.

   ```
   redshift:DescribeClusters
   ```

Etapa 2: Criar um cluster do Amazon EMR

O comando COPY carrega dados de arquivos no Amazon EMR Hadoop Distributed File System (HDFS). Ao criar o cluster Amazon EMR, configure-o para enviar arquivos de dados para o HDFS do cluster.

Para criar um cluster do Amazon EMR

1. Crie um cluster do Amazon EMR na região da AWS como cluster do Amazon Redshift.

Se o cluster do Amazon Redshift estiver em uma VPC, o cluster do Amazon EMR deverá estar no mesmo grupo de VPC. Se o cluster do Amazon Redshift usa o modo EC2-Classic (ou seja, não está em um VPC), o cluster do Amazon EMR também deve usar o modo EC2-Classic. Para obter mais informações, consulte "Gerenciamento de clusters em uma VPC" no Guia de gerenciamento de clusters do Amazon Redshift.

2. Configure o cluster para enviar arquivos de dados para o HDFS do cluster. Os nomes de arquivos do HDFS não devem conter asteriscos (*) ou pontos de interrogação (?).

> **Important**
>
> Os nomes de arquivos não devem conter asteriscos (*) ou pontos de interrogação (?).

3. Especifique No (Não) para a opção Auto-terminate (Terminar automaticamente) na configuração de cluster do Amazon EMR para que o cluster permaneça disponível enquanto o comando COPY for executado.

> **Important**
>
> Se um dos arquivos de dados for alterado ou excluído antes de COPY ser concluído, você poderá ter resultados inesperados ou a operação COPY poderá falhar.

4. Observe o ID do cluster e o DNS público primário (o endpoint da instância do Amazon EC2 que hospeda o cluster). Você usará essas informações em etapas subsequentes.

Etapa 3: Recuperar a chave pública do cluster do Amazon Redshift e os endereços IP do nó do cluster

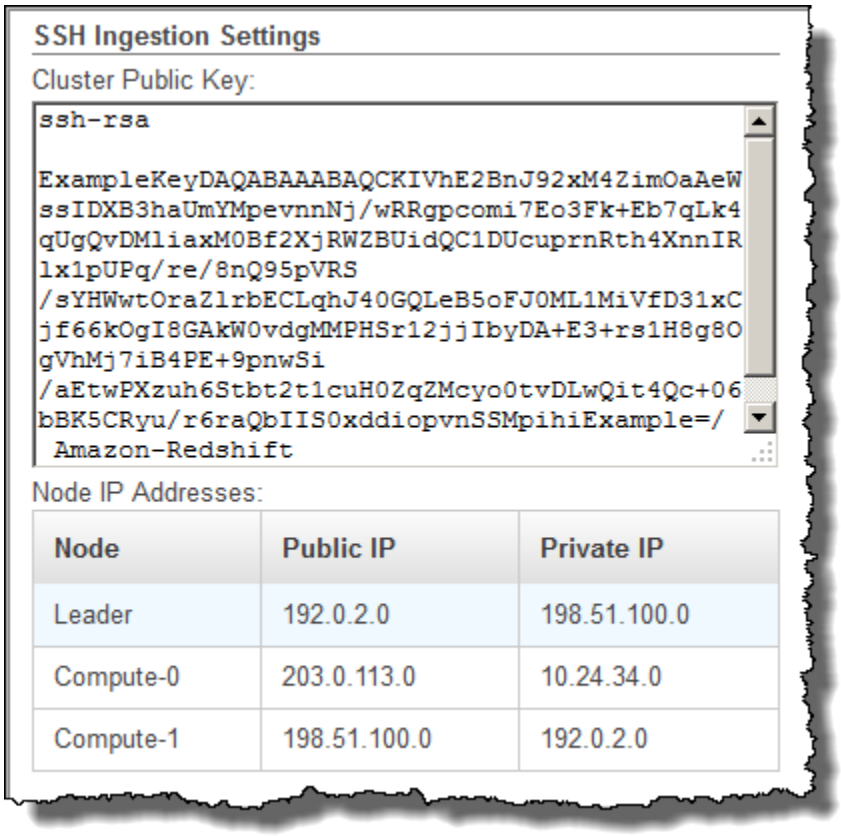
Você usará o endereço IP de cada nó do cluster para configurar os grupos de segurança do host para permitir o acesso de seu cluster Amazon Redshift usando esses endereços IP.

Para recuperar a chave pública do cluster do Amazon Redshift e os endereços IP do nó do cluster para o seu cluster usando o console

1. Acesse o Console de Gerenciamento do Amazon Redshift.
2. No painel de navegação, selecione o link Clusters.
3. Selecione seu cluster na lista.

4. Localize o grupo Configurações de ingestão do SSH.

   Observe a Chave pública do cluster e Endereços IP dos nós. Você vai usá-los em etapas subsequentes.



   Você usará os endereços IP privados na Etapa 3 para configurar o host do Amazon EC2 para aceitar a conexão do Amazon Redshift.

Para recuperar a chave pública do cluster e os endereços IP do nó do cluster para seu cluster usando a CLI do Amazon Redshift, execute o comando describe-clusters. Por exemplo:

```
aws redshift describe-clusters --cluster-identifier <cluster-identifier>
```

A resposta incluirá um valor de ClusterPublicKey e a lista de endereços IP privados e públicos, semelhante ao seguinte:

```
{
    "Clusters": [
        {
            "VpcSecurityGroups": [],
```

```
            "ClusterStatus": "available",
            "ClusterNodes": [
                {
                    "PrivateIPAddress": "10.nnn.nnn.nnn",
                    "NodeRole": "LEADER",
                    "PublicIPAddress": "10.nnn.nnn.nnn"
                },
                {
                    "PrivateIPAddress": "10.nnn.nnn.nnn",
                    "NodeRole": "COMPUTE-0",
                    "PublicIPAddress": "10.nnn.nnn.nnn"
                },
                {
                    "PrivateIPAddress": "10.nnn.nnn.nnn",
                    "NodeRole": "COMPUTE-1",
                    "PublicIPAddress": "10.nnn.nnn.nnn"
                }
            ],
            "AutomatedSnapshotRetentionPeriod": 1,
            "PreferredMaintenanceWindow": "wed:05:30-wed:06:00",
            "AvailabilityZone": "us-east-1a",
            "NodeType": "dc2.large",
            "ClusterPublicKey": "ssh-rsa AAAABexamplepublickey...Y3TAl Amazon-
Redshift",
             ...
             ...
}
```

Para recuperar a chave pública do cluster e os endereços IP do nó do cluster para o seu cluster usando a API do Amazon Redshift, use a ação `DescribeClusters`. Para obter mais informações, consulte describe-clusters no Guia de CLI do Amazon Redshift ou DescribeClusters no Guia de API do Amazon Redshift.

Etapa 4: Adicionar a chave pública do cluster do Amazon Redshift a cada arquivo de chaves autorizadas do host do Amazon EC2

Você adiciona a chave pública do cluster ao arquivo de chaves autorizadas de cada host para todos os nós do cluster do Amazon EMR para que os hosts reconheçam o Amazon Redshift e aceitem a conexão SSH.

Para adicionar a chave pública do cluster do Amazon Redshift ao arquivo de chaves autorizadas do host

1. Acesse o host usando uma conexão SSH.

   Para obter informações sobre como se conectar a uma instância usando SSH, consulte Conectar-se à instância do Linux no Manual do usuário do Amazon EC2.
2. Copie a chave pública do Amazon Redshift do console ou do texto de resposta da CLI.
3. Copie e cole os conteúdos da chave pública no arquivo `/home/<ssh_username>/.ssh/authorized_keys` no host. Inclua a string completa com o prefixo "`ssh-rsa`" e o sufixo "`Amazon-Redshift`". Por exemplo:

   ```
   ssh-rsa AAAACTP3isxgGzVWoIWpbVvRCOzYdVifMrh… uA70BnMHCaMiRdmvsDOedZDOedZ Amazon-
   Redshift
   ```

Etapa 5: Configurar os hosts para aceitar todos os endereços IP do cluster do Amazon Redshift

Para permitir o tráfego de entrada para as instâncias do host, edite o grupo de segurança e adicione uma regra de entrada para cada nó de cluster do Amazon Redshift. Para Tipo, selecione SSH com protocolo TCP na porta 22. Em Source (Fonte), insira os endereços IP privados do nó do cluster do Amazon Redshift que você recuperou em Etapa 3: Recuperar a chave pública do cluster do Amazon Redshift e os endereços IP do nó do cluster. Para obter informações sobre como adicionar regras a um grupo de segurança do Amazon EC2, consulte Autorizar tráfego de entrada para as instâncias no Manual do usuário do Amazon EC2.

Etapa 6: Executar o comando COPY para carregar os dados

Execute um comando COPY para se conectar ao cluster do Amazon EMR e carregar os dados em uma tabela do Amazon Redshift. O cluster do Amazon EMR deve continuar em execução até que o comando COPY seja concluído. Por exemplo, não configure o encerramento automático do cluster.

> **Important**
>
> Se um dos arquivos de dados for alterado ou excluído antes de COPY ser concluído, você poderá ter resultados inesperados ou a operação COPY poderá falhar.

No comando COPY, especifique o ID do cluster Amazon EMR e o caminho do arquivo HDFS e o nome do arquivo.

```
COPY sales
FROM 'emr://myemrclusterid/myoutput/part*' CREDENTIALS
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Você pode usar caracteres curinga asterisco (*) e ponto de interrogação (?) como parte do argumento do nome do arquivo. Por exemplo, `part*` carrega os arquivos `part-0000`, `part-0001` e assim por diante. Se você especificar somente um nome de pasta, COPY tentará carregar todos os arquivos na pasta.

> **Important**
>
> Se você usar caracteres curinga ou usar somente o nome da pasta, certifique-se de que nenhum arquivo indesejado seja carregado ou o comando COPY falhará. Por exemplo, alguns processos podem gravar um arquivo de log na pasta de saída.

Carregamento de dados de hosts remotos

Você pode usar o comando COPY para carregar dados em paralelo de um ou mais hosts remotos, como instâncias do Amazon EC2 ou outros computadores. O COPY se conecta aos hosts remotos usando SSH e executa comandos nos hosts remotos para gerar a saída de texto.

O host remoto pode ser uma instância do Amazon EC2 Linux ou outro computador Unix ou Linux configurado para aceitar conexões SSH. Este guia assume que seu host remoto é uma instância do Amazon EC2. Quando o procedimento for diferente para outro computador, o guia indicará a diferença.

O Amazon Redshift pode conectar-se a vários hosts e abrir várias conexões SSH para cada host. O Amazon Redshifts envia um comando exclusivo por meio de cada conexão a fim de gerar saída de texto para a saída padrão do host, que o Amazon Redshift lê como faria com um arquivo de texto.

Antes de começar

Antes de começar, você deve ter o seguinte:

- Uma ou mais máquinas host, como instâncias do Amazon EC2, às quais você pode se conectar usando SSH.

- Fontes de dados nos hosts.

  Você fornecerá comandos que o cluster do Amazon Redshift executará nos hosts para gerar a saída de texto. Depois que o cluster se conecta a um host, o comando COPY executa os comandos, lê o texto da saída padrão dos hosts e carrega os dados em paralelo em uma tabela do Amazon Redshift. A saída de texto deve estar em um formulário que o comando COPY possa ingerir. Para obter mais informações, consulte . Preparação de dados de entrada
- Acesso aos hosts para seu computador.

  Para uma instância do Amazon EC2, você usará uma conexão SSH para acessar o host. Você precisará acessar o host para adicionar a chave pública do cluster do Amazon Redshift ao arquivo de chaves autorizadas do host.
- Um cluster do Amazon Redshift em execução.

  Para obter informações sobre como iniciar um cluster, consulte o Guia de conceitos básicos do Amazon Redshift.

Processo de carregamento de dados

Esta seção orienta você pelo processo de carregamento de dados de hosts remotos. As seções a seguir fornecem os detalhes de que você precisa para realizar cada etapa.

- Etapa 1: Recuperar a chave pública do cluster e os endereços IP dos nós do cluster

  A chave pública permite que os nós de cluster do Amazon Redshift estabeleçam conexões SSH com os hosts remotos. Você usará o endereço IP de cada nó do cluster para configurar os grupos de segurança do host ou firewall para permitir o acesso de seu cluster Amazon Redshift usando esses endereços IP.
- Etapa 2: Adicionar a chave pública do cluster do Amazon Redshift ao arquivo de chaves autorizadas do host

  Você adiciona a chave pública do cluster do Amazon Redshift ao arquivo de chaves autorizadas do host para que o host reconheça o cluster do Amazon Redshift e aceite a conexão SSH.
- Etapa 3: Configurar o host para aceitar todos os endereços IP do cluster do Amazon Redshift

  Para Amazon EC2, modifique os grupos de segurança da instância para adicionar regras de entrada para aceitar os endereços IP do Amazon Redshift. Para outros hosts, modifique o firewall para que os nós do Amazon Redshift possam estabelecer conexões SSH com o host remoto.

- Etapa 4: Obter a chave pública para o host

  Você pode, opcionalmente, especificar que o Amazon Redshift deve usar a chave pública para identificar o host. Você precisará localizar a chave pública e copiar o texto em seu arquivo de manifesto.

- Etapa 5: Criar um arquivo manifesto

  O manifesto é um arquivo de texto formatado em JSON com os detalhes de que o Amazon Redshift precisa para se conectar aos hosts e buscar os dados.

- Etapa 6: Carregar o arquivo manifesto para um bucket do Amazon S3

  O Amazon Redshift lê o manifesto e usa essas informações para se conectar ao host remoto. Se o bucket do Amazon S3 não residir na mesma região que seu cluster Amazon Redshift, você deve usar a opção REGION para especificar a região na qual os dados estão localizados.

- Etapa 7: Executar o comando COPY para carregar os dados

  De um banco de dados do Amazon Redshift, execute o comando COPY para carregar os dados em uma tabela do Amazon Redshift.

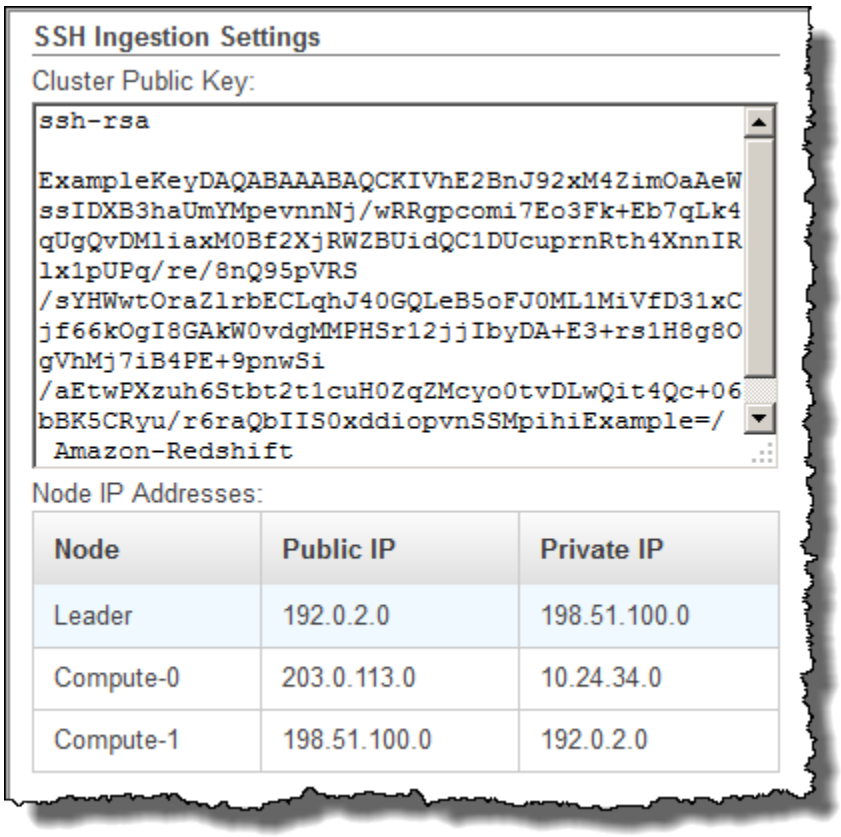
Etapa 1: Recuperar a chave pública do cluster e os endereços IP dos nós do cluster

Você usará o endereço IP de cada nó do cluster para configurar os grupos de segurança do host para permitir o acesso de seu cluster Amazon Redshift usando esses endereços IP.

Para recuperar a chave pública do cluster e os endereços IP dos nós de seu cluster usando o console

1. Acesse o Console de Gerenciamento do Amazon Redshift.
2. No painel de navegação, selecione o link Clusters.
3. Selecione seu cluster na lista.
4. Localize o grupo Configurações de ingestão do SSH.

   Observe a Chave pública do cluster e Endereços IP dos nós. Você vai usá-los em etapas subsequentes.



SSH Ingestion Settings

Cluster Public Key:

```
ssh-rsa

ExampleKeyDAQABAAABAQCKIVhE2BnJ92xM4ZimOaAeW
ssIDXB3haUmYMpevnnNj/wRRgpcomi7Eo3Fk+Eb7qLk4
qUgQvDMliaxM0Bf2XjRWZBUidQC1DUcuprnRth4XnnIR
lx1pUPq/re/8nQ95pVRS
/sYHWwtOraZlrbECLqhJ40GQLeB5oFJ0ML1MiVfD31xC
jf66kOgI8GAkW0vdgMMPHSr12jjIbyDA+E3+rs1H8g8O
gVhMj7iB4PE+9pnwSi
/aEtwPXzuh6Stbt2t1cuH0ZqZMcyo0tvDLwQit4Qc+06
bBK5CRyu/r6raQbIIS0xddiopvnSSMpihiExample=/
 Amazon-Redshift
```

Node IP Addresses:

Node	Public IP	Private IP
Leader	192.0.2.0	198.51.100.0
Compute-0	203.0.113.0	10.24.34.0
Compute-1	198.51.100.0	192.0.2.0

Você usará os endereços IP na Etapa 3 para configurar o host para aceitar a conexão do Amazon Redshift. Dependendo à qual tipo de host você se conectar e se ele está em uma VPC, você usará endereços IP públicos ou endereços IP privados.

Para recuperar a chave pública do cluster e os endereços IP do nó do cluster para seu cluster usando a CLI do Amazon Redshift, execute o comando describe-clusters.

Por exemplo:

```
aws redshift describe-clusters --cluster-identifier <cluster-identifier>
```

A resposta incluirá o valor de ClusterPublicKey e a lista de endereços IP privados e públicos, semelhante ao seguinte:

```
{
    "Clusters": [
        {
            "VpcSecurityGroups": [],
            "ClusterStatus": "available",
```

```
            "ClusterNodes": [
                {
                    "PrivateIPAddress": "10.nnn.nnn.nnn",
                    "NodeRole": "LEADER",
                    "PublicIPAddress": "10.nnn.nnn.nnn"
                },
                {
                    "PrivateIPAddress": "10.nnn.nnn.nnn",
                    "NodeRole": "COMPUTE-0",
                    "PublicIPAddress": "10.nnn.nnn.nnn"
                },
                {
                    "PrivateIPAddress": "10.nnn.nnn.nnn",
                    "NodeRole": "COMPUTE-1",
                    "PublicIPAddress": "10.nnn.nnn.nnn"
                }
            ],
            "AutomatedSnapshotRetentionPeriod": 1,
            "PreferredMaintenanceWindow": "wed:05:30-wed:06:00",
            "AvailabilityZone": "us-east-1a",
            "NodeType": "dc2.large",
            "ClusterPublicKey": "ssh-rsa AAAABexamplepublickey...Y3TAl Amazon-
Redshift",
             ...
             ...
}
```

Para recuperar a chave pública do cluster e os endereços IP do nó do cluster para o seu cluster usando a API Amazon Redshift, use a ação DescribeClusters. Para obter mais informações, consulte describe-clusters no Guia de CLI do Amazon Redshift ou DescribeClusters no Guia de API do Amazon Redshift.

Etapa 2: Adicionar a chave pública do cluster do Amazon Redshift ao arquivo de chaves autorizadas do host

Você adiciona a chave pública do cluster ao arquivo de chaves autorizadas de cada host para que o host reconheça o Amazon Redshift e aceite a conexão SSH.

Para adicionar a chave pública do cluster do Amazon Redshift ao arquivo de chaves autorizadas do host

1. Acesse o host usando uma conexão SSH.

Para obter informações sobre como se conectar a uma instância usando SSH, consulte Conectar-se à instância do Linux no Manual do usuário do Amazon EC2.

2. Copie a chave pública do Amazon Redshift do console ou do texto de resposta da CLI.
3. Copie e cole os conteúdos da chave pública no arquivo `/home/<ssh_username>/.ssh/authorized_keys` no host remoto. O `<ssh_username>` deve corresponder ao valor do campo "nome de usuário" no arquivo manifesto. Inclua a string completa com o prefixo "`ssh-rsa`" e o sufixo "`Amazon-Redshift`". Por exemplo:

```
ssh-rsa AAAACTP3isxgGzVWoIWpbVvRCOzYdVifMrh… uA70BnMHCaMiRdmvsDOedZDOedZ Amazon-Redshift
```

Etapa 3: Configurar o host para aceitar todos os endereços IP do cluster do Amazon Redshift

Se você estiver trabalhando com uma instância do Amazon EC2 ou um cluster do Amazon EMR, adicione regras de entrada ao grupo de segurança do host para permitir o tráfego de cada nó do cluster do Amazon Redshift. Para Tipo, selecione SSH com protocolo TCP na porta 22. Para Fonte, insira os endereços IP do nó do cluster do Amazon Redshift que você recuperou em Etapa 1: Recuperar a chave pública do cluster e os endereços IP dos nós do cluster. Para obter informações sobre como adicionar regras a um grupo de segurança do Amazon EC2, consulte Autorizar tráfego de entrada para as instâncias no Manual do usuário do Amazon EC2.

Use os endereços IP privados quando:

- Você tem um cluster do Amazon Redshift que não está em uma Virtual Private Cloud (VPC) e uma instância do Amazon EC2-Classic, ambos na mesma região da AWS.
- Você tem um cluster do Amazon Redshift que está em um VPC e uma instância do Amazon EC2 - VPC, ambos na mesma região da AWS e no mesmo VPC.

Caso contrário, use os endereços IP públicos.

Para obter mais informações sobre como usar o Amazon Redshift em uma VPC, consulte "Gerenciamento de clusters em uma VPC" no Guia de gerenciamento de clusters do Amazon Redshift.

Etapa 4: Obter a chave pública para o host

Opcionalmente, você pode fornecer a chave pública do host no arquivo de manifesto para que o Amazon Redshift possa identificar o host. O comando COPY não requer a chave pública do host, mas por motivo de segurança, recomendamos veementemente o uso de uma chave pública para ajudar a evitar ataques “man-in-the-middle”.

Você pode encontrar a chave pública do host no seguinte local, onde `<ssh_host_rsa_key_name>` é o nome exclusivo para a chave pública do host:

```
:  /etc/ssh/<ssh_host_rsa_key_name>.pub
```

> **Note**
>
> O Amazon Redshift é compatível somente com chaves RSA. O Amazon Redshift não oferece suporte a chaves DSA.

Ao criar seu arquivo manifesto na etapa 5, você colará o texto da chave pública no campo “chave pública” na entrada do arquivo manifesto.

Etapa 5: Criar um arquivo manifesto

O comando COPY pode conectar-se a vários hosts usando SSH e criar várias conexões SSH para cada host. O COPY executa um comando em cada conexão do host e carrega a saída dos comandos para a tabela em paralelo. O arquivo manifesto é um arquivo de texto no formato JSON que o Amazon Redshift usa para se conectar ao host. O arquivo manifesto especifica os endpoints do host SSH e os comandos que serão executados nos hosts para retornar dados ao Amazon Redshift. Você também pode incluir a chave pública do host, o nome de usuário de login e um sinalizador obrigatório para cada entrada.

Crie o arquivo manifesto no computador local. Em uma etapa posterior, você carrega o arquivo no Amazon S3.

O arquivo manifesto está no seguinte formato:

```
{
   "entries": [
     {"endpoint":"<ssh_endpoint_or_IP>",
```

```
        "command": "<remote_command>",
        "mandatory":true,
        "publickey": "<public_key>",
        "username": "<host_user_name>"},
      {"endpoint":"<ssh_endpoint_or_IP>",
        "command": "<remote_command>",
        "mandatory":true,
        "publickey": "<public_key>",
        "username": "host_user_name"}
    ]
}
```

O arquivo manifesto contém um construto "entradas" para cada conexão SSH. Cada entrada representa uma única conexão SSH. Você pode ter várias conexões para um único host ou várias conexões para vários hosts. As aspas duplas são obrigatórias conforme mostrado, tanto para os nomes dos campos quanto para os valores. O único valor que não precisa de aspas duplas é o valor booleano **`true`** ou **`false`** para o campo obrigatório.

Veja a seguir as descrições dos campos no arquivo manifesto.

endpoint

O endereço URL ou endereço IP do host. Por exemplo, "`ec2-111-222-333.compute-1.amazonaws.com`" ou "`22.33.44.56`"

command

O comando que será executado pelo host para gerar a saída de texto ou a saída binária (gzip, lzop ou bzip2). O comando pode ser qualquer um que o usuário "host_user_name" tenha permissão para executar. O comando pode ser tão simples quanto imprimir um arquivo ou pode ser consultar um banco de dados ou iniciar um script. A saída (arquivo de texto, arquivo binário gzip, arquivo binário lzop ou arquivo bzip2) deve estar em um formato que o comando COPY do Amazon Redshift possa ingerir. Para obter mais informações, consulte Preparação de dados de entrada.

publickey

(Opcional) A chave pública do host. Se fornecida, o Amazon Redshift usará a chave pública para identificar o host. Se a chave pública não for fornecida, o Amazon Redshift não tentará a identificação do host. Por exemplo, se a chave pública do host remoto for: `ssh-rsa AbcCbaxxx…xxxDHKJ root@amazon.com`, insira o seguinte texto no campo de chave pública: `AbcCbaxxx…xxxDHKJ`.

mandatory

(Opcional) Indica se o comando COPY deve falhar se a conexão falhar. O padrão é `false`. Se o Amazon Redshift não fizer com sucesso pelo menos uma conexão, o comando COPY falhará.

username

(Opcional) O nome de usuário que será utilizado para fazer logon no sistema do host e executar o comando remoto. O nome de login do usuário deve ser o mesmo do login que foi usado para adicionar a chave pública ao arquivo de chaves autorizadas do host na etapa 2. O nome de usuário padrão é "redshift".

O exemplo a seguir mostra um manifesto concluído para abrir quatro conexões ao mesmo host e executar um comando diferente em cada conexão:

```
{
  "entries": [
       {"endpoint":"ec2-184-72-204-112.compute-1.amazonaws.com",
          "command": "cat loaddata1.txt",
          "mandatory":true,
          "publickey": "ec2publickeyportionoftheec2keypair",
          "username": "ec2-user"},
       {"endpoint":"ec2-184-72-204-112.compute-1.amazonaws.com",
          "command": "cat loaddata2.txt",
          "mandatory":true,
          "publickey": "ec2publickeyportionoftheec2keypair",
          "username": "ec2-user"},
       {"endpoint":"ec2-184-72-204-112.compute-1.amazonaws.com",
          "command": "cat loaddata3.txt",
          "mandatory":true,
          "publickey": "ec2publickeyportionoftheec2keypair",
          "username": "ec2-user"},
       {"endpoint":"ec2-184-72-204-112.compute-1.amazonaws.com",
          "command": "cat loaddata4.txt",
          "mandatory":true,
          "publickey": "ec2publickeyportionoftheec2keypair",
          "username": "ec2-user"}
     ]
}
```

Etapa 6: Carregar o arquivo manifesto para um bucket do Amazon S3

Carregue o arquivo manifesto para um bucket do Amazon S3. Se o bucket do Amazon S3 não residir na mesma região da AWS que seu cluster do Amazon Redshift, você deve usar a opção REGION para especificar a região da AWS na qual o manifesto está localizado. Para obter informações sobre como criar um bucket do Amazon S3 e carregar um arquivo, consulte Guia do usuário do Amazon Simple Storage Service.

Etapa 7: Executar o comando COPY para carregar os dados

Execute um comando COPY para conectar-se ao host e carregar os dados em uma tabela do Amazon Redshift. No comando COPY, especifique o caminho do objeto Amazon S3 explícito para o arquivo manifesto e inclua a opção SSH. Por exemplo,

```
COPY sales
FROM 's3://amzn-s3-demo-bucket/ssh_manifest'
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
DELIMITER '|'
SSH;
```

> **Note**
>
> Se você usar a compactação automática, o comando COPY realizará duas leituras de dados, ou seja, executará o comando remoto duas vezes. A primeira leitura é para fornecer um exemplo para a análise de compactação e a segunda leitura efetivamente carrega os dados. Se a execução do comando remoto duas vezes causar um problema devido a possíveis efeitos colaterais, você deverá desativar a compactação automática. Para desativar a compactação automática, execute o comando COPY com a opção COMPUPDATE definida como desativado. Para obter mais informações, consulte Carregamento de tabelas com compactação automática.

Carregar dados de uma tabela do Amazon DynamoDB

Você pode usar o comando COPY para carregar uma tabela com dados de uma única tabela do Amazon DynamoDB.

> **Important**
>
> A tabela do Amazon DynamoDB que fornece os dados deve ser criada na mesma região da AWS de seu cluster, a menos que você use a opção REGION para especificar a região da AWS na qual a tabela do Amazon DynamoDB está localizada.

O comando COPY utiliza a arquitetura de processamento maciçamente paralelo (MPP) do Amazon Redshift para ler e carregar dados em paralelo de uma tabela do Amazon DynamoDB. Você pode tirar o máximo proveito do processamento paralelo definindo estilos de distribuição em suas tabelas do Amazon Redshift. Para obter mais informações, consulte Distribuição de dados para otimização de consultas.

> **Important**
>
> Quando o comando COPY lê dados da tabela Amazon DynamoDB, o throughput resultante faz parte do throughput provisionado dessa tabela.

Para evitar o consumo de quantidades excessivas de throughput de leitura provisionada, recomendamos que você não carregue dados de tabelas do Amazon DynamoDB que estão em ambientes de produção. Se você carregar dados de tabelas de produção, recomendamos que você defina a opção READRATIO muito mais baixa que a porcentagem média do throughput provisionado não utilizado. Uma baixa configuração de READRATIO ajudará a minimizar problemas de limitação. Para usar todo o throughput provisionado de uma tabela do Amazon DynamoDB, defina READRATIO como 100.

O comando COPY combina os nomes dos atributos nos itens recuperados da tabela DynamoDB com os nomes das colunas em uma tabela existente do Amazon Redshift usando as seguintes regras:

- As colunas da tabela do Amazon Redshift não fazem distinção entre maiúsculas e minúsculas com os atributos de item do Amazon DynamoDB. Se um item em uma tabela do DynamoDB contém vários atributos que diferem somente na capitalização, tal como Price e PRICE, o comando COPY falhará.
- As colunas da tabela Amazon Redshift que não correspondem a um atributo na tabela Amazon DynamoDB são carregadas como NULL ou vazias, dependendo do valor especificado com a opção EMPTYASNULL no comando COPY.

- Os atributos do Amazon DynamoDB que não correspondem a uma coluna na tabela do Amazon Redshift são descartados. Os atributos são lidos antes de serem combinados, portanto mesmo os atributos descartados consomem parte do throughput provisionado da tabela.
- Apenas atributos do Amazon DynamoDB com tipos de dados escalares STRING e NUMBER são aceitos. Os tipos de dados Amazon DynamoDB BINARY e SET não são aceitos. Se um comando COPY tentar carregar um atributo com um tipo de dados não compatível, o comando falhará. Se o atributo não corresponder a uma coluna da tabela Amazon Redshift, COPY não tentará carregá-lo e não gerará um erro.

O comando COPY usa a seguinte sintaxe para carregar dados de uma tabela do Amazon DynamoDB:

```
COPY <redshift_tablename> FROM 'dynamodb://<dynamodb_table_name>'
authorization
readratio '<integer>';
```

Os valores para autorização são as credenciais da AWS necessárias para acessar a tabela do Amazon DynamoDB. Se essas credenciais corresponderem a um usuário, ele deverá ter permissão para executar os comandos SCAN e DESCRIBE na tabela do Amazon DynamoDB que está sendo carregada.

Os valores para autorização fornecem a autorização da AWS que seu cluster necessita para acessar a tabela do Amazon DynamoDB. A permissão deve incluir SCAN e DESCRIBE para a tabela do Amazon DynamoDB que está sendo carregada. Para obter mais informações sobre as permissões necessárias, consulte Permissões do IAM para COPY, UNLOAD e CREATE LIBRARY. O método preferido de autenticação é especificar o parâmetro IAM_ROLE e fornecer o nome de recurso da Amazon (ARN) para uma função do IAM com as permissões necessárias. Para obter mais informações, consulte Controle de acesso com base em função.

Para autenticação usando o parâmetro IAM_ROLE, *<aws-account-id>* e *<role-name>* conforme exibido na sintaxe a seguir.

```
IAM_ROLE 'arn:aws:iam::<aws-account-id>:role/<role-name>'
```

O seguinte exemplo mostra a autenticação usando uma função do IAM.

```
COPY favoritemovies
```

```
FROM 'dynamodb://ProductCatalog'
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Para obter mais informações sobre outras opções de autorização, consulte Parâmetros de autorização

Se você quiser validar seus dados sem realmente carregar a tabela, use a opção NOLOAD com o comando COPY.

O exemplo a seguir carrega a tabela FAVORITEMOVIES com dados da tabela my-favorite-movies-table do DynamoDB. A atividade de leitura pode consumir até 50% do throughput provisionado.

```
COPY favoritemovies FROM 'dynamodb://my-favorite-movies-table'
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
READRATIO 50;
```

Para maximizar o throughput, o comando COPY carrega dados de uma tabela do Amazon DynamoDB em paralelo entre os nós de computação no cluster.

Throughput provisionado com compactação automática

Por padrão, o comando COPY aplica a compactação automática sempre que você especifica uma tabela de destino vazia sem codificação de compactação. A análise de compressão automática inicialmente obtém amostras de um grande número de linhas da tabela Amazon DynamoDB. O tamanho da amostra é baseado no valor do parâmetro COMPROWS. O padrão é 100.000 linhas por fatia.

Após o teste, as linhas da amostra são descartadas e a tabela inteira é carregada. Como resultado, muitas linhas são lidas duas vezes. Para obter mais informações sobre como funciona a compactação automática, consulte Carregamento de tabelas com compactação automática.

> **Important**
>
> Quando o comando COPY lê dados da tabela Amazon DynamoDB, incluindo as linhas usadas para amostragem, o throughput de dados resultante faz parte do throughput provisionado dessa tabela.

Carregar dados multibyte do Amazon DynamoDB

Se seus dados incluem caracteres multibyte não ASCII (tal como caracteres chineses ou cirílicos), você deve carregar os dados em colunas VARCHAR. O tipo de dados VARCHAR é compatível com caracteres UTF-8 de quatro bytes, mas o tipo de dados CHAR aceita apenas caracteres ASCII de único byte. Você não pode carregar caracteres de cinco bytes ou mais nas tabelas do Amazon Redshift. Para obter mais informações sobre CHAR e VARCHAR, consulte Tipos de dados.

Como verificar se os dados foram carregados corretamente

Após a conclusão da operação de carregamento, consulte a tabela de sistema STL_LOAD_COMMITS para certificar-se de que os arquivos esperados foram carregados. Execute o comando COPY e a verificação do carregamento na mesma transação, para que seja possível reverter toda a transação se houver um problema com o carregamento.

A consulta a seguir retorna as entradas para um carregamento de tabelas no banco de dados TICKIT:

```
SELECT query, trim(filename) AS filename, curtime, status
FROM stl_load_commits
WHERE filename like '%tickit%' order by query;


 query |         filename          |          curtime           | status
-------+---------------------------+----------------------------+--------
 22475 | tickit/allusers_pipe.txt  | 2013-02-08 20:58:23.274186 |      1
 22478 | tickit/venue_pipe.txt     | 2013-02-08 20:58:25.070604 |      1
 22480 | tickit/category_pipe.txt  | 2013-02-08 20:58:27.333472 |      1
 22482 | tickit/date2008_pipe.txt  | 2013-02-08 20:58:28.608305 |      1
 22485 | tickit/allevents_pipe.txt | 2013-02-08 20:58:29.99489  |      1
 22487 | tickit/listings_pipe.txt  | 2013-02-08 20:58:37.632939 |      1
 22489 | tickit/sales_tab.txt      | 2013-02-08 20:58:37.632939 |      1
(6 rows)
```

Validação de dados de entrada

Para validar os dados nos arquivos de entrada do Amazon S3 ou na tabela do Amazon DynamoDB antes de realmente carregar os dados, use a opção NOLOAD com o comando COPY. Use NOLOAD com os mesmos comandos e opções de COPY que você usaria para carregar os dados. NOLOAD verifica a integridade de todos os dados sem carregá-los no banco de dados. A opção NOLOAD exibirá os erros que ocorrerão se você tentar carregar os dados.

Por exemplo, se você especificou o caminho incorreto do Amazon S3 para o arquivo de entrada, o Amazon Redshift exibirá o erro a seguir.

```
ERROR:  No such file or directory
DETAIL:
-----------------------------------------------
Amazon Redshift error:  The specified key does not exist
code:      2
context:   S3 key being read :
location:  step_scan.cpp:1883
process:   xenmaster [pid=22199]
-----------------------------------------------
```

Para solucionar mensagens de erro, consulte Referência de erros de carregamento.

Para ver um exemplo usando a opção NOLOAD, consulte Comando COPY com a opção NOLOAD.

Carregamento de tabelas com compactação automática

Você pode aplicar codificações de compactação a colunas em tabelas manualmente com base em sua própria avaliação dos dados. Ou você pode usar o comando COPY com COMPUPDATE definido como ON para analisar e aplicar compactação automaticamente com base nos dados de amostra.

Você pode usar a compactação automática ao criar e carregar uma nova tabela. O comando COPY executa uma análise de compactação. Você também pode executar uma análise de compactação sem carregar dados ou alterar a compactação em uma tabela executando o comando ANALYZE COMPRESSION em uma tabela já preenchida. Por exemplo, você pode executar ANALYZE COMPRESSION quando quiser analisar a compactação em uma tabela para uso futuro, preservando as instruções de linguagem de definição de dados (DDL) existentes.

A compactação automática equilibra a performance geral ao escolher codificações de compactação. As varreduras restritas por intervalo podem apresentar má performance se as colunas de chaves de classificação forem mais altamente compactadas do que outras colunas na mesma consulta. Como resultado, a compactação automática ignora a fase de análise de dados nas colunas de chave de classificação e mantém os tipos de codificação definidos pelo usuário.

A compactação automática escolherá a codificação RAW se você não tiver definido explicitamente um tipo de codificação. ANALYZE COMPRESSION se comporta da mesma forma. Para obter a performance ideal de consultas, considere o uso de RAW para chaves de classificação.

Como a compactação automática funciona

Quando o parâmetro COMPUPDATE é definido como ON, o comando COPY aplica a compactação automática sempre que você executa o comando COPY com uma tabela de destino vazia e todas as colunas da tabela têm a codificação RAW ou nenhuma codificação.

Para aplicar a compactação automática em uma tabela vazia, independentemente de suas codificações atuais de compactação, execute o comando COPY com a opção COMPUPDATE definida como ON. Para desativar a compactação automática, execute o comando COPY com a opção COMPUPDATE definida como desativado.

Você não pode aplicar a compactação automática em uma tabela que já contém dados.

> **Note**
>
> A análise de compactação automática requer linhas suficientes nos dados de carregamento (pelo menos 100.000 linhas por fatia) para gerar uma amostra significativa.

A compactação automática executa essas operações em segundo plano como parte da transação de carregamento:

1. Uma amostra inicial das linhas do arquivo de entrada é carregada. O tamanho da amostra é baseado no valor do parâmetro COMPROWS. O padrão é 100,000.
2. As opções de compactação são escolhidas para cada coluna.
3. As linhas da amostra são removidas da tabela.
4. A tabela é recriada com as codificações de compactação escolhidas.
5. O arquivo de entrada é carregado e compactado usando as novas codificações.

Quando você executa o comando COPY, a tabela é totalmente carregada, compactada e está pronta para uso. Se você carregar mais dados depois, as linhas adicionadas são compactadas de acordo com a codificação existente.

Se você deseja executar somente uma análise de compactação, execute ANALYZE COMPRESSION que é mais eficiente que a execução de um COPY completo. Então, você pode avaliar os resultados para decidir se deve usar a compactação automática ou recriar a tabela manualmente.

A compactação automática é compatível somente com o comando COPY. Como alternativa, você pode aplicar a codificação de compactação manualmente ao criar a tabela. Para obter informações sobre a codificação de compactação manual, consulte Compactação de colunas para reduzir o tamanho dos dados armazenados.

Exemplo de compactação automática

Neste exemplo, suponha que o banco de dados TICKIT contenha uma cópia da tabela LISTING chamada BIGLIST e que você queira aplicar a compactação automática nessa tabela quando ela estiver carregada com aproximadamente 3 milhões de linhas.

Para carregar e compactar automaticamente a tabela

1. A tabela deve estar vazia. Você pode aplicar a compactação automática somente em uma tabela vazia:

   ```
   TRUNCATE biglist;
   ```

2. Carregue a tabela com um único comando COPY. Embora a tabela esteja vazia, alguma codificação anterior pode ter sido especificada. Para garantir que o Amazon Redshift execute uma análise de compactação, defina o parâmetro COMPUPDATE como ativado.

   ```
   COPY biglist FROM 's3://amzn-s3-demo-bucket/biglist.txt'
   IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
   DELIMITER '|' COMPUPDATE ON;
   ```

   Como nenhuma opção COMPROWS está especificada, o tamanho padrão e recomendado de amostra de 100.000 linhas por fatia é usado.

3. Observe o novo esquema para a tabela BIGLIST para revisar os esquemas de codificação escolhidos automaticamente.

   ```
   SELECT "column", type, encoding
   from pg_table_def where tablename = 'biglist';


        Column     |            Type             | Encoding
   ----------------+-----------------------------+----------
    listid         | integer                     | az64
    sellerid       | integer                     | az64
    eventid        | integer                     | az64
   ```

```
 dateid         | smallint                    | none
 numtickets     | smallint                    | az64
 priceperticket | numeric(8,2)                | az64
 totalprice     | numeric(8,2)                | az64
 listtime       | timestamp without time zone | az64
```

4. Verifique que o número esperado de linhas foi carregado:

```
select count(*) from biglist;

count
---------
3079952
(1 row)
```

Quando linhas forem adicionadas posteriormente a essa tabela usando instruções COPY ou INSERT, as mesmas codificações de compactação serão aplicadas.

Otimização de armazenamento para tabelas estreitas

Se você tiver uma tabela com muito poucas colunas, mas um número muito grande de linhas, as três colunas de identidade de metadados ocultas (INSERT_XID, DELETE_XID, ROW_ID) consumirão uma quantidade desproporcional de espaço em disco para a tabela.

Para otimizar a compactação das colunas ocultas, carregue a tabela em uma única transação COPY sempre que possível. Se você carregar a tabela com vários comandos COPY distintos, a coluna INSERT_XID não será bem compactada. Você deverá realizar uma operação de limpeza se usar vários comandos COPY, mas isso não melhorará a compactação de INSERT_XID.

Carregamento de valores padrão de coluna

Opcionalmente, é possível definir uma lista de colunas em seu comando COPY. Se uma coluna na tabela for omitida da lista de colunas, COPY carregará a coluna com o valor fornecido pela opção DEFAULT especificada no comando CREATE TABLE ou com NULL, se a opção DEFAULT não tiver sido especificada.

Se COPY tentar atribuir NULL a uma coluna definida como NOT NULL, o comando COPY falhará. Para informações sobre como atribuir a opção DEFAULT, consulte CRIAR TABELA.

Ao carregar de arquivos de dados no Amazon S3, as colunas na lista de colunas devem estar na mesma ordem que os campos no arquivo de dados. Se um campo no arquivo de dados não tem uma coluna correspondente na lista de coluna, o comando COPY falha.

Ao carregar da tabela do Amazon DynamoDB, a ordem não importa. Todos os campos nos atributos do Amazon DynamoDB que não correspondem a uma coluna na tabela do Amazon Redshift são descartados.

As seguintes restrições se aplicam ao usar o comando COPY para carregar valores DEFAULT em uma tabela:

- Se uma coluna IDENTITY é incluída na lista de colunas, a opção EXPLICIT_IDS também deve ser especificada no comando COPY ou o comando COPY falha. Da mesma forma, se uma coluna IDENTITY é omitida da lista de colunas e a opção EXPLICIT_IDS é especificada, a operação COPY falha.
- Como a expressão DEFAULT avaliada para determinada coluna é a mesma para todas as linhas carregadas, uma expressão DEFAULT que usa uma função RANDOM() atribuirá o mesmo valor para todas as linhas.
- As expressões PADRÃO contendo CURRENT_DATE ou SYSDATE são definidas com o timestamp da transação atual.

Para obter um exemplo, consulte “Carregar dados de um arquivo com valores padrão” em Exemplos de COPY.

Solução de problemas de carregamento de dados

Ao carregar dados em tabelas do Amazon Redshift, você pode se deparar com erros do Amazon S3, dados de entrada inválidos e erros do comando COPY. As seções a seguir fornecem informações sobre como identificar e solucionar erros de carregamentos de dados.

Tópicos

- Solução de problemas de integração de eventos do S3 e erros de COPY JOB
- Erros S3ServiceException
- Tabelas de sistema para solucionar problemas de carregamento de dados
- Erros no carregamento de caracteres multibyte
- Referência de erros de carregamento

Solução de problemas de integração de eventos do S3 e erros de COPY JOB

Use as informações a seguir para solucionar problemas comuns com integrações de eventos do Amazon S3 e COPY JOB com o Amazon Redshift.

Falha na criação da integração de eventos do S3

Se a criação da integração de eventos do S3 falhar, o status da integração será `Inactive`. Verifique se os seguintes itens estão corretos para o data warehouse do Amazon Redshift:

- Você adicionou a entidade principal autorizada e a origem de integração correta ao namespace de destino no Amazon Redshift. Consulte Pré-requisitos para criar uma integração de eventos do S3.
- Você adicionou a política baseada em recursos correta ao bucket de origem do Amazon S3. Consulte Pré-requisitos para criar uma integração de eventos do S3.

Os dados do Amazon S3 não estão aparecendo no banco de dados de destino

Se os dados de um COPY JOB não aparecerem, verifique o seguinte:

- Consulte SYS_COPY_JOB_DETAIL para ver se o arquivo do Amazon S3 foi carregado, se a ingestão está pendente ou se há um erro. Para obter mais informações, consulte SYS_COPY_JOB_DETAIL.
- Consulte STL_ERROR ou SYS_COPY_JOB_INFO se o arquivo do Amazon S3 não estiver lá ou se houver um tempo de espera inesperado. Procure por erros de credenciais ou um detalhe que sugira que a integração está inativa. Para obter mais informações, consulte STL_ERROR e SYS_COPY_JOB_INFO.

Erros S3ServiceException

Os erros mais comuns de s3ServiceException são causados por uma string de credenciais incorreta ou formatada incorretamente, tendo seu cluster e seu bucket em diferentes regiões da AWS e permissões insuficientes do Amazon S3.

A seção fornece informações para a solução de problemas de cada tipo de erro.

String de credenciais inválida

Se sua string de credenciais foi formatada inadequadamente, você receberá a seguinte mensagem de erro:

```
ERROR: Invalid credentials. Must be of the format: credentials
'aws_access_key_id=<access-key-id>;aws_secret_access_key=<secret-access-key>
[;token=<temporary-session-token>]'
```

Verifique se a sequência de credenciais não contém espaços ou quebras de linha e está entre aspas simples.

ID de chave de acesso inválido

Se seu ID de chave de acesso não existir, você receberá a seguinte mensagem de erro:

```
[Amazon](500310) Invalid operation: S3ServiceException:The AWS Access Key Id you
 provided does not exist in our records.
```

Muitas vezes, trata-se de um erro de copiar e colar. Certifique-se de que o ID de chave de acesso foi inserido corretamente. Além disso, se você está usando chaves de sessão temporárias, verifique se o valor de `token` foi definido.

Chave de acesso secreta inválida

Se sua chave de acesso secreta estiver incorreta, você receberá a seguinte mensagem de erro:

```
[Amazon](500310) Invalid operation: S3ServiceException:The request signature we
 calculated does not match the signature you provided.
Check your key and signing method.,Status 403,Error SignatureDoesNotMatch
```

Muitas vezes, trata-se de um erro de copiar e colar. Certifique-se de que a chave de acesso secreta foi inserida corretamente e que ela é a chave correta para o ID de chave de acesso.

O bucket está em uma região diferente

O bucket do Amazon S3 especificado no comando COPY deve estar na mesma região do cluster AWS. Se o seu bucket do Amazon S3 e seu cluster estiverem em regiões diferentes, você receberá um erro semelhante ao seguinte:

```
ERROR: S3ServiceException:The bucket you are attempting to access must be addressed
 using the specified endpoint.
```

Você pode criar um bucket do Amazon S3 em uma região específica selecionando a região ao criar o bucket usando o Console de Gerenciamento do Amazon S3 ou especificando um endpoint ao

criar o bucket usando a API ou CLI do Amazon S3. Para obter mais informações, consulte Carregar arquivos no Amazon S3 para usar com COPY.

Consulte mais informações sobre as regiões do Amazon S3 em Acessar um bucket no Guia do usuário do Amazon Simple Storage Service.

Você também pode especificar a região usando a opção REGION com o comando COPY.

Acesso negado

Se o usuário não tiver permissões suficientes, você receberá a seguinte mensagem de erro:

```
ERROR: S3ServiceException:Access Denied,Status 403,Error AccessDenied
```

Uma possível causa é o usuário identificado pelas credenciais não ter acesso de LIST e GET ao bucket do Amazon S3. Para outras causas, consulte Solucionar erros de Acesso Negado (403 Proibido) no Amazon S3 no Guia do usuário do Amazon Simple Storage Service.

Para obter informações sobre o gerenciamento do acesso de usuários aos buckets, consulte Gerenciamento de identidade e acesso no Amazon S3 no Guia do usuário do Amazon Simple Storage Service.

Tabelas de sistema para solucionar problemas de carregamento de dados

As seguintes tabelas de sistema do Amazon Redshift podem ser úteis na solução de problemas de carregamento de dados:

- Consulte STL_LOAD_ERRORS para descobrir erros que ocorreram durante os carregamentos específicos.
- Consulte STL_FILE_SCAN para visualizar o tempo de carregamento de arquivos específicos ou para ver se um arquivo específico foi mesmo lido.
- Consulte STL_S3CLIENT_ERROR para encontrar detalhes de erros encontrados durante a transferência de dados do Amazon S3.

Para encontrar e diagnosticar erros de carregamento

1. Crie uma exibição ou defina uma consulta que retorne detalhes dos erros de carregamento. O seguinte exemplo junta a tabela STL_LOAD_ERRORS à tabela STV_TBL_PERM a IDs para comparar os IDs de tabela aos nomes reais da tabela.

```
create view loadview as
(select distinct tbl, trim(name) as table_name, query, starttime,
trim(filename) as input, line_number, colname, err_code,
trim(err_reason) as reason
from stl_load_errors sl, stv_tbl_perm sp
where sl.tbl = sp.id);
```

2. Defina a opção MAXERRORS em seu comando COPY como um valor grande o suficiente para permitir que COPY retorne informações úteis sobre seus dados. Se COPY encontrar erros, uma mensagem de erro o direciona a consultar a tabela STL_LOAD_ERRORS para obter detalhes.
3. Consulte a exibição LOADVIEW para visualizar detalhes de erros. Por exemplo:

```
select * from loadview where table_name='venue';
```

```
  tbl   | table_name | query |         starttime
--------+------------+-------+----------------------------
 100551 | venue      | 20974 | 2013-01-29 19:05:58.365391

|     input      | line_number | colname | err_code |       reason
+----------------+-------------+---------+----------+--------------------
| venue_pipe.txt |           1 |       0 |     1214 | Delimiter not found
```

4. Corrija o problema no arquivo de entrada ou no script do carregamento com base nas informações que a exibição retorna. Alguns erros típicos de carregamento a observar incluem:
   - Divergência entre os tipos de dados na tabela e valores nos campos nos dados de entrada.
   - Divergência entre o número de colunas na tabela e o número de campos nos dados de entrada.
   - Aspas incompatíveis. O Amazon Redshift oferece suporte a aspas simples e duplas; no entanto, essas aspas devem ser balanceadas adequadamente.
   - Formato incorreto de dados de data/hora nos arquivos de entrada.
   - Valores fora do intervalo nos arquivos de entrada (para colunas numéricas).
   - O número de valores distintos de uma coluna excede a limitação de sua codificação de compactação.

Erros no carregamento de caracteres multibyte

As colunas com um tipo de dados CHAR aceitam somente caracteres UTF-8 de único byte, com valor de byte de até 127, ou 7F hex, que também é o conjunto de caracteres ASCII. Colunas VARCHAR aceitam caracteres UTF-8 multibyte até o máximo de quatro bytes. Para obter mais informações, consulte Tipos de caracteres.

Se uma linha em seus dados de carregamento contiver um caractere que não for válido para o tipo de dados da coluna, COPY retornará um erro e registrará uma linha na tabela de log de sistema STL_LOAD_ERRORS com o erro número 1220. O campo ERR_REASON inclui a sequência de bytes, em hex, para o caractere inválido.

Uma alternativa para corrigir caracteres não válidos em seus dados de carregamento é substituir os caracteres não válidos durante o processo de carregamento. Para substituir os caracteres UTF-8 não válidos, especifique a opção ACCEPTINVCHARS com o comando COPY. Se a opção ACCEPTINVCHARS estiver definida, o caractere especificado substituirá o ponto de código. Se a opção ACCEPTINVCHARS não estiver definida, o Amazon Redshift aceitará os caracteres como UTF-8 válidos. Para obter mais informações, consulte ACCEPTINVCHARS.

A lista de pontos de código a seguir são caracteres UTF-8 válidos. As operações COPY não retornarão um erro se a opção ACCEPTINVCHARS não estiver definida. No entanto, esses pontos de código são caracteres inválidos. Use opção ACCEPTINVCHARS para substituir o ponto de código por um caractere especificado por você. Esses pontos de código incluem o intervalo de valores de `0xFDD0` a `0xFDEF` e valores até `0x10FFFF`, terminados em `FFFE` ou `FFFF`:

- `0xFFFE`, `0x1FFFE`, `0x2FFFE`, ..., `0xFFFFE`, `0x10FFFE`
- `0xFFFF`, `0x1FFFF`, `0x2FFFF`, ..., `0xFFFFF`, `0x10FFFF`

O exemplo a seguir mostra o motivo do erro quando COPY tenta carregar o caractere UTF-8 `e0 a1 c7a4` em uma coluna CHAR:

```
Multibyte character not supported for CHAR
(Hint: Try using  VARCHAR). Invalid char: e0 a1 c7a4
```

Se o erro for relacionado a um tipo de dados VARCHAR, o motivo do erro incluirá um código de erro e a sequência hexadecimal UTF-8 inválida. O exemplo a seguir mostra o motivo do erro quando COPY tenta carregar UTF-8 `a4` em um campo VARCHAR:

```
String contains invalid or unsupported UTF-8 codepoints.
```

```
Bad UTF-8 hex sequence: a4 (error 3)
```

A tabela a seguir lista as descrições e ações alternativas sugeridas para erros de carregamento VARCHAR. Se um desses erros ocorrer, substitua o caractere com uma sequência de código UTF-8 válida ou remova o caractere.

Código de erro	Descrição
1	A sequência de bytes UTF-8 excede o máximo de quatro bytes compatível com VARCHAR.
2	A sequência de bytes UTF-8 está incompleta. COPY não encontrou o número esperado de bytes de continuação para um caractere multibyte antes do término da string.
3	O caractere UTF-8 de único byte está fora do intervalo. O byte inicial não deve ser 254, 255 ou qualquer caractere entre 128 e 191 (inclusive).
4	O valor do byte final na sequência de bytes está fora do intervalo. O byte de continuação deve ser entre 128 e 191 (inclusive).
5	O caractere UTF-8 é reservado como um substituto. Os pontos de código substituto (U+D800 a U+DFFF) não são válidos.
8	A sequência de bytes excede o ponto de código UTF-8 máximo.
9	A sequência de bytes UTF-8 não tem um ponto de código correspondente.

Referência de erros de carregamento

Se um erro ocorrer durante o carregamento de dados de um arquivo, consulte a tabela STL_LOAD_ERRORS para identificar o erro e determinar a possível explicação. A tabela a seguir lista todos os códigos de erros que podem ocorrer durante carregamentos de dados:

Códigos de erro de carregamento

Código de erro	Descrição
1200	Erro de análise desconhecido. Entre em contato com o suporte.
1201	O delimitador do campo não foi localizado no arquivo de entrada.
1202	Os dados de entrada tinham mais colunas do que definido no DDL.
1203	Os dados de entrada tinham menos colunas do que definido no DDL.
1204	Os dados de entrada excederam o intervalo aceitável para o tipo de dados.
1205	O formato de data é inválido. Consulte Strings DATEFORMAT e TIMEFORMAT para os formatos válidos.
1206	O formato de carimbo de data/hora é inválido. Consulte Strings DATEFORMAT e TIMEFORMAT para os formatos válidos.
1207	Os dados continham um valor fora do intervalo esperado de 0-9.
1208	Erro de formato do tipo de dados FLOAT.
1209	Erro de formato do tipo de dados DECIMAL.
1210	Erro de formato do tipo de dados BOOLEAN.
1211	A linha de entrada não continha dados.
1212	O arquivo de carregamento não foi encontrado.
1213	Um campo especificado como NOT NULL não continha dados.
1214	Delimitador não encontrado.
1215	Erro do campo CHAR.
1.216	A linha de entrada não é válida.

Código de erro	Descrição
1217	O valor da coluna de identidade não é válido.
1218	Ao usar NULL AS '\0', um campo contendo um terminador nulo (NUL ou UTF-8 0000) continha mais que um byte.
1219	O hexadecimal UTF-8 contém um dígito inválido.
1220	A string contém pontos de código UTF-8 inválidos ou incompatíveis.
1221	A codificação do arquivo não é a mesma que a especificada no comando COPY.
1222	Erro de estouro de valor inteiro.
1223	O tipo de dado não é válido.
1224	Os dados de entrada não estão bem formados no formato JSON para o tipo de dados super.
8001	O parâmetro COPY com MANIFEST requer caminho completo de um objeto do Amazon S3.
9005	Chave final inválida especificada.

Criar uma integração de eventos do S3 para copiar automaticamente arquivos de buckets do Amazon S3

Você pode usar uma tarefa de cópia automática para carregar dados em suas tabelas do Amazon Redshift usando arquivos armazenados no Amazon S3. O Amazon Redshift detecta quando novos arquivos do Amazon S3 são adicionados ao caminho especificado em seu comando COPY. Depois, um comando COPY é executado automaticamente sem que você precise criar um pipeline externo de ingestão de dados. O Amazon Redshift mantém o controle de quais arquivos foram carregados. O Amazon Redshift determina o número de arquivos agrupados por comando COPY. Você pode ver os comandos COPY resultantes nas visualizações do sistema.

O primeiro passo para criar um COPY JOB automático é criar uma integração de eventos do S3. Quando um novo arquivo aparece no bucket de origem do Amazon S3, o Amazon Redshift gerencia o carregamento dos arquivos no banco de dados usando o comando COPY.

Pré-requisitos para criar uma integração de eventos do S3

Para configurar a integração de eventos do S3, confirme se os pré-requisitos a seguir foram atendidos.

- O bucket do Amazon S3 deve ter uma política de bucket que conceda diversas permissões do Amazon S3. Por exemplo, a política de exemplo a seguir concede permissões para o bucket de recursos `amzn-s3-demo-bucket` que está hospedado em *`us-east-1`*. Tanto o bucket do Amazon S3 quanto a integração estão na mesma Região da AWS.

  JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Auto-Copy-Policy-01",
            "Effect": "Allow",
            "Principal": {
                "Service": "redshift.amazonaws.com"
                },
            "Action": [
                "s3:GetBucketNotification",
                "s3:PutBucketNotification",
                "s3:GetBucketLocation"
            ],
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket:*",
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:redshift:us-
east-1:111122223333:integration:*"
                },
                "StringEquals": {
                    "aws:SourceAccount": "111122223333"
                }
            }
        }
    ]
```

```
}
```

- O cluster provisionado do Amazon Redshift de destino ou o namespace do Redshift sem servidor deve ter permissão para o bucket. Confirme se o perfil do IAM associado ao cluster ou namespace sem servidor tem uma política do IAM que concede as permissões adequadas. A política deve permitir `s3:GetObject` para um recurso de bucket, como *amzn-s3-demo-bucket*, e `s3:ListBucket` para um recurso de bucket e o respectivo conteúdo, como *amzn-s3-demo-bucket*`/*`.

  JSON

  ```
  {
      "Version":"2012-10-17",
      "Statement": [
          {
              "Sid": "AutoCopyReadId",
              "Effect": "Allow",
              "Action": [
                  "s3:GetObject",
                  "s3:ListBucket"
              ],
              "Resource": [
                  "arn:aws:s3:::amzn-s3-demo-bucket",
                  "arn:aws:s3:::amzn-s3-demo-bucket/*"
              ]
          }
      ]
  }
  ```

  Adicione a política a um perfil do IAM que tenha uma relação de confiança para o perfil da seguinte maneira:

  JSON

  ```
  {
      "Version":"2012-10-17",
      "Statement": [
          {
              "Effect": "Allow",
              "Principal": {
                  "Service": [
                      "redshift.amazonaws.com"
  ```

```
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Se o data warehouse de destino for um cluster provisionado, você poderá associar um perfil do IAM ao cluster provisionado usando o console do Amazon Redshift na guia Permissões do cluster nos detalhes do cluster. Para obter informações sobre como associar um perfil ao cluster provisionado, consulte Associar funções do IAM a clusters no Guia de gerenciamento do Amazon Redshift.

Se o data warehouse de destino for o Redshift sem servidor, você poderá associar o perfil do IAM ao namespace sem servidor usando o console do Redshift sem servidor, na guia Segurança e criptografia nos detalhes do namespace. Para obter informações sobre como associar um perfil ao namespace sem servidor, consulte Conceder as permissões necessárias para o Amazon Redshift Serverless no Guia de gerenciamento do Amazon Redshift.

- O data warehouse do Amazon Redshift também deve ter uma política de recursos que permita o bucket do Amazon S3. Se você usar o console do Amazon Redshift, ao criar a integração de eventos do S3, o Amazon Redshift fornecerá a opção Corrigir para mim para adicionar essa política ao data warehouse do Amazon Redshift. Para atualizar uma política de recursos, você pode usar o comando put-resource-policy da AWS CLI. Por exemplo, para anexar uma política de recursos ao cluster provisionado do Amazon Redshift e realizar uma integração de eventos do S3 com um bucket do Amazon S3, execute um comando da AWS CLI semelhante ao apresentado abaixo. O exemplo a seguir mostra uma política para um namespace de cluster provisionado na Região da AWS *us-east-1* referente à conta de usuário *123456789012*. O nome do bucket é *amzn-s3-demo-bucket*.

```
aws redshift put-resource-policy \
--policy file://rs-rp.json \
--resource-arn "arn:aws:redshift: us-east-1:123456789012:namespace/cc4ffe56-
ad2c-4fd1-a5a2-f29124a56433"
```

Onde `rs-rp.json` contém:

JSON

```
{
```

```
 "Version":"2012-10-17",
 "Statement": [
  {
   "Effect": "Allow",
   "Principal": {
    "Service": "redshift.amazonaws.com"
   },
   "Action": "redshift:AuthorizeInboundIntegration",
   "Resource": "arn:aws:redshift:us-east-1:123456789012:namespace:cc4ffe56-
ad2c-4fd1-a5a2-f29124a56433",
   "Condition": {
    "StringEquals": {
     "aws:SourceArn": "arn:aws:s3:::amzn-s3-demo-bucket",
     "aws:SourceAccount": 111122223333
    }
   }
  },
  {
   "Effect": "Allow",
   "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/myRedshiftRole"
   },
   "Action": "redshift:CreateInboundIntegration",
   "Resource": "arn:aws:redshift:us-east-1:123456789012:namespace:cc4ffe56-
ad2c-4fd1-a5a2-f29124a56433",
   "Condition": {
    "StringEquals": {
     "aws:SourceArn": "arn:aws:s3:::amzn-s3-demo-bucket",
     "aws:SourceAccount": 111122223333
    }
   }
  }
 ]
}
```

Para anexar uma política de recursos ao namespace do Redshift sem servidor e realizar uma integração de eventos do S3 com um bucket do Amazon S3, execute um comando da AWS CLI semelhante ao apresentado abaixo. O exemplo a seguir mostra uma política para um namespace sem servidor na Região da AWS *us-east-1* referente à conta de usuário *123456789012*. O nome do bucket é *amzn-s3-demo-bucket*.

```
aws redshift put-resource-policy \
```

```
--policy file://rs-rp.json \
--resource-arn "arn:aws:redshift-serverless:us-east-1:123456789012:namespace/
namespace-1"
```

Onde `rs-rp.json` contém:

JSON

```
{
 "Version":"2012-10-17",
 "Statement": [
  {
   "Effect": "Allow",
   "Principal": {
    "Service": "redshift.amazonaws.com"
   },
   "Action": "redshift:AuthorizeInboundIntegration",
   "Resource": "arn:aws:redshift-serverless:us-east-1:123456789012:namespace/
namespace-1",
   "Condition": {
    "StringEquals": {
     "aws:SourceArn": "arn:aws:s3:::amzn-s3-demo-bucket",
     "aws:SourceAccount": 111122223333

    }
   }
  },
  {
   "Effect": "Allow",
   "Principal": {
    "AWS": "arn:aws:iam::123456789012:user/myUser"
   },
   "Action": "redshift:CreateInboundIntegration",
   "Resource": "arn:aws:redshift-serverless:us-east-1:123456789012:namespace/
namespace-1",
   "Condition": {
    "StringEquals": {
     "aws:SourceArn": "arn:aws:s3:::amzn-s3-demo-bucket",
     "aws:SourceAccount": 111122223333
    }
   }
  }
 ]
```

```
}
```

Criar uma integração de eventos do S3

Para configurar o trabalho de cópia, primeiro defina uma integração de eventos do S3.

Amazon Redshift console

Como criar uma integração de eventos do Amazon S3 no console do Amazon Redshift

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. No painel de navegação à esquerda, selecione Integração de eventos do S3.
3. Escolha Criar integração de eventos do Amazon S3 para abrir o assistente e criar uma integração de eventos do S3 para usar com cópia automática. O bucket de origem do Amazon S3 e o data warehouse de destino do Amazon Redshift devem estar na mesma Região da AWS. Especifique as seguintes informações ao executar as etapas para criar uma integração:

   - Nome da integração: é um identificador exclusivo entre todas as integrações de propriedade da Conta da AWS na Região da AWS atual.
   - Descrição: é um texto que descreve a integração de eventos do Amazon S3 para referência posterior.
   - Bucket do S3 de origem: o bucket do Amazon S3 está na Conta da AWS e Região da AWS atual, que é a origem dos dados de ingestão no Amazon Redshift.
   - Data warehouse do Amazon Redshift: é o cluster provisionado do Amazon Redshift de destino ou o grupo de trabalho do Redshift sem servidor que recebe os dados da integração.

     Se o Amazon Redshift de destino estiver na mesma conta, você poderá selecionar o destino. Se o destino estiver em uma conta diferente, especifique o ARN do data warehouse do Amazon Redshift. O destino deve ter uma política de recursos com entidades principais autorizadas e origem de integração. Se você não tiver as políticas de recursos corretas no destino e ele estiver na mesma conta, selecione a opção Corrigir isso para mim para aplicar automaticamente as políticas de recursos durante o processo de criação da integração. Se o destino estiver em uma Conta da AWS diferente, você precisará aplicar manualmente a política de recursos no warehouse do Amazon Redshift.

4. Insira até 50 Chaves de tag e com um valor opcional para fornecer metadados adicionais sobre a integração.
5. É exibida uma página de revisão na qual você pode selecionar Criar integração de eventos do S3.

AWS CLI

Para criar uma integração de eventos do Amazon S3 usando a AWS CLI, use o comando `create-integration` com as seguintes opções:

- `integration-name`: especifique um nome para a integração.
- `source-arn`: especifique o ARN do bucket de origem do Amazon S3.
- `target-arn`: especifique o ARN do namespace do cluster provisionado do Amazon Redshift ou do destino do grupo de trabalho do Redshift sem servidor.

O exemplo a seguir cria uma integração fornecendo o nome da integração, o ARN de origem e o ARN de destino. A integração não é criptografada.

```
aws redshift create-integration \
--integration-name s3-integration \
--source-arn arn:aws:s3:us-east-1::s3-example-bucket \
--target-arn arn:aws:redshift:us-east-1:123456789012:namespace:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222
          {
    "IntegrationArn": "arn:aws:redshift:us-east-1:123456789012:integration:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "IntegrationName": "s3-integration",
    "SourceArn": "arn:aws:s3:::s3-example-bucket",
    "SourceType": "s3-event-notifications",
    "TargetArn": "arn:aws:redshift:us-east-1:123456789012:namespace:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "Status": "creating",
    "Errors": [],
    "CreateTime": "2024-10-09T19:08:52.758000+00:00",
    "Tags": []
}
```

Você também pode usar os comandos da AWS CLI a seguir para gerenciar sua integração de eventos do S3.

- `delete-integration`: especifique um ARN de integração para excluir uma integração de eventos do S3.
- `modify-integration`: especifique um ARN de integração para alterar o nome ou a descrição (ou ambos) de uma integração de eventos do S3.
- `describe-integrations`: especifique um ARN de integração para visualizar as propriedades de uma integração de eventos do S3.

Consulte o Guia de CLI do Amazon Redshift para obter mais informações sobre esses comandos.

O Amazon Redshift então cria uma integração de eventos do S3 com a origem e o destino associados, o status e as informações sobre o status de um trabalho de cópia automática correspondente. Você pode visualizar informações sobre uma integração de eventos do S3 no console do Amazon Redshift escolhendo Integrações de eventos do S3 e selecionando a integração para mostrar os detalhes. As integrações são separadas entre aquelas criadas Na minha conta e De outras contas. A lista Na minha conta mostra integrações em que a origem e o destino estão na mesma conta. A lista De outras contas mostra integrações em que a origem pertence a outra conta.

Se você excluir uma integração de eventos do S3, o status correspondente do COPY JOB mudará de 1 (ativo) para `0` (inativo/pendente). Entretanto, o COPY JOB correspondente não é descartado automaticamente. Se mais tarde você tentar criar um COPY JOB com o mesmo nome, poderá haver um conflito.

Criar e monitorar um COPY JOB

Depois que a integração for criada, na página Detalhes da integração de eventos do S3 da integração que você criou, selecione Criar tarefa de cópia automática para acessar o Editor de Consultas V2 do Amazon Redshift, no qual você pode criar a tarefa de cópia automática para a integração. O Amazon Redshift compara o bucket na cláusula FROM na instrução COPY JOB CREATE com o bucket usado na integração de eventos do S3. Para obter mais informações sobre o Editor de Consultas V2 do Amazon Redshift, acesse Consultar um banco de dados usando o Editor de Consultas V2 no Guia de gerenciamento do Amazon Redshift. Por exemplo, execute o comando COPY a seguir no Editor de Consultas V2 para criar um COPY JOB automático que associe o bucket `s3://amzn-s3-demo-bucket/staging-folder` do Amazon S3 a uma integração de eventos do Amazon S3.

```
COPY public.target_table
FROM 's3://amzn-s3-demo-bucket/staging-folder'
```

```
IAM_ROLE 'arn:aws:iam::123456789012:role/MyLoadRoleName'
JOB CREATE my_copy_job_name
AUTO ON;
```

Você define um COPY JOB uma vez. Os mesmos parâmetros serão usados em execuções futuras.

Para definir e gerenciar um COPY JOB, você deve ter permissão. Para obter informações sobre como conceder e revogar permissão em um COPY JOB, consulte GRANT e REVOKE. Para obter mais informações sobre como conceder e revogar permissões com escopo definido para um COPY JOB, consulte Concessão de permissões em escopo e Revogação de permissões em escopo.

Você gerencia as operações de carregamento usando as opções CREATE, LIST, SHOW, DROP, ALTER e RUN para trabalhos. Para obter mais informações, consulte COPY JOB.

Você pode consultar as visualizações do sistema para ver o status e o progresso de COPY JOB. As visualizações são fornecidas da seguinte forma:

- SYS_COPY_JOB: contém uma linha para cada COPY JOB definido no momento.
- SYS_COPY_JOB_DETAIL: contém detalhes sobre arquivos pendentes, com erro e ingeridos para cada COPY JOB.
- SYS_COPY_JOB_INFO: contém mensagens registradas sobre um COPY JOB.
- SYS_LOAD_HISTORY: contém detalhes de comandos COPY.
- SYS_LOAD_ERROR_DETAIL: contém detalhes de erros de comandos COPY.
- SVV_COPY_JOB_INTEGRATIONS: contém detalhes das integrações de eventos do S3.
- STL_LOAD_ERRORS: contém erros de comandos COPY.
- STL_LOAD_COMMITS: contém informações usadas para solucionar problemas de carregamento de dados do comando COPY.

Para obter informações sobre como solucionar erros de integração de eventos do S3, consulte Solução de problemas de integração de eventos do S3 e erros de COPY JOB.

Para obter a lista de arquivos carregados por um COPY JOB, execute o seguinte SQL, mas primeiro substitua *<job_id>*:

```
SELECT job_id, job_name, data_source, copy_query, filename, status, curtime
FROM sys_copy_job copyjob
JOIN stl_load_commits loadcommit
```

```
ON copyjob.job_id = loadcommit.copy_job_id
WHERE job_id = <job_id>;
```

Considerações ao criar uma integração de eventos do S3 para cópia automática

Considere o seguinte ao usar a cópia automática:

- Você pode criar no máximo 200 COPY JOBS para cada cluster ou grupo de trabalho em uma Conta da AWS.
- Você pode criar no máximo 50 integrações de eventos do S3 para cada destino do Amazon Redshift.
- Não é possível criar uma integração de eventos do S3 com um bucket de origem do Amazon S3 que tenha um ponto (.) no nome do bucket.
- Só é possível criar uma integração de eventos do S3 entre a mesma origem e destino. Ou seja, só pode haver uma integração de eventos do S3 entre um bucket do Amazon S3 e um data warehouse do Amazon Redshift por vez.
- Você não pode ter nenhuma notificação de evento existente para o tipo de evento `S3_OBJECT_CREATED` que esteja definido no bucket de origem do Amazon S3. No entanto, após a criação de uma integração de eventos do S3, é possível atualizar a notificação de eventos do bucket do Amazon S3 usando um prefixo/sufixo com um escopo mais restrito. Dessa forma, você também pode configurar `S3_OBJECT_CREATED` para outro prefixo/sufixo de outros destinos e evitar um conflito com a integração de eventos do S3. Se você enfrentar problemas em que a cópia automática não estava sendo executada como esperado, prepare o log do AWS CloudTrail da ação `s3:PutBucketNotificationConfiguration` no bucket do S3 para o período em questão ao entrar em contato com o AWS Support.

Regiões com suporte

As regiões a seguir estão disponíveis para cópia automática.

Região	Cópia automática
África (Cidade do Cabo)	Available (Disponível)
Ásia-Pacífico (Hong Kong)	Available (Disponível)

Região	Cópia automática
Ásia-Pacífico (Taipei)	Available (Disponível)
Ásia-Pacífico (Tóquio)	Available (Disponível)
Ásia-Pacífico (Seul)	Available (Disponível)
Ásia-Pacífico (Osaka)	Available (Disponível)
Ásia-Pacífico (Mumbai)	Available (Disponível)
Ásia-Pacífico (Hyderabad)	Available (Disponível)
Ásia-Pacífico (Singapura)	Available (Disponível)
Ásia-Pacífico (Sydney)	Available (Disponível)
Ásia-Pacífico (Jacarta)	Available (Disponível)
Ásia-Pacífico (Melbourne)	Available (Disponível)
Ásia-Pacífico (Malásia)	Available (Disponível)
Ásia-Pacífico (Nova Zelândia)	Indisponível
Ásia-Pacífico (Tailândia)	Available (Disponível)
Canadá (Central)	Available (Disponível)
Oeste do Canadá (Calgary)	Available (Disponível)
China (Pequim)	Available (Disponível)
China (Ningxia)	Available (Disponível)
Europa (Frankfurt)	Available (Disponível)
Europa (Zurique)	Available (Disponível)
Europa (Estocolmo)	Available (Disponível)

Região	Cópia automática
Europa (Milão)	Available (Disponível)
Europa (Espanha)	Available (Disponível)
Europa (Irlanda)	Available (Disponível)
Europa (Londres)	Available (Disponível)
Europa (Paris)	Available (Disponível)
Israel (Tel Aviv)	Available (Disponível)
Oriente Médio (Emirados Árabes Unidos)	Available (Disponível)
Oriente Médio (Bahrein)	Available (Disponível)
México (Centro)	Available (Disponível)
América do Sul (São Paulo)	Available (Disponível)
Leste dos EUA (Norte da Virgínia)	Available (Disponível)
Leste dos EUA (Ohio)	Available (Disponível)
Oeste dos EUA (N. da Califórnia)	Available (Disponível)
Oeste dos EUA (Oregon)	Available (Disponível)
AWS GovCloud (US-East)	Available (Disponível)
AWS GovCloud (US-West)	Available (Disponível)

Considerações ao restaurar um destino de integração de eventos do S3

Ao restaurar um namespace do Amazon Redshift sem servidor a partir de um snapshot ou ponto de recuperação para o mesmo namespace sem servidor, as integrações de eventos do S3 e seus COPY JOBs associados são mantidos automaticamente. As seguintes considerações se aplicam:

- Restaurar um snapshot para um namespace diferente não mantém as integrações de eventos do S3.
- Após a restauração, o Amazon Redshift identifica automaticamente os arquivos que foram carregados para o bucket de origem do Amazon S3 depois que o snapshot foi criado. Todos os arquivos presentes no Amazon S3 que ainda não haviam sido carregados são ingeridos automaticamente. Durante esse processo, a ingestão de novos arquivos provenientes de notificações de eventos do S3 continua funcionando em paralelo.

  > **Note**
  >
  > Após a restauração, somente os arquivos presentes no bucket do Amazon S3 serão elegíveis para ingestão. Os arquivos carregados após a criação do snapshot e posteriormente excluídos antes da conclusão da restauração não são ingeridos.

- Se uma integração de eventos do S3 tiver sido criada após a criação do snapshot, a integração entrará no estado `NEEDS_ATTENTION` após a restauração. Para resolver isso, você pode restaurar a partir de um snapshot mais recente que inclua a integração ou excluir a integração.
- Se uma integração de eventos do S3 tiver sido excluída após a criação do snapshot, a integração não será restaurada. O status do COPY JOB correspondente será alterado para `PENDING`. Você poderá criar uma integração para o mesmo bucket do Amazon S3 e recriar o COPY JOB. Os arquivos adicionados ao S3 entre o momento da criação do snapshot e o momento da recriação dos COPY JOBs não poderão ser recuperados nem ingeridos novamente. Para identificar esses arquivos, verifique o horário de criação do snapshot no console e o horário de exclusão da integração no CloudTrail. Os arquivos que chegaram ao bucket do S3 entre esses dois momentos são os arquivos que não foram processados.
- Se os COPY JOBs tiverem sido alterados após a criação do snapshot, o namespace restaurado usará os COPY JOBs existentes no snapshot. O Amazon Redshift reconciliará todas as diferenças e atualizará a integração para refletir a configuração restaurada dos COPY JOBs.
- Para optar por não manter as integrações de eventos do S3 durante uma restauração, desmarque a caixa Manter integrações na página de restauração do Console de gerenciamento da AWS, ou, se estiver usando a AWS CLI, defina o parâmetro `--no-maintain-integration` ao chamar a operação de API restore-from-snapshot ou restore-from-recovery-point. Quando você desativa, as integrações entram no estado `FAILED` após a restauração.
- Esse recurso se aplica ao Amazon Redshift sem servidor somente quando restaurado no mesmo namespace com tecnologia sem servidor. As restaurações de snapshot em clusters provisionados não mantêm as integrações de eventos do S3.

Carregar tabelas com comandos DML

O Amazon Redshift oferece suporte a comandos de linguagem de manipulação de dados (DML) padrão (INSERT, UPDATE e DELETE) que você pode usar para modificar linhas em tabelas. Você também pode usar o comando TRUNCATE para fazer exclusões em massa rápidas.

> **Note**
>
> Recomendamos veementemente o uso do comando COPY para carregar grandes quantidades de dados. O uso de instruções INSERT individuais para povoar uma tabela pode ser proibitivamente lento. Como alternativa, se seus dados já existem em outras tabelas de banco de dados do Amazon Redshift, use INSERT INTO... SELECT FROM ou CREATE TABLE AS para aprimorar a performance. Para obter informações, consulte INSERT ou CREATE TABLE AS.

Se você inserir dados, atualizar ou excluir um número significativo de linhas de uma tabela em relação ao número de linhas antes das alterações, execute os comandos ANALYZE e VACUUM na tabela quando você tiver terminado. Se um número de pequenas alterações se acumula ao longo do tempo em seu aplicativo, você pode querer agendar a execução dos comandos ANALYZE e VACUUM em intervalos regulares. Para obter mais informações, consulte Análise de tabelas e Vacuum de tabelas.

Tópicos

- Atualização e inserção de novos dados

Atualização e inserção de novos dados

É possível adicionar novos dados de forma eficiente a uma tabela existente usando o comando MERGE. Execute uma operação de mesclagem criando uma tabela intermediária e, depois, use um dos métodos descritos nesta seção para atualizar a tabela de destino pela tabela de preparação. Para obter mais informações sobre o comando MERGE, consulte MERGE.

O Exemplos de mesclagem usa um exemplo de conjunto de dados para o Amazon Redshift, chamado conjunto de dados TICKIT. Como pré-requisito, você pode configurar as tabelas e os dados do TICKIT seguindo as instruções disponíveis em Conceitos básicos das tarefas comuns do banco

de dados. Informações mais detalhadas sobre o exemplo de conjunto de dados estão disponíveis em Exemplo de banco de dados.

Método de mesclagem 1: substituição de linhas existentes

Se você estiver sobrescrevendo todas as colunas na tabela de destino, o método mais rápido para realizar uma mesclagem será substituir as linhas existentes. Isso verifica a tabela de destino apenas uma vez, usando uma junção interna para excluir as linhas que serão atualizadas. Após a exclusão das linhas, elas são substituídas por novas linhas por uma única operação de inserção da tabela de preparação.

Use este método se todos os itens a seguir forem verdadeiros:

- Sua tabela de destino e sua tabela de preparação contêm as mesmas colunas.
- Você pretende substituir todos os dados nas colunas da tabela de destino por todas as colunas da tabela de preparação.
- Você utilizará todas as linhas da tabela de preparação na mesclagem.

Se qualquer um desses critérios não se aplicar, use o “Método de mesclagem 2: especificação de uma lista de colunas sem usar MERGE”, descrito na seção a seguir.

Se você não vai usar todas as linhas da tabela de preparação, filtre as instruções DELETE e INSERT usando uma cláusula WHERE para ignorar linhas que não estejam sendo alteradas. No entanto, se a maioria das linhas da tabela de preparação não participará da mesclagem, recomendamos executar um UPDATE e um INSERT em etapas separadas, conforme descrito posteriormente nesta seção.

Método de mesclagem 2: especificação de uma lista de colunas sem usar MERGE

Use este método para atualizar colunas específicas na tabela de destino em vez de substituir linhas inteiras. Este método leva mais tempo que o método anterior, pois ele requer uma etapa de atualização adicional e não usa o comando MERGE. Use este método se qualquer um dos itens a seguir for verdadeiro:

- Não todas as colunas da tabela de destino devem ser atualizadas.
- A maioria das linhas na tabela de preparação não serão usadas nas atualizações.

Tópicos

- Criação de uma tabela de preparação temporária
- Execução de uma operação de mesclagem com a substituição de linhas existentes
- Realizar uma operação de mesclagem especificando uma lista de colunas sem usar o comando MERGE
- Exemplos de mesclagem

Criação de uma tabela de preparação temporária

A tabela de preparação é uma tabela temporária que guarda todos os dados que serão usados para fazer alterações na tabela de destino, incluindo atualizações e inserções.

Uma operação de mesclagem requer uma junção entre a tabela de preparação e a tabela de destino. Para posicionar as linhas da mesclagem, defina a chave de distribuição da tabela de preparação na mesma coluna que a chave de distribuição da tabela de destino. Por exemplo, se a tabela de destino usa uma coluna de chave estrangeira como chave de distribuição, use a mesma coluna para a chave de distribuição da lista de preparação. Se você cria uma tabela de preparação usando um comando CREATE TABLE LIKE, a tabela de preparação herda a chave de distribuição da tabela pai. Se você utiliza uma instrução CREATE TABLE AS, a nova tabela não herda a chave de distribuição. Para obter mais informações, consulte . Distribuição de dados para otimização de consultas

Se a chave de distribuição não é a mesma que a chave primária e a chave de distribuição não é atualizada como parte da operação de mesclagem, adicione um predicado de junção redundante nas colunas de chave de distribuição para permitir uma junção colocada. Por exemplo:

```
where target.primarykey = stage.primarykey
and target.distkey = stage.distkey
```

Para verificar se consulta usará uma junção colocada, execute a consulta com EXPLAIN e verifique DS_DIST_NONE em todas as junções. Para obter mais informações, consulte . Avaliação do plano de consulta

Execução de uma operação de mesclagem com a substituição de linhas existentes

Ao executar a operação de mesclagem detalhada no procedimento, coloque todas as etapas, exceto de criação e exclusão da tabela de preparação temporária, em uma única transação. A transação será revertida se ocorrer uma falha na etapa. O uso de uma única transação também reduz o número de confirmações, que economiza tempo e recursos.

Para executar uma operação de mesclagem com a substituição de linhas existentes

1. Crie uma tabela de preparação e preencha-a com os dados a serem mesclados, conforme exibido no seguinte pseudocódigo.

```
CREATE temp table stage (like target);

INSERT INTO stage
SELECT * FROM source
WHERE source.filter = 'filter_expression';
```

2. Use MERGE para realizar uma junção interna com a tabela de preparação e atualizar as linhas da tabela de destino que correspondem à tabela de preparação e, depois, insira todas as linhas restantes na tabela de destino que não correspondam à tabela de preparação.

   Recomendamos que você execute as operações de atualização e inserção em um único comando MERGE.

```
MERGE INTO target
USING stage [optional alias] on (target.primary_key = stage.primary_key)
WHEN MATCHED THEN
UPDATE SET col_name1 = stage.col_name1 , col_name2= stage.col_name2, col_name3 =
 {expr}
WHEN NOT MATCHED THEN
INSERT (col_name1 , col_name2, col_name3) VALUES (stage.col_name1, stage.col_name2,
 {expr});
```

3. Descarte a tabela de preparação.

```
DROP TABLE stage;
```

Realizar uma operação de mesclagem especificando uma lista de colunas sem usar o comando MERGE

Ao executar a operação de mesclagem detalhada no procedimento, coloque todas as etapas em uma única transação. A transação será revertida se ocorrer uma falha na etapa. O uso de uma única transação também reduz o número de confirmações, que economiza tempo e recursos.

Para executar uma operação de mesclagem através da especificação de uma lista de colunas

1. Coloque toda a operação em um único bloco de transações.

```
BEGIN transaction;
…
END transaction;
```

2. Crie uma tabela de preparação e preencha-a com os dados a serem mesclados, conforme exibido no seguinte pseudocódigo.

```
create temp table stage (like target);
insert into stage
select * from source
where source.filter = 'filter_expression';
```

3. Atualize a tabela de destino usando um junção interna com a tabela de preparação.

   - Na cláusula UPDATE, liste explicitamente as colunas a serem atualizadas.
   - Execute uma junção interna com a tabela de preparação.
   - Se a chave de distribuição for diferente da chave primária e a chave de distribuição não estiver sendo atualizada, adicione uma junção redundante à chave de distribuição. Para verificar se consulta usará uma junção colocada, execute a consulta com EXPLAIN e verifique DS_DIST_NONE em todas as junções. Para obter mais informações, consulte . Avaliação do plano de consulta
   - Se sua tabela de destino for classificada por timestamp, adicione um predicado para tirar proveito das varreduras de intervalo restrito na tabela de destino. Para obter mais informações, consulte Práticas recomendadas do Amazon Redshift para criar consultas.
   - Se você não pretende usar todas as linhas na mesclagem, adicione uma cláusula para filtrar as linhas que precisam ser alteradas. Por exemplo, adicione um filtro de desigualdade em uma ou mais colunas para excluir as linhas que não alteraram.
   - Coloque as operações de atualização, exclusão e inserção em um único bloco de transações de forma que, se houver um problema, tudo seja revertido.

   Por exemplo:

```
begin transaction;
```

```
update target
set col1 = stage.col1,
col2 = stage.col2,
col3 = 'expression'
from stage
where target.primarykey = stage.primarykey
and target.distkey = stage.distkey
and target.col3 > 'last_update_time'
and (target.col1 != stage.col1
or target.col2 != stage.col2
or target.col3 = 'filter_expression');
```

4. Exclua linhas indesejadas da tabela de preparação usando um junção interna com a tabela de destino. Algumas linhas na tabela de destino já correspondem a linhas na tabela de preparação e outras foram atualizadas na etapa anterior. Em ambos os casos, elas não são necessárias para a inserção.

```
delete from stage
using target
where stage.primarykey = target.primarykey;
```

5. Insira as linhas remanescentes da tabela de preparação. Use a mesma lista de coluna na cláusula VALUES que você usou na instrução UPDATE da etapa dois.

```
insert into target
(select col1, col2, 'expression'
from stage);

end transaction;
```

6. Descarte a tabela de preparação.

```
drop table stage;
```

Exemplos de mesclagem

Os seguintes exemplos executam uma mesclarem para atualizar as tabela SALES. O primeiro exemplo usa o método mais simples de excluir a tabela de destino e depois inserir todas as linhas da tabela de preparação. O segundo exemplo exige a atualização de colunas selecionadas da tabela de destino, portanto ele inclui uma etapa de atualização adicional.

O Exemplos de mesclagem usa um exemplo de conjunto de dados para o Amazon Redshift, chamado conjunto de dados TICKIT. Como pré-requisito, você pode configurar as tabelas e os dados do TICKIT seguindo as instruções disponíveis no guia Conceitos básicos das tarefas comuns do banco de dados. Informações mais detalhadas sobre o exemplo de conjunto de dados estão disponíveis em Exemplo de banco de dados.

Amostra de fonte de dados de mesclagem

Os exemplos nesta seção usam uma fonte de dados de exemplo que inclui atualizações e inserções. Para configurar os dados de exemplo, crie uma tabela chamada SALES_UPDATE a partir da tabela SALES e preencha-a com dados aleatórios que representem novas atividades de vendas de dezembro. Os exemplos a seguir usam a tabela SALES_UPDATE como tabela de preparação.

```
-- Create a sample table as a copy of the SALES table.

create table tickit.sales_update as
select * from tickit.sales;

-- Change every fifth row to have updates.

update tickit.sales_update
set qtysold = qtysold*2,
pricepaid = pricepaid*0.8,
commission = commission*1.1
where saletime > '2008-11-30'
and mod(sellerid, 5) = 0;

-- Add some new rows to have inserts.
-- This example creates a duplicate of every fourth row.

insert into tickit.sales_update
select (salesid + 172456) as salesid, listid, sellerid, buyerid, eventid, dateid,
 qtysold, pricepaid, commission, getdate() as saletime
from tickit.sales_update
where saletime > '2008-11-30'
and mod(sellerid, 4) = 0;
```

Exemplo de uma mesclagem que substitui linhas com base em chaves correspondentes

O exemplo a seguir usa a tabela SALES_UPDATE para realizar uma operação de mesclagem na tabela SALES com novos dados referentes à atividade de vendas de dezembro. A mesclagem

substitui as linhas da tabela SALES que contêm atualizações, modificando as colunas `qtysold` e `pricepaid`, enquanto mantém `commission` e `saletime` inalteradas.

```
MERGE into tickit.sales
USING tickit.sales_update sales_update
on ( sales.salesid = sales_update.salesid
and sales.listid = sales_update.listid
and sales_update.saletime > '2008-11-30'
and (sales.qtysold != sales_update.qtysold
or sales.pricepaid != sales_update.pricepaid))
WHEN MATCHED THEN
update SET qtysold = sales_update.qtysold,
pricepaid = sales_update.pricepaid
WHEN NOT MATCHED THEN
INSERT (salesid, listid, sellerid, buyerid, eventid, dateid, qtysold , pricepaid,
 commission, saletime)
values (sales_update.salesid, sales_update.listid, sales_update.sellerid,
 sales_update.buyerid, sales_update.eventid,
sales_update.dateid, sales_update.qtysold , sales_update.pricepaid,
 sales_update.commission, sales_update.saletime);

-- Drop the staging table.
drop table tickit.sales_update;

-- Test to see that commission and salestime were not impacted.
SELECT sales.salesid, sales.commission, sales.salestime, sales_update.commission,
 sales_update.salestime
FROM tickit.sales
INNER JOIN tickit.sales_update sales_update
ON
sales.salesid = sales_update.salesid
AND sales.listid = sales_update.listid
AND sales_update.saletime > '2008-11-30'
AND (sales.commission != sales_update.commission
OR sales.salestime != sales_update.salestime);
```

Exemplo de uma mesclagem que especifica uma lista de colunas sem usar MERGE

O exemplo a seguir realiza uma operação de mesclagem na tabela SALES com novos dados referentes à atividade de dezembro, usando a tabela SALES_UPDATE como fonte de dados. Os dados de exemplo incluem atualizações, inserções e linhas inalteradas. A mesclagem atualiza as colunas `qtysold` e `pricepaid`, mas mantém `commission` e `saletime` inalteradas.

```
-- Create a staging table and populate it with rows from SALES_UPDATE for Dec
create temp table stagesales as select * from sales_update
where saletime > '2008-11-30';

-- Start a new transaction
begin transaction;

-- Update the target table using an inner join with the staging table
-- The join includes a redundant predicate to collocate on the distribution key -- A
 filter on saletime enables a range-restricted scan on SALES

update sales
set qtysold = stagesales.qtysold,
pricepaid = stagesales.pricepaid
from stagesales
where sales.salesid = stagesales.salesid
and sales.listid = stagesales.listid
and stagesales.saletime > '2008-11-30'
and (sales.qtysold != stagesales.qtysold
or sales.pricepaid != stagesales.pricepaid);

-- Delete matching rows from the staging table
-- using an inner join with the target table

delete from stagesales
using sales
where sales.salesid = stagesales.salesid
and sales.listid = stagesales.listid;

-- Insert the remaining rows from the staging table into the target table
insert into sales
select * from stagesales;

-- End transaction and commit
end transaction;

-- Drop the staging table
drop table stagesales;
```

Execução de uma cópia profunda

Uma cópia profunda recria e preenche novamente uma tabela usando uma inserção em massa, que classifica automaticamente a tabela. Se uma tabela tem uma grande região não classificada, uma cópia profunda é muito mais rápida que um vacuum. Recomendamos que você só faça atualizações simultâneas durante uma operação de cópia profunda se conseguir rastreá-las. Depois que o processo for concluído, mova as atualizações delta para a nova tabela. Uma operação VACUUM oferece suporte para atualizações simultâneas automaticamente.

Você pode escolher um dos seguintes métodos para criar uma cópia da tabela original:

- Use o DDL da tabela original.

  Se CREATE TABLE DDL estiver disponível, este é o método preferido e mais rápido. Se você criar uma nova tabela, você pode especificar todos os atributos da tabela e da coluna, incluindo a chave primária e chaves estrangeiras. É possível encontrar o DDL original usando a função SHOW TABLE.
- Use CREATE TABLE LIKE.

  Se o DDL original não estiver disponível, você pode usar CREATE TABLE LIKE para recriar a tabela original. A nova tabela herda a codificação, a chave de distribuição, a chave de classificação e os atributos não nulos da tabela pai. A nova tabela não herda os atributos de chave primária e chaves estrangeiras da tabela pai, mas você pode adicioná-los usando ALTER TABLE.
- Crie uma tabela temporária e trunque a tabela original.

  Se você precisar manter os atributos da chave primária e da chave externa da tabela principal. Se a tabela principal tiver dependências, você poderá usar CREATE TABLE... AS (CTAS) para criar uma tabela temporária. Depois, trunque a tabela original e preencha-a a partir da tabela temporária.

  O uso de uma tabela temporária aprimora a performance significativamente comparado ao uso de uma tabela permanente, mas há risco de perda de dados. Uma tabela temporária é automaticamente descartada no final da sessão na qual ela é criada. TRUNCATE confirma imediatamente, mesmo se ele estiver dentro de bloco de transações. Se TRUNCATE tiver êxito, mas a sessão for concluída antes da conclusão de INSERT subsequente, os dados serão perdidos. Se a perda de dados for inaceitável, use uma tabela permanente.

Depois de criar uma cópia de uma tabela, talvez seja necessário conceder acesso à nova tabela. Você pode usar GRANT para definir privilégios de acesso. Para visualizar e conceder todos os privilégios de acesso de uma tabela, você deve ser um dos seguintes:

- Um superusuário.
- O proprietário da tabela que você deseja copiar.
- Um usuário com o privilégio ACCESS SYSTEM TABLE para ver os privilégios da tabela e com o privilégio de concessão para todas as permissões relevantes.

Além disso, talvez seja necessário conceder permissão de uso para o esquema em que sua cópia profunda está inserida. A concessão de permissão de uso será necessária se o esquema da cópia profunda for diferente do esquema da tabela original e também não for o esquema `public`. Para visualizar e conceder os privilégios de acesso de uma tabela, você deve ser um dos seguintes:

- Um superusuário.
- Um usuário que possa conceder a permissão USAGE para o esquema da cópia profunda.

Para realizar uma cópia profunda usando o DDL da tabela original

1. (Opcional) Recrie o DDL da tabela executando um script chamado `v_generate_tbl_ddl`.
2. Crie uma cópia da tabela usando o CREATE TABLE DDL original.
3. Use uma instrução INSERT INTO… SELECT para povoar a cópia com dados da tabela original.
4. Confira as permissões concedidas na tabela antiga. Você pode ver essas permissões na visualização do sistema SVV_RELATION_PRIVILEGES.
5. Se necessário, conceda as permissões da tabela antiga para a nova tabela.
6. Conceda permissão de uso a cada grupo e usuário que tenha privilégios na tabela original. Essa etapa não será necessária se sua tabela de cópia profunda estiver no esquema `public` ou estiver no mesmo esquema da tabela original.
7. Descarte a tabela original.
8. Use uma instrução ALTER TABLE para renomear a cópia com o nome da tabela original.

O exemplo a seguir executa uma cópia profunda na tabela SAMPLE usando uma duplicação de SAMPLE chamada sales_copy.

```
--Create a copy of the original table in the sample_namespace namespace using the
 original CREATE TABLE DDL.
create table sample_namespace.sample_copy ( … );

--Populate the copy with data from the original table in the public namespace.
insert into sample_namespace.sample_copy (select * from public.sample);

--Check SVV_RELATION_PRIVILEGES for the original table's privileges.
select * from svv_relation_privileges where namespace_name = 'public' and relation_name
 = 'sample' order by identity_type, identity_id, privilege_type;

--Grant the original table's privileges to the copy table.
grant DELETE on table sample_namespace.sample_copy to group group1;
grant INSERT, UPDATE on table sample_namespace.sample_copy to group group2;
grant SELECT on table sample_namespace.sample_copy to user1;
grant INSERT, SELECT, UPDATE on table sample_namespace.sample_copy to user2;

--Grant usage permission to every group and user that has privileges in the original
 table.
grant USAGE on schema sample_namespace to group group1, group group2, user1, user2;

--Drop the original table.
drop table public.sample;

--Rename the copy table to match the original table's name.
alter table sample_namespace.sample_copy rename to sample;
```

Para realizar uma cópia profunda usando CREATE TABLE LIKE

1. Crie uma nova tabela usando CREATE TABLE LIKE.
2. Use uma instrução INSERT INTO… SELECT para copiar as linhas da tabela atual para a nova tabela.
3. Confira as permissões concedidas na tabela antiga. Você pode ver essas permissões na visualização do sistema SVV_RELATION_PRIVILEGES.
4. Se necessário, conceda as permissões da tabela antiga para a nova tabela.
5. Conceda permissão de uso a cada grupo e usuário que tenha privilégios na tabela original. Essa etapa não será necessária se sua tabela de cópia profunda estiver no esquema `public` ou estiver no mesmo esquema da tabela original.
6. Descarte a tabela atual.

7. Use uma instrução ALTER TABLE para renomear a nova tabela com o nome da tabela original.

O exemplo a seguir executa uma cópia profunda na tabela SAMPLE usando CREATE TABLE LIKE.

```
--Create a copy of the original table in the sample_namespace namespace using CREATE
 TABLE LIKE.
create table sameple_namespace.sample_copy (like public.sample);

--Populate the copy with data from the original table.
insert into sample_namespace.sample_copy (select * from public.sample);

--Check SVV_RELATION_PRIVILEGES for the original table's privileges.
select * from svv_relation_privileges where namespace_name = 'public' and relation_name
 = 'sample' order by identity_type, identity_id, privilege_type;

--Grant the original table's privileges to the copy table.
grant DELETE on table sample_namespace.sample_copy to group group1;
grant INSERT, UPDATE on table sample_namespace.sample_copy to group group2;
grant SELECT on table sample_namespace.sample_copy to user1;
grant INSERT, SELECT, UPDATE on table sample_namespace.sample_copy to user2;

--Grant usage permission to every group and user that has privileges in the original
 table.
grant USAGE on schema sample_namespace to group group1, group group2, user1, user2;

--Drop the original table.
drop table public.sample;

--Rename the copy table to match the original table's name.
alter table sample_namespace.sample_copy rename to sample;
```

Para executar uma cópia profunda criando uma tabela temporária e truncando a tabela original

1. Use CREATE TABLE AS para criar uma tabela temporária com as linhas da tabela original.
2. Trunque a tabela atual.
3. Use uma instrução INSERT INTO… SELECT para copiar as linhas da tabela temporária para a tabela original.
4. Descarte a tabela temporária.

O exemplo a seguir executa uma cópia profunda na tabela SALES criando uma tabela temporária e truncando a tabela original. Como a tabela original permanece, você não precisa conceder permissões à tabela de cópia.

```
--Create a temp table copy using CREATE TABLE AS.
create temp table salestemp as select * from sales;

--Truncate the original table.
truncate sales;

--Copy the rows from the temporary table to the original table.
insert into sales (select * from salestemp);

--Drop the temporary table.
drop table salestemp;
```

Análise de tabelas

A operação ANALYZE atualiza os metadados estatísticos que o planejador de consulta utiliza para escolher os planos ideais.

Na maioria dos casos, não é preciso executar explicitamente o comando ANALYZE. O Amazon Redshift monitora as alterações no workload e atualiza automaticamente as estatísticas em segundo plano. Além disso, o comando COPY executa uma análise automaticamente quando carrega dados em uma tabela vazia.

Para analisar explicitamente uma tabela ou o banco de dados inteiro, execute o comando ANALYZE.

Análise automática

O Amazon Redshift monitora continuamente seu banco de dados e executa automaticamente as operações de análise em segundo plano. Para minimizar o impacto na performance do sistema, a análise automática é executada durante os períodos em que os workloads são leves.

A análise automática está habilitada por padrão. Para desativar a análise automática, defina o parâmetro `auto_analyze` como **`false`** modificando o grupo de parâmetros de seu cluster.

Para reduzir o tempo de processamento e melhorar a performance geral do sistema, o Amazon Redshift ignora a análise automática de qualquer tabela em que a extensão das modificações seja pequena.

Uma operação de análise ignora tabelas com estatísticas atualizadas. Se você executar ANALYZE como parte de seu fluxo de trabalho de extração, transformação e carregamento (ETL), a análise automática ignorará as tabelas com estatísticas atuais. Da mesma forma, um ANALYZE explícito ignorará as tabelas quando a análise automática atualizar as estatísticas da tabela.

Análise de novos dados da tabela

Por padrão, o comando COPY executa um ANALYZE após carregar dados em uma tabela vazia. Você pode forçar um ANALYZE mesmo se uma tabela estiver vazia definindo STATUPDATE ON. Se você especificar STATUPDATE OFF, nenhum ANALYZE será realizado. Somente o proprietário da tabela ou um superusuário podem executar o comando ANALYZE ou executar o comando COPY com STATUPDATE definida como ON.

O Amazon Redshift também analisa novas tabelas que você cria com os seguintes comandos:

- CREATE TABLE AS (CTAS)
- CREATE TEMP TABLE AS
- SELECT INTO

O Amazon Redshift retorna uma mensagem de aviso quando você executa uma consulta em uma nova tabela que não foi analisada depois que seus dados foram carregados inicialmente. Nenhum aviso ocorre quando você consulta uma tabela após uma atualização ou carregamento subsequente. A mesma mensagem de aviso é retornada quando você executa o comando EXPLAIN em uma consulta que se refere a tabelas que não foram analisadas.

Sempre que a adição de dados a uma tabela não vazia alterar significativamente o tamanho da tabela, é possível atualizar estatísticas explicitamente. Isso é feito com a execução de um comando ANALYZE ou usando a opção STATUPDATE ON com o comando COPY. Para visualizar detalhes sobre o número de linhas que foram inseridas ou excluídas desde o último ANALYZE, consulte a tabela de catálogo do sistema PG_STATISTIC_INDICATOR.

Você pode especificar o escopo do comando ANALYZE para um dos seguintes:

- Todo o banco de dados atual
- Uma tabela única
- Uma ou mais colunas específicas em uma tabela única
- Colunas que provavelmente serão usadas como predicados em consultas

O comando ANALYZE obtém um exemplo de linhas da tabela, faz alguns cálculos e salva as estatísticas de coluna resultantes. Por padrão, o Amazon Redshift executa uma passagem de amostra para a coluna DISTKEY e outra passagem de amostra para todas as outras colunas na tabela. Se você quiser gerar estatísticas para um subconjunto de colunas, você pode especificar uma lista de colunas separadas por vírgula. É possível executar ANALYZE com a cláusula PREDICATE COLUMNS para ignorar colunas que não são usadas como predicados.

ANALYZE são operações com uso intenso de recursos, portanto execute as operações somente em tabelas e colunas que realmente exigem atualizações de estatísticas. Você não precisa analisar regularmente todas as colunas em todas as tabelas ou no mesmo agendamento. Se os dados são alterados substancialmente, analise as colunas que são usadas frequentemente para o seguinte:

- Operações de agrupamento e classificação
- Junções
- Predicados de consultas

Para reduzir o tempo de processamento e melhorar a performance geral do sistema, o Amazon Redshift ignora ANALYZE para qualquer tabela que tenha uma baixa porcentagem de linhas alteradas, conforme determinado pelo parâmetro analyze_threshold_percent. Por padrão, o limite de análise é definido como 10 por cento. Você pode alterar o limite de análise para a sessão atual executando um comando SET.

Colunas menos propensas a exigir análises frequentes são aquelas que representam fatos e medidas e quaisquer atributos relacionado que nunca são realmente consultados, tais como grandes colunas VARCHAR. Por exemplo, considere a tabela LISTING no banco de dados TICKIT.

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'listing';


column         |        type        | encoding | distkey | sortkey
---------------+--------------------+----------+---------+---------
listid         | integer            | none     | t       | 1
sellerid       | integer            | none     | f       | 0
eventid        | integer            | mostly16 | f       | 0
dateid         | smallint           | none     | f       | 0
numtickets     | smallint           | mostly8  | f       | 0
priceperticket | numeric(8,2)       | bytedict | f       | 0
totalprice     | numeric(8,2)       | mostly32 | f       | 0
```

```
listtime       | timestamp with...  | none     | f       | 0
```

Se essa tabela for carregada todos os dias com um grande número de novos registros, a coluna LISTID, que é usada frequentemente em consultas como uma chave de junção, deve ser analisada regularmente. Se TOTALPRICE e LISTTIME forem restrições frequentemente usadas em consultas, você pode analisar essas colunas e a chave de distribuição todos os dias úteis.

```
analyze listing(listid, totalprice, listtime);
```

Suponha que os vendedores e eventos no aplicativo sejam muito mais estáticos e os IDs de data se refiram a um conjunto fixo de dias que abrangem apenas dois ou três anos. Nesse caso, os valores exclusivos dessas colunas não são alterados significativamente. Contudo, o número de instâncias de cada valor exclusivo aumentará continuamente.

Além disso, considere o caso em que as medidas NUMTICKETS e PRICEPERTICKET são consultadas raramente em comparação à coluna TOTALPRICE. Nesse caso, você pode executar o comando ANALYZE em toda a tabela todos os finais de semana para atualizar as estatísticas para as cinco colunas que não são analisadas diariamente:

Colunas de predicados

Como uma alternativa conveniente para a especificação de uma lista de colunas, você poderá escolher analisar apenas as colunas que provavelmente serão usadas como predicados. Quando você executa uma consulta, todas as colunas que são usadas em uma cláusula de junção, condição de filtro ou group by são marcadas como colunas de predicados no catálogo de sistema. Quando você executa ANALYZE com a cláusula PREDICATE COLUMNS, a operação de análise inclui somente as colunas que atendem aos seguintes critérios:

- A coluna é marcada como uma coluna de predicado.
- A coluna é uma chave de distribuição.
- A coluna faz parte de uma chave de classificação.

Se nenhuma das colunas de uma tabela está marcada como predicado, ANALYZE inclui todas as colunas, mesmo quando PREDICATE COLUMNS é especificado. Se nenhuma coluna está marcada como coluna de predicado, pode ser que a tabela ainda não tenha sido consultada.

Você pode optar por usar PREDICATE COLUMNS quando o padrão de consulta de seu workload é relativamente estável. Quando o padrão de consulta é variável, com diferentes colunas sendo

frequentemente usadas como predicados, o uso de PREDICATE COLUMNS pode temporariamente resultar em estatísticas obsoletas. Estatísticas obsoletas podem ocasionar planos de ambiente de tempo de execução de consulta pouco satisfatórios e tempos de ambiente de tempo de execução longos. Contudo, na próxima vez que você executar ANALYZE usando PREDICATE COLUMNS, as novas colunas de predicado são incluídas.

Para visualizar detalhes de colunas de predicado, use o seguinte SQL para criar uma exibição chamada PREDICATE_COLUMNS.

```
CREATE VIEW predicate_columns AS
WITH predicate_column_info as (
SELECT ns.nspname AS schema_name, c.relname AS table_name, a.attnum as col_num,
 a.attname as col_name,
        CASE
            WHEN 10002 = s.stakind1 THEN array_to_string(stavalues1, '||')
            WHEN 10002 = s.stakind2 THEN array_to_string(stavalues2, '||')
            WHEN 10002 = s.stakind3 THEN array_to_string(stavalues3, '||')
            WHEN 10002 = s.stakind4 THEN array_to_string(stavalues4, '||')
            ELSE NULL::varchar
        END AS pred_ts
   FROM pg_statistic s
   JOIN pg_class c ON c.oid = s.starelid
   JOIN pg_namespace ns ON c.relnamespace = ns.oid
   JOIN pg_attribute a ON c.oid = a.attrelid AND a.attnum = s.staattnum)
SELECT schema_name, table_name, col_num, col_name,
       pred_ts NOT LIKE '2000-01-01%' AS is_predicate,
       CASE WHEN pred_ts NOT LIKE '2000-01-01%' THEN (split_part(pred_ts,
 '||',1))::timestamp ELSE NULL::timestamp END as first_predicate_use,
       CASE WHEN pred_ts NOT LIKE '%||2000-01-01%' THEN (split_part(pred_ts,
 '||',2))::timestamp ELSE NULL::timestamp END as last_analyze
FROM predicate_column_info;
```

Suponha que você execute a consulta a seguir na tabela LISTING. Observe que LISTID, LISTTIME e EVENTID são usados nas cláusulas de junção, filtro e group by.

```
select s.buyerid,l.eventid, sum(l.totalprice)
from listing l
join sales s on l.listid = s.listid
where l.listtime > '2008-12-01'
group by l.eventid, s.buyerid;
```

Quando você consulta a exibição PREDICATE_COLUMNS, conforme exibido no seguinte exemplo, você vê que LISTID, EVENTID e LISTTIME, estão marcadas como colunas de predicado.

```
select * from predicate_columns
where table_name = 'listing';
```

```
schema_name | table_name | col_num | col_name       | is_predicate |
 first_predicate_use | last_analyze
------------+------------+---------+----------------+--------------
+---------------------+---------------------
public      | listing    |       1 | listid         | true         | 2017-05-05
 19:27:59 | 2017-05-03 18:27:41
public      | listing    |       2 | sellerid       | false        |
  | 2017-05-03 18:27:41
public      | listing    |       3 | eventid        | true         | 2017-05-16
 20:54:32 | 2017-05-03 18:27:41
public      | listing    |       4 | dateid         | false        |
  | 2017-05-03 18:27:41
public      | listing    |       5 | numtickets     | false        |
  | 2017-05-03 18:27:41
public      | listing    |       6 | priceperticket | false        |
  | 2017-05-03 18:27:41
public      | listing    |       7 | totalprice     | false        |
  | 2017-05-03 18:27:41
public      | listing    |       8 | listtime       | true         | 2017-05-16
 20:54:32 | 2017-05-03 18:27:41
```

Manter as estatísticas atualizadas melhora a performance das consultas, permitindo que o planejador de consultas escolha os planos ideais. O Amazon Redshift atualiza as estatísticas automaticamente em segundo plano, e você pode executar explicitamente o comando ANALYZE. Se você optar por executar explicitamente ANALYZE, faça o seguinte:

- Execute o comando ANALYZE antes de executar consultas.
- Execute o comando ANALYZE nos bancos de dados rotineiramente ao final de cada ciclo regular de carregamento ou atualização.
- Execute o comando ANALYZE em todas as tabelas novas que você criar e em todas as tabelas ou colunas existentes submetidas a alterações significativas.
- Considere executar operações ANALYZE em diferentes agendamentos para os diferentes tipos de tabelas e colunas, dependendo de seu uso em consultas e de sua propensão a alterações.

- Para economizar tempo e recursos do cluster, use a cláusula PREDICATE COLUMNS ao executar ANALYZE.

Você não precisa executar explicitamente o comando ANALYZE depois de restaurar um snapshot em um cluster provisionado ou namespace sem servidor, nem depois de retomar um cluster provisionado pausado. O Amazon Redshift preserva as informações da tabela do sistema nesses casos, tornando desnecessários os comandos manuais ANALYZE. O Amazon Redshift continuará a executar operações de análise automática conforme necessário.

Uma operação de análise ignora tabelas com estatísticas atualizadas. Se você executar ANALYZE como parte de seu fluxo de trabalho de extração, transformação e carregamento (ETL), a análise automática ignorará as tabelas com estatísticas atuais. Da mesma forma, um ANALYZE explícito ignorará as tabelas quando a análise automática atualizar as estatísticas da tabela.

Histórico do comando ANALYZE

É útil saber quando o último comando ANALYZE foi executado em uma tabela ou banco de dados. Quando um comando ANALYZE é executado, o Amazon Redshift executa várias consultas semelhantes a esta:

```
padb_fetch_sample: select * from table_name
```

Consultar STL_ANALYZE para visualizar o histórico de operações de análise. Se o Amazon Redshift analisa uma tabela usando análise automática, a coluna `is_background` é definida como `t` (true). Caso contrário, ele será definido como `f` (falso). O exemplo a seguir une a STV_TBL_PERM para mostrar o nome da tabela e os detalhes do ambiente de tempo de execução.

```
select distinct a.xid, trim(t.name) as name, a.status, a.rows, a.modified_rows,
 a.starttime, a.endtime
from stl_analyze a
join stv_tbl_perm t  on t.id=a.table_id
where name = 'users'
order by starttime;


xid    | name  | status          | rows  | modified_rows | starttime          |
 endtime
-------+-------+-----------------+-------+---------------+--------------------
+--------------------
```

```
  1582 | users | Full            | 49990 |         49990 | 2016-09-22 22:02:23 |
 2016-09-22 22:02:28
244287 | users | Full            | 24992 |         74988 | 2016-10-04 22:50:58 |
 2016-10-04 22:51:01
244712 | users | Full            | 49984 |         24992 | 2016-10-04 22:56:07 |
 2016-10-04 22:56:07
245071 | users | Skipped         | 49984 |             0 | 2016-10-04 22:58:17 |
 2016-10-04 22:58:17
245439 | users | Skipped         | 49984 |          1982 | 2016-10-04 23:00:13 |
 2016-10-04 23:00:13
(5 rows)
```

Como alternativa, você pode executar uma consulta mais complexa que retorna todas as instruções executadas em todas as transações concluídas que incluíram um comando ANALYZE:

```
select xid, to_char(starttime, 'HH24:MM:SS.MS') as starttime,
datediff(sec,starttime,endtime ) as secs, substring(text, 1, 40)
from svl_statementtext
where sequence = 0
and xid in (select xid from svl_statementtext s where s.text like 'padb_fetch_sample
%' )
order by xid desc, starttime;

xid  |  starttime   | secs |                 substring
-----+--------------+------+-----------------------------------------
1338 | 12:04:28.511 |    4 | Analyze date
1338 | 12:04:28.511 |    1 | padb_fetch_sample: select count(*) from
1338 | 12:04:29.443 |    2 | padb_fetch_sample: select * from date
1338 | 12:04:31.456 |    1 | padb_fetch_sample: select * from date
1337 | 12:04:24.388 |    1 | padb_fetch_sample: select count(*) from
1337 | 12:04:24.388 |    4 | Analyze sales
1337 | 12:04:25.322 |    2 | padb_fetch_sample: select * from sales
1337 | 12:04:27.363 |    1 | padb_fetch_sample: select * from sales
...
```

Vacuum de tabelas

O Amazon Redshift pode classificar e executar automaticamente uma operação VACUUM DELETE nas tabelas em segundo plano. Para limpar tabelas após um carregamento ou uma série de atualizações incrementais, você também pode executar o comando VACUUM no banco de dados inteiro ou em tabelas individuais.

> **Note**
>
> Somente usuários com as permissões de tabela necessárias podem efetivamente limpar uma tabela. Se VACUUM for executado sem as permissões necessárias de tabela, a operação será concluída com êxito, mas sem efeito. Para obter uma lista das permissões de tabela válidas para executar efetivamente VACUUM, consulte VACUUM.
> Por esse motivo, é recomendável limpar tabelas individuais conforme necessário. Também recomendamos essa abordagem porque limpar todo o banco de dados é uma operação potencialmente cara.

Classificação automática de tabela

O Amazon Redshift classifica os dados automaticamente em segundo plano para manter os dados da tabela na ordem de sua chave de classificação. O Amazon Redshift mantém o controle de suas consultas de verificação para determinar quais seções da tabela se beneficiarão com a classificação. O Amazon Redshift também acompanha as consultas de verificação de clusters de escalabilidade simultânea. Para arquiteturas de vários clusters que usam o Amazon Redshift Data Sharing, o Amazon Redshift também rastreia consultas de varredura originadas de clusters ou grupos de trabalho consumidores no seu data mesh, incluindo clusters ou grupos de trabalho em diferentes regiões. As estatísticas de verificação do cluster principal, dos clusters de escalabilidade de simultaneidade e dos clusters de consumidor são agregadas para determinar quais seções da tabela se beneficiarão da classificação.

Dependendo da carga no sistema, o Amazon Redshift inicia automaticamente a classificação. A classificação automática reduz a necessidade de executar o comando VACUUM para manter os dados na ordem da chave de classificação. Se for necessário classificar os dados totalmente na ordem da chave de classificação, por exemplo, depois de um grande carregamento de dados, você ainda poderá executar o comando VACUUM manualmente. Para determinar se a tabela terá algum benefício com a execução de VACUUM SORT, monitore a coluna `vacuum_sort_benefit` em SVV_TABLE_INFO.

O Amazon Redshift rastreia consultas de varredura que usam a chave de classificação em cada tabela. O Amazon Redshift estima a porcentagem máxima de melhoria na verificação e filtragem de dados para cada tabela (se a tabela foi totalmente classificada). A estimativa é visível na coluna `vacuum_sort_benefit` em SVV_TABLE_INFO. Você pode usar essa coluna, junto com a coluna `unsorted`, para determinar quando as consultas podem se beneficiar da execução manual de VACUUM SORT em uma tabela. A coluna `unsorted` reflete a ordem de classificação física de

uma tabela. A coluna `vacuum_sort_benefit` especifica o impacto da classificação de uma tabela executando manualmente VACUUM SORT.

Por exemplo, considere a seguinte consulta:

```
select "table", unsorted,vacuum_sort_benefit from svv_table_info order by 1;
```

```
 table | unsorted | vacuum_sort_benefit
-------+----------+---------------------
 sales |    85.71 |                5.00
 event |    45.24 |               67.00
```

Para a tabela “sales”, embora a tabela esteja ~86% não classificada fisicamente, o impacto da performance da consulta na tabela que está 86% não classificada é de apenas 5%. Isso pode ocorrer porque apenas uma parte pequena da tabela é acessada por consultas ou porque muito poucas consultas acessaram a tabela. No caso da tabela “event”, a tabela está ~45% não classificada fisicamente. Mas o impacto da performance da consulta de 67% indica que uma parte maior da tabela foi acessada por consultas ou o número de consultas que acessaram a tabela era grande. A tabela “event” pode se beneficiar potencialmente da execução de VACUUM SORT.

Exclusão de vacuum automática

Ao executar uma exclusão, as linhas são marcadas para exclusão, mas não removidas. O Amazon Redshift executa automaticamente uma operação VACUUM DELETE em segundo plano com base no número de linhas excluídas nas tabelas de banco de dados. O Amazon Redshift programa o VACUUM DELETE para execução durante períodos de carga reduzida e pausa a operação durante períodos de alta carga.

Tópicos

- Frequência de VACUUM
- Estágio do classificação e estágio de mesclagem
- Limite de vacuum
- Tipos de vacuum
- Minimizar tempos de limpeza

Frequência de VACUUM

Você deve limpar com a frequência necessária para manter a performance consistente de consulta. Considere esses fatores ao determinar com que frequência executar o comando VACUUM:

- Execute VACUUM durante períodos em que você espera atividade mínima no cluster, tais como noites ou durante janelas designadas de administração do banco de dados.
- Execute os comandos VACUUM fora das janelas de manutenção. Para obter mais informações, consulte Agendamento em torno de janelas de manutenção.
- Uma grande região não classificada resulta em tempos mais longos de limpeza. Se você adiar a limpeza, ela levará mais tempo, pois mais dados precisam ser reorganizados.
- VACUUM é uma operação uso intensivo de E/S, portanto quanto mais tempo sua limpeza levar para concluir, maior será o impacto sobre consultas simultâneas e outras operações de banco de dados em execução no seu cluster.
- VACUUM leva mais tempo para tabelas que usam classificação intercalada. Para avaliar se tabelas intercaladas precisam ser reclassificadas, consulte a visualização SVV_INTERLEAVED_COLUMNS.

Estágio do classificação e estágio de mesclagem

O Amazon Redshift realiza uma operação de vácuo em dois estágios: primeiro, ele classifica as linhas na região não classificada e, se necessário, mescla as linhas recém-classificadas no final da tabela com as linhas existentes. Ao limpar uma tabela grande, a operação de limpeza continua em uma série de etapas que consistem em classificações incrementais seguidas de mesclagens. Se a operação falhar ou se o Amazon Redshift ficar off-line durante a limpeza, a tabela ou o banco de dados parcialmente limpo estará em um estado consistente, mas você precisará reiniciar manualmente a operação de limpeza. As classificações incrementais serão perdidas, mas as linhas mescladas que foram confirmadas antes da falha não precisam ser limpas novamente. Se a região não classificada é grande, o tempo perdido pode ser significativo. Para obter mais informações sobre as etapas de classificação e mesclagem, consulte Reduzir o volume de linhas mescladas.

Os usuários podem acessar tabelas enquanto elas estão sendo limpas. Você pode executar consultas e operações de gravação enquanto uma tabela está sendo limpa, mas quando DML e uma limpeza são executados simultaneamente, ambos podem levar mais tempo. Se você executar instruções UPDATE e DELETE durante uma limpeza, a performance do sistema pode ser reduzida.

Mesclagens incrementais bloqueiam temporariamente operações UPDATE e DELETE simultâneas, e operações UPDATE e DELETE simultâneas, por usa vez, bloqueia as etapas de mesclagem incremental nas tabelas afetadas. As operações DDL, tal como ALTER TABLE, são bloqueadas até que a operação de limpeza termine com a tabela.

> **Note**
>
> Vários modificadores para VACUUM controlam a forma como ele funciona. Você pode usá-los para adaptar a operação de vácuo para a necessidade atual. Por exemplo, usando VACUUM RECLUSTER encurta a operação de vácuo não executando uma operação de mesclagem completa. Para obter mais informações, consulte VACUUM.

Limite de vacuum

Por padrão, VACUUM ignora a fase de classificação para qualquer tabela em que mais de 95 por cento das linhas da tabela já estejam classificadas. Ignorar a fase de classificação pode melhorar significativamente a performance de VACUUM. Para alterar o limite de classificação padrão para uma única tabela, inclua o nome da tabela e o parâmetro TO limite PERCENT ao executar o comando VACUUM.

Tipos de vacuum

Para obter informações sobre os diferentes tipos de vácuo, consulte VACUUM.

Minimizar tempos de limpeza

O Amazon Redshift classifica os dados automaticamente e executa VACUUM DELETE em segundo plano. Isso reduz a necessidade de executar o comando VACUUM. O processo de limpeza pode ser demorado. Dependendo da natureza dos dados, recomendamos seguir as práticas nesta seção para minimizar os tempos de limpeza.

Tópicos

- Decidir sobre a reindexação
- Reduzir o tamanho da região não classificada
- Reduzir o volume de linhas mescladas
- Carregar os dados por ordem de chave de classificação

- Usar tabelas de séries temporais para reduzir os dados armazenados

Decidir sobre a reindexação

Frequentemente, você pode melhorar significativamente a performance da consulta usando um estilo intercalado de classificação, mas ao longo do tempo a performance pode se degradar se a distribuição de valores nas colunas de chaves de classificação alterar.

Ao carregar inicialmente uma tabela intercalada vazia usando COPY ou CREATE TABLE AS, o Amazon Redshift constrói automaticamente o índice intercalado. Se você inicialmente carregar uma tabela intercalada usando INSERT, você precisa executar um VACUUM REINDEX posteriormente para inicializar o índice intercalado.

Com o tempo, à medida que você adiciona linhas com novos valores de chave de classificação, a performance pode se degradar se a distribuição de valores nas colunas de chaves de classificação alterar. Se suas novas linhas caem principalmente no intervalo de valores de chave de classificação existente, você não precisa reindexar. Execute VACUUM SORT ONLY ou VACUUM FULL para restaurar a ordem de classificação.

O mecanismo de consulta é capaz de usar a ordem de classificação para selecionar com eficiência quais blocos de dados devem ser varridos para processar uma consulta. Para uma classificação intercalada, o Amazon Redshift analisa os valores da coluna de chave de classificação para determinar a ordem de classificação ideal. Se a distribuição dos valores de chave muda ou é distorcida à medida que linhas são adicionadas, a estratégia de classificação não será mais ideal e o benefício de performance de classificação se degradará. Para reanalisar a distribuição de chaves de classificação, você pode executar um VACUUM REINDEX. A operação de reindexação consome tempo, portanto para decidir se uma tabela se beneficiará de uma de reindexação, consulte a exibição SVV_INTERLEAVED_COLUMNS.

Por exemplo, a seguinte consulta exibe os detalhes de tabelas que usam chaves de classificação intercaladas.

```
select tbl as tbl_id, stv_tbl_perm.name as table_name,
col, interleaved_skew, last_reindex
from svv_interleaved_columns, stv_tbl_perm
where svv_interleaved_columns.tbl = stv_tbl_perm.id
and interleaved_skew is not null;


 tbl_id | table_name | col | interleaved_skew | last_reindex
```

```
--------+------------+-----+------------------+--------------------
 100048 | customer   |   0 |             3.65 | 2015-04-22 22:05:45
 100068 | lineorder  |   1 |             2.65 | 2015-04-22 22:05:45
 100072 | part       |   0 |             1.65 | 2015-04-22 22:05:45
 100077 | supplier   |   1 |             1.00 | 2015-04-22 22:05:45
(4 rows)
```

O valor para `interleaved_skew` é uma proporção que indica a quantidade de distorção. Um valor de 1 significa que não há distorção. Se a distorção for maior que 1,4, um VACUUM REINDEX melhorará a performance a menos que a distorção seja inerente ao conjunto subjacente.

Você pode usar o valor de data em `last_reindex` para determinar quanto tempo se passou desde a última reindexação.

Reduzir o tamanho da região não classificada

A região não classificada aumenta quando você carrega grandes quantidades de novos dados em tabelas que já contêm dados ou quando você não limpa as tabelas como parte de suas operações de manutenção de rotina. Para evitar operações de limpeza de longa execução, use as seguintes práticas:

- Execute operações de limpeza em uma programação regular.

  Se você carregar suas tabelas em pequenos incrementos (como atualizações diárias que representam uma pequena porcentagem do número total de linhas da tabela) executar VACUUM regularmente ajudará a garantir que as operações individuais de VACUUM sejam rápidas.
- Execute o maior carregamento primeiro.

  Se você precisar carregar uma nova tabela com várias operações COPY, execute o maior carregamento primeiro. Quando você executa um carregamento inicial em uma tabela nova ou truncada, todos os dados são carregados diretamente na região classificada, portanto nenhuma limpeza é necessária.
- Trunque uma tabela em vez de excluir todas as linhas.

  A exclusão de linhas de uma tabela não recupera o espaço que as linhas ocupavam até que você execute uma operação de limpeza; entretanto, truncar uma tabela esvazia a tabela e recupera o espaço em disco, portanto nenhuma limpeza é necessária. Como alternativa, descarte a tabela e volte a criá-la.
- Trunque ou descarte tabelas de teste.

Se você estiver carregando um pequeno número de linhas em uma tabela para fins de teste, não exclua as linhas quando tiver terminado. Em vez disso, trunque a tabela e recarregue essas linhas como parte da operação de carregamento de produção subsequente.

- Execute uma cópia profunda.

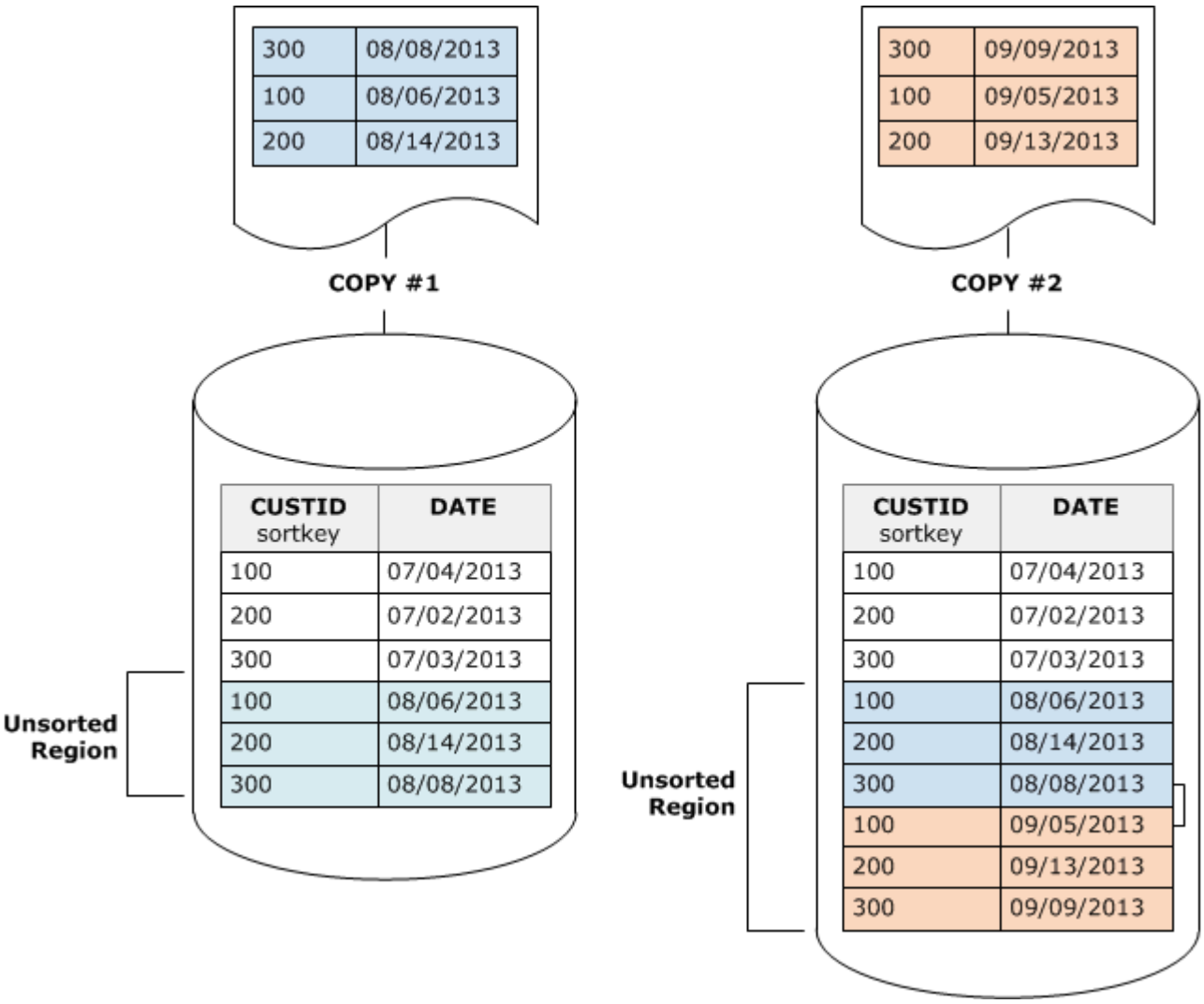
  Se uma tabela que usa uma chave de classificação composta tem uma grande região não classificada, uma cópia profunda é muito mais rápida que um vacuum. Uma cópia profunda recria e preenche novamente uma tabela usando uma inserção em massa, que reclassifica a tabela automaticamente. Se uma tabela tem uma grande região não classificada, uma cópia profunda é muito mais rápida que um vacuum. A diferença é que você não pode realizar atualizações simultâneas durante uma operação de cópia profunda, o que pode ocorrer durante um vacuum. Para obter mais informações, consulte Práticas recomendadas do Amazon Redshift para criar consultas.

Reduzir o volume de linhas mescladas

Se uma operação de limpeza precisar mesclar novas linhas na região classificada de uma tabela, o tempo necessário para uma limpeza aumentará à medida que a tabela crescer. Você pode melhorar a performance da limpeza reduzindo o número de linhas que devem ser mescladas.

Antes de uma limpeza, uma tabela consiste em uma região classificada no topo da tabela, seguida por uma região não classificada, que aumenta sempre que linhas são adicionadas ou atualizadas. Quando um conjunto de linhas é adicionado por uma operação COPY, o novo conjunto de linhas é classificado na chave de classificação à medida que é adicionado à região não classificada no final da tabela. As novas linhas são ordenadas dentro de seu próprio conjunto, mas não na região não classificada.

O diagrama a seguir ilustra a região não classificada após duas operações COPY sucessivas, onde a chave de classificação é CUSTID. Por simplicidade, este exemplo mostra uma chave de classificação composta, mas os mesmos princípios se aplicam à chaves de classificação intercaladas, salvo que o impacto da região não classificada é maior para tabelas intercaladas.



Uma limpeza restaura a ordem de classificação de uma tabela em duas etapas:

1. Classifique a região não classificada em uma região recém-classificada.

   A primeira etapa é relativamente barata, pois somente a região não classificada é regravada. Se o intervalo de valores de chave de classificação da região recém-classificada for maior que o intervalo existente, somente as novas linhas precisarão ser regravadas e a limpeza será concluída. Por exemplo, se a região classificada contém valores de ID de 1 a 500 e as operações copy subsequentes adicionam valores de chave maiores que 500, então somente a região não classificada precisa ser regravada.

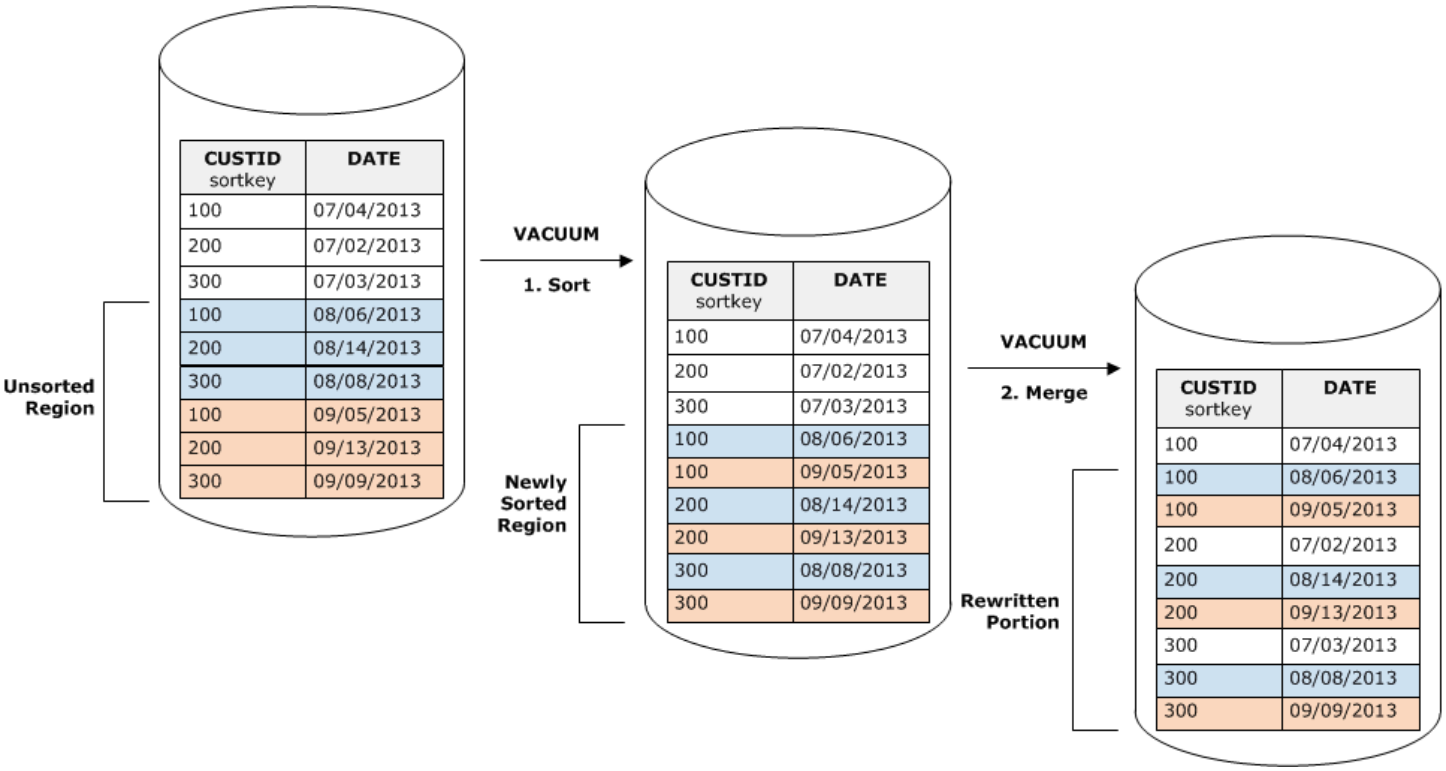
2. Mesclagem da região recém-classificada com a região previamente classificada.

   Se as chaves na região classificada recentemente classificada sobrepõem-se às chaves da região classificada, VACUUM precisa mesclar as linhas. Começando no início da região recém-

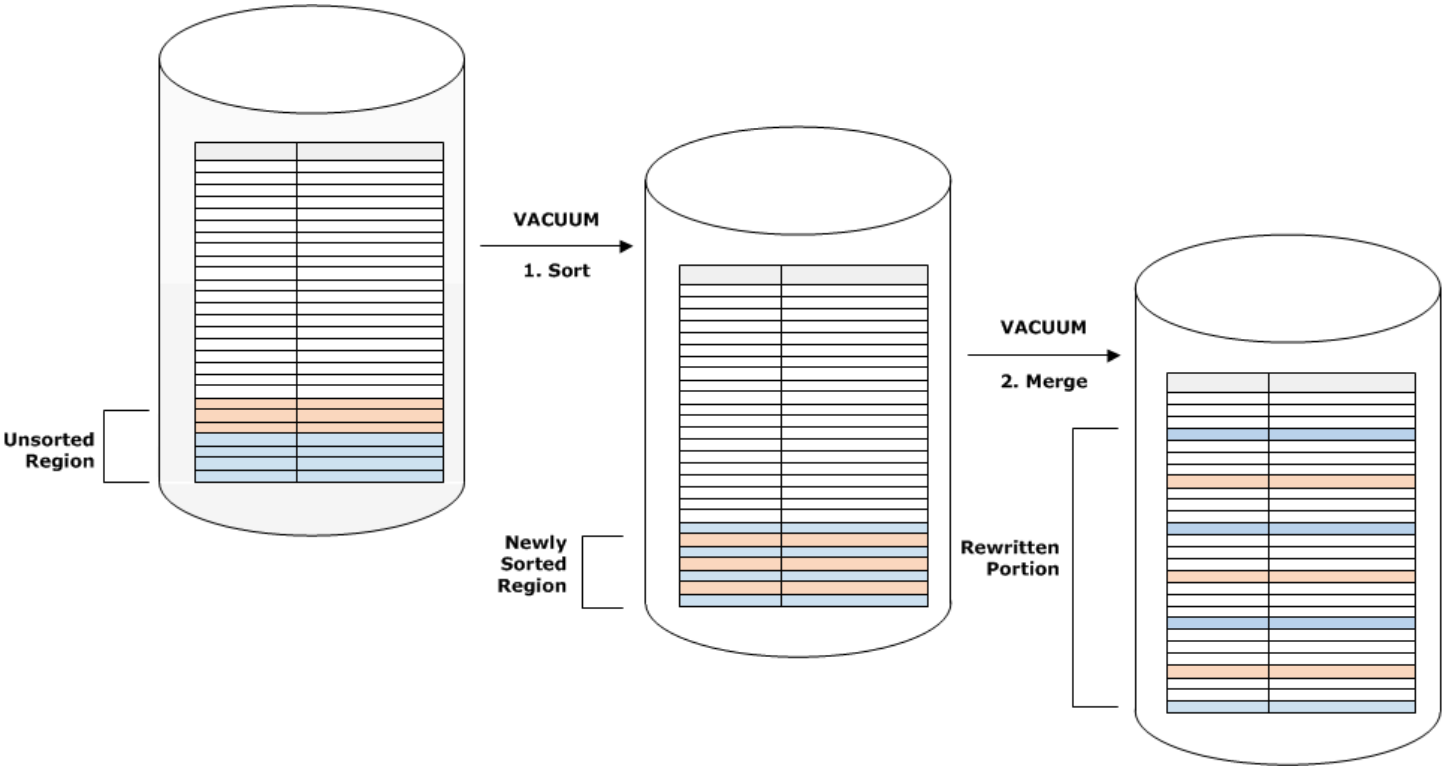
classificada (a chave de classificação mais baixa), a limpeza grava as linhas mescladas da região previamente classificada e a região recém classificada em um novo conjunto de blocos.

A extensão com que o novo intervalo de chaves de classificação sobrepõe as chaves de classificação existentes determina a extensão na qual a região previamente classificada deve ser regravada. Se as chaves não classificadas estão espalhadas pelo intervalo de classificação existente, a limpeza pode precisar regravar porções existentes da tabela.

O diagrama a seguir mostra como uma limpeza classificaria e mesclaria as linhas que são adicionadas em uma tabela em que CUSTID é a chave de classificação. Como cada operação copy adiciona um novo conjunto de linhas com valores de chave que se sobrepõem às chaves existente, quase toda a tabela precisa ser regravada. O diagrama mostra uma única classificação e mesclagem, mas na prática, uma grande limpeza consiste em uma serie de etapas de classificação e mesclagem incrementais.



Se o intervalo de chaves de classificação em um novo conjunto de linhas sobrepõe o intervalo de chaves existente, o custo da etapa de mesclagem continua crescendo na proporção do tamanho da tabela à medida que ela cresce, enquanto o custo da etapa de classificação permanece proporcional ao tamanho da região não classificada. Nesse caso, o custo da etapa de mesclagem ofusca o custo da etapa de classificação, conforme exibido no diagrama a seguir.



Para determinar que proporção de uma tabela foi remesclada, consulte SVV_VACUUM_SUMMARY após a conclusão da operação de limpeza. A seguir consulta exibe os efeitos de seis limpezas consecutivas à medida que CUSTSALES cresceu ao longo do tempo.

```
select * from svv_vacuum_summary
where table_name = 'custsales';


 table_name | xid  | sort_      | merge_     | elapsed_   | row_  | sortedrow_ | block_
  | max_merge_
            |      | partitions | increments | time       | delta | delta      | delta
  | partitions
 -----------+------+------------+------------+------------+-------+------------
+---------+---------------
  custsales | 7072 |          3 |          2 |  143918314 |     0 |   88297472 |   1524
  |      47
  custsales | 7122 |          3 |          3 |  164157882 |     0 |   88297472 |    772
  |      47
  custsales | 7212 |          3 |          4 |  187433171 |     0 |   88297472 |    767
  |      47
  custsales | 7289 |          3 |          4 |  255482945 |     0 |   88297472 |    770
  |      47
```

```
 custsales | 7420 |          3 |          5 |  316583833 |     0 |   88297472 |    769
 |      47
 custsales | 9007 |          3 |          6 |  306685472 |     0 |   88297472 |    772
 |      47
(6 rows)
```

A coluna merge_increments fornece uma indicação da quantidade de dados que foram mesclados em cada operação de limpeza. Se o número de incrementos de mesclagem ao longo de limpezas consecutivas aumentar em proporção ao crescimento do tamanho da tabela, isso é uma indicação de que cada operação de limpeza está mesclando novamente um número crescente de linhas na tabela em decorrência da sobreposição das regiões classificadas existentes e novas.

Carregar os dados por ordem de chave de classificação

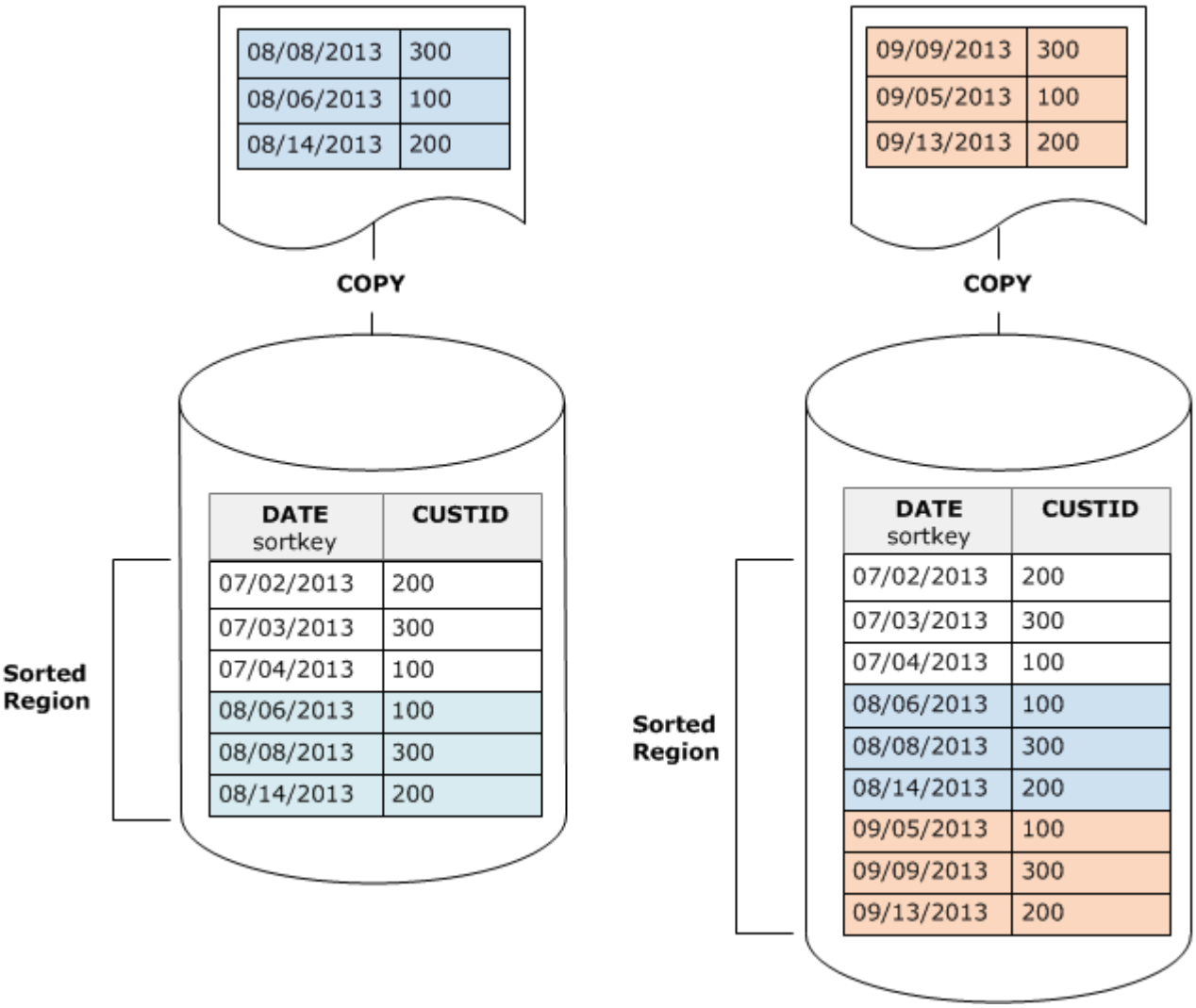
Se você carregar os dados na ordem da chave de classificação usando o comando COPY, poderá reduzir ou até mesmo eliminar a necessidade de limpeza.

COPY adiciona automaticamente linhas novas à região classificada da tabela quando todas as seguintes opções são verdadeiras:

- A tabela usa uma chave de classificação composta com somente uma coluna de classificação.
- A coluna de classificação é NOT NULL.
- A tabela está 100 por cento classificada ou vazia.
- Todas as linhas novas são mais altas na ordem de classificação que as linhas existentes, incluindo as linhas marcadas para exclusão. Nesse caso, o Amazon Redshift usa os primeiros oito bytes da chave de classificação para determinar a ordem de classificação.
- O comando COPY não aciona determinadas otimizações de carga. Ao carregar grandes volumes de dados, o Amazon Redshift pode otimizar a performance criando partições classificadas em vez de adicionar linhas à região classificada da tabela.

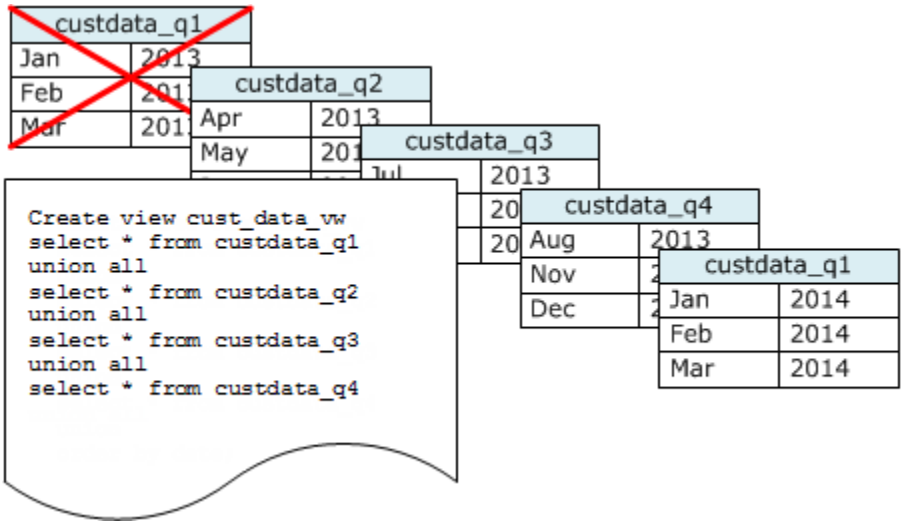
Por exemplo, suponha que você tenha uma tabela que registre eventos de clientes usando um ID de cliente e hora. Se você classificar pelo ID de cliente, é provável que o intervalo de chaves de classificação de novas linhas adicionadas por carregamentos incrementais se sobreponha ao intervalo existente, conforme mostrado no exemplo anterior, resultando em uma operação de limpeza dispendiosa.

Se você definir sua chave de classificação como uma coluna de carimbo de data/hora, suas novas linhas serão anexadas por ordem de classificação no final da tabela, conforme exibido no diagrama a seguir, reduzindo ou até eliminando a necessidade de limpeza.



Usar tabelas de séries temporais para reduzir os dados armazenados

Se você mantém dados para um período de rolamento, use uma série de tabelas, como o diagrama a seguir ilustra.



Crie uma nova tabela sempre que você adicionar um conjunto de dados e, então, exclua a tabela mais velha da série. Você obtém um benefício duplo:

- Você evita o custo adicional da exclusão das linhas, pois uma operação DROP TABLE é muito mais eficiente que um DELETE em massa.
- Se as tabelas são classificadas por timestamp, nenhuma limpeza é necessária. Se cada tabela contém os dados para um mês, uma limpeza no máximo terá que regravar um mês de dados, mesmo que as tabelas não estejam classificadas por timestamp.

Você pode criar uma exibição UNION ALL para uso reportando consultas que ocultam o fato de que os dados são armazenados em várias tabelas. Se uma consulta filtrar na chave de classificação, o planejador de consulta pode eficientemente ignorar todas as tabelas que não são usadas. Um UNION ALL pode ser menos eficiente para outros tipos de consultas, portanto você deve avaliar a performance de consultas no contexto de todas as consultas que utilizam as tabelas.

Gerenciamento de operações de gravação simultâneas

Alguns aplicativos exigem não apenas consultas e o carregamentos simultâneos, mas também a capacidade de gravar em várias tabelas ou na mesma tabela simultaneamente. Nesse contexto, simultaneamente significa em sobreposição, e não programada para execução precisamente ao mesmo tempo. Duas transações são consideradas simultâneas se a segunda começar antes da confirmação da primeira transação. Operações simultâneas podem originar de sessões diferentes que são controladas pelo mesmo usuário ou por usuários diferentes.

O Amazon Redshift oferece suporte a esses tipos de aplicativos, permitindo que as tabelas sejam lidas enquanto são carregadas ou modificadas de forma incremental. Consultas simplesmente observam a última versão confirmada, ou snapshot dos dados, em vez de esperar pela confirmação da próxima versão. Caso você queira que determinada consulta aguarde a confirmação de outra operação de gravação, você deve programá-la adequadamente.

> **Note**
>
> O Amazon Redshift oferece suporte a um comportamento de confirmação automática padrão em que cada comando SQL executado separadamente é confirmado individualmente. Se você inserir um conjunto de comandos em um bloco de transação (definido pelas instruções BEGIN e END), o bloco é confirmado como uma transação, de forma que você possa revertê-la se necessário. Exceções a esse comportamento são os comandos TRUNCATE e VACUUM, que confirmam automaticamente todas as alterações pendentes feitas na transação atual.
>
> Alguns clientes SQL emitem comandos BEGIN e COMMIT automaticamente, de modo que o cliente possa controlar se um grupo de instruções é executado como uma transação ou cada instrução individual é executada como sua própria transação. Verifique a documentação da interface que você está usando. Por exemplo, ao usar o driver Amazon Redshift JDBC, um `PreparedStatement` JDBC com uma string de consulta que contém vários comandos SQL (separados por ponto e vírgula) executa todas as instruções como uma única transação. Por outro lado, se você usar SQL Workbench/J, definir AUTO COMMIT ON e executar várias instruções, cada instrução será executada como sua própria transação.

Os seguintes tópicos descrevem alguns dos conceitos chave e casos de uso que envolvem transações, snapshots de banco de dados, atualizações e comportamentos concorrentes.

Tópicos

- Níveis de isolamento no Amazon Redshift
- Operações de gravação e de leitura/gravação
- Exemplos de gravações simultâneas
- Solucionar problemas de erros de isolamento serializável

Níveis de isolamento no Amazon Redshift

As operações de gravação simultâneas são aceitas no Amazon Redshift de forma protetora, usando bloqueios de gravação em tabelas e o princípio de isolamento serializável. O isolamento serializável preserva a ilusão de que uma transação em execução em uma tabela é a única transação em execução naquela tabela.

Os bancos de dados do Amazon Redshift comportam operações de gravação simultâneas fazendo com que cada operação use a versão confirmada mais recente, ou o snapshot, de seus dados no início da transação. Um snapshot do banco de dados é criado em uma transação na primeira ocorrência da maioria das instruções SELECT, comandos DML tais como COPY, DELETE, INSERT, UPDATE e TRUNCATE, e após os seguintes comandos DDL:

- ALTER TABLE (para adicional ou descartar colunas)
- CRIAR TABELA
- DESCARTAR TABELA
- TRUNCATE TABLE

Nenhuma outra transação consegue alterar esse snapshot, o que significa que as transações são isoladas umas das outras. Ou seja, as transações simultâneas são invisíveis entre si; não podem detectar as alterações uma das outras.

Qualquer execução simultânea de transações deve produzir os mesmos resultados que a execução em série dessas transações. Se nenhuma execução serial dessas transações pode produzir os mesmos resultados, a transação que executa uma instrução que pode quebrar a capacidade de serializar é interrompida e revertida.

Por exemplo, suponha que um usuário tente executar duas transações simultâneas, T1 e T2. A execução de T1 e T2 deve produzir os mesmos resultados de pelo menos um dos seguintes casos:

- T1 e T2 são executadas em série nessa ordem.
- T2 e T1 são executadas em série nessa ordem.

Os níveis de isolamento no Amazon Redshift evitam os seguintes problemas:

- Leituras sujas: uma leitura suja ocorre quando uma transação lê dados que ainda não foram confirmados. Por exemplo, suponha que a transação 1 atualize uma linha. A transação 2 lê a linha

atualizada antes que T1 confirme a atualização. Se T1 reverter a alteração, T2 terá lido dados em linhas não confirmadas que o Amazon Redshift agora considera que nunca existiram.

- Leituras não repetíveis: uma leitura não repetível ocorre quando uma única transação lê a mesma linha duas vezes, mas recebe dados diferentes a cada vez. Por exemplo, digamos que a transação 1 leia uma linha. A transação 2 atualiza ou exclui essa linha e confirma a atualização ou exclusão. Se T1 reler a linha, recuperará valores de linha diferentes ou descobrirá que a linha foi excluída.
- Fantasmas: fantasma é uma linha que corresponde aos critérios de pesquisa, mas não é vista inicialmente. Por exemplo, suponha que a transação 1 leia um conjunto de linhas que atenda aos seus critérios de pesquisa. A transação 2 gera uma nova linha em uma instrução UPDATE ou INSERT que corresponda aos critérios de pesquisa para T1. Se T1 executar novamente sua instrução de pesquisa, ele receberá um conjunto diferente de linhas.

Isolamento SNAPSHOT e SERIALIZABLE

Os isolamentos SNAPSHOT e SERIALIZABLE são os dois níveis de isolamento serializáveis disponíveis no Amazon Redshift.

O isolamento SNAPSHOT é o nível de isolamento padrão ao criar clusters provisionados e grupos de trabalho sem servidor, permitindo processar volumes maiores de dados do que o isolamento SERIALIZABLE em menos tempo.

O isolamento SERIALIZABLE leva mais tempo, mas implementa restrições mais rígidas em transações simultâneas. Esse nível de isolamento evita problemas, como anomalias de distorção de gravação, permitindo que apenas uma transação seja confirmada, enquanto cancela todas as outras transações simultâneas com um erro de violação de isolamento serializável.

Veja a seguir um exemplo de cronograma de como duas operações de gravação simultâneas seriam tratadas ao usar o isolamento SNAPSHOT. A declaração UPDATE de cada usuário pode ser confirmada porque ela não entra em conflito ao tentar atualizar as mesmas linhas.

Hora	Ação do usuário 1	Ação do usuário 2
1	BEGIN;	
2		BEGIN;

Hora	Ação do usuário 1	Ação do usuário 2

3	SELECT * FROM Numbers
	`digits`
	`------`
	`0`
	`1`	

4		SELECT * FROM Numbers;
		`digits`
		`------`
		`0`
		`1`

5	UPDATE Numbers SET digits=0 WHERE digits=1;	

Hora	Ação do usuário 1	Ação do usuário 2

6	SELECT * FROM Numbers
	`digits`
	`------`
	`0`
	`0`	

7	COMMIT	
8		Update Numbers SET digits=1 WHERE digits=0;

9		SELECT * FROM Numbers;
		`digits`
		`------`
		`1`
		`1`

10		COMMIT;

Hora	Ação do usuário 1	Ação do usuário 2

11	SELECT * FROM Numbers
	`digits`
	`------`
	`1`
	`0`	

12		SELECT * FROM Numbers;
		`digits`
		`------`
		`1`
		`0`

Se o mesmo cenário for executado usando o isolamento serializável, o Amazon Redshift encerrará o usuário 2 devido a uma violação serializável e retornará um erro `1023`. Para obter mais informações, consulte Solucionar problemas de erros de isolamento serializável. Nesse caso, somente o usuário 1 pode confirmar com sucesso.

Considerações

Ao usar os níveis de isolamento no Amazon Redshift, pense no seguinte:

- Consulte a visualização do catálogo STV_DB_ISOLATION_LEVEL para visualizar o nível de isolamento que seu banco de dados está usando. Para obter mais informações, consulte STV_DB_ISOLATION_LEVEL.

- Consulte a visualização PG_DATABASE_INFO para ver quantas transações simultâneas são aceitas em seu banco de dados. Para obter mais informações, consulte PG_DATABASE_INFO.
- As tabelas de catálogo de sistema (PG) e outras tabelas de sistema do Amazon Redshift não são bloqueadas em uma transação. Portanto, alterações nos objetos do banco de dados decorrentes de operações DDL e TRUNCATE são visíveis na confirmação para quaisquer transações simultâneas.

  Por exemplo, suponha que a tabela A exista no banco de dados quando duas transações simultâneas, T1 e T2, começam. Suponha que T2 retorna uma lista de tabelas selecionando na tabela de catálogo PG_TABLES. Em seguida, T1 descarta a tabela A e confirmações e, em seguida, T2 lista as tabelas novamente. A tabela A agora não está mais listada. Se T2 tentar consultar a tabela descartada, o Amazon Redshift retornará um erro "relação não existe". A consulta de catálogo que retorna a lista de tabelas para T2 ou verifica se a tabela A existe não está sujeita às mesmas regras de isolamento que as operações realizadas nas tabelas do usuário.

  Transações para atualizações dessas tabelas são executadas em um modo de isolamento de leitura confirmada.
- As tabelas de catálogo com prefixo PG não aceitam o isolamento SNAPSHOT.

Operações de gravação e de leitura/gravação

Você pode gerenciar o comportamento específico de operações simultâneas de gravação decidindo quando e como executar diferentes tipos de comandos. Os seguintes comandos são relevantes para esta discussão:

- Comandos COPY, que executam carregamentos (iniciais ou incrementais)
- Comandos INSERT que anexam uma ou mais linhas de cada vez
- Comandos UPDATE, que modificam linhas existentes
- Comandos DELETE, que removem linhas

As operações COPY e INSERT são operações de gravação pura. As operações DELETE e UPDATE são operações de leitura/gravação (para que as linhas sejam excluídas ou atualizadas, elas devem ser lidas primeiro). Os resultados de operações simultâneas de gravação dependem dos comandos específicos que estão sendo executados simultaneamente.

Operações UPDATE e DELETE comportam-se de maneira diferente, pois elas dependem de uma leitura inicial da tabela antes de realizar qualquer gravação. Considerando que transações simultâneas são invisíveis entre si, operações de UPDATE e DELETE devem ler o snapshot dos dados da última confirmação. Quando a primeira operação UPDATE ou delete libera seu bloqueio, a segunda operação UPDATE ou DELETE precisa determinar se os dados com os quais ela vai trabalhar são potencialmente obsoletos. Eles não serão obsoletos, pois a segunda transação não obtém seu snapshot dos dados até depois que a primeira transação tenha liberado seu bloqueio.

Possível situação de deadlock para transações simultâneas de gravação envolvendo várias tabelas

Quando transações envolvem atualizações de mais que uma tabela, sempre existe a possibilidade de que as transações em execução simultânea fiquem com deadlocks quando ambas tentam gravar no mesmo conjunto de tabelas. Uma transação libera todos os seus bloqueios de tabela de uma só vez ao confirmar ou reverter; ela não libera os bloqueios um de cada vez.

Por exemplo, suponha que as transações T1 e T2 comecem aproximadamente ao mesmo tempo. Se T1 começar a gravar na tabela A e T2 começar a gravar na tabela B, as duas transações poderão prosseguir sem conflito. No entanto, se T1 terminar de gravar na tabela A e precisar começar a gravar na tabela B, ela não poderá prosseguir porque T2 ainda manterá o bloqueio na B. Da mesma forma, se T2 terminar de gravar na tabela B e precisar começar a gravar na tabela A, ela não poderá prosseguir porque T1 ainda manterá o bloqueio na A. Como nenhuma das transações pode liberar seus bloqueios até a confirmação de todas as suas operações de gravação, nenhuma transação pode prosseguir. Para evitar esse tipo de deadlock, você precisa programar operações simultâneas de gravação cuidadosamente. Por exemplo, você deve sempre atualizar as tabelas na mesma ordem nas transações e, se estiver especificando bloqueios, bloquear as tabelas na mesma ordem antes de executar qualquer operação DML.

Possível situação de deadlock para transações simultâneas de gravação envolvendo uma tabela

Em um ambiente de isolamento de snapshot, podem ocorrer deadlocks na execução de transações de gravação simultâneas na mesma tabela. O deadlock de isolamento de snapshot ocorre quando declarações INSERT ou COPY simultâneas estão compartilhando um bloqueio e progredindo, e outra declaração precisa realizar uma operação (UPDATE, DELETE, MERGE ou DDL) que exija um bloqueio exclusivo na mesma tabela.

Considere o seguinte cenário:

Transação 1 (T1):

```
INSERT/COPY INTO table_A;
```

Transação 2 (T2):

```
INSERT/COPY INTO table_A;
            <UPDATE/DELETE/MERGE/DDL statement> table_A
```

Um deadlock pode ocorrer quando várias transações com operações INSERT ou COPY estão sendo executadas simultaneamente na mesma tabela com um bloqueio compartilhado e uma dessas transações segue sua operação de gravação pura com uma operação que exija um bloqueio exclusivo, como uma declaração UPDATE, MERGE, DELETE ou DDL.

Para evitar o deadlock nessas situações, é possível separar declarações que exigem um bloqueio exclusivo (declarações UPDATE/MERGE/DELETE/DDL) para uma transação diferente, de modo que quaisquer declarações INSERT/COPY possam progredir simultaneamente e as declarações que exijam bloqueios exclusivos possam ser executadas depois delas. Como alternativa, no caso de transações com declarações INSERT/COPY e MERGE/UPDATE/DELETE na mesma tabela, é possível incluir a lógica de repetição em suas aplicações para contornar possíveis deadlocks.

Exemplos de gravações simultâneas

Os seguintes exemplos de pseudocódigo demonstram como as transações continuam ou aguardam ao serem executadas simultaneamente.

Exemplos de gravação simultânea com isolamento serializável

Operações COPY simultâneas na mesma tabela com isolamento serializável

A transação 1 copia linhas na tabela LISTING:

```
begin;
copy listing from ...;
end;
```

A transação 2 começa simultaneamente em uma sessão separada e tenta copiar mais linhas na tabela LISTING. A transação 2 deve aguardar até que a transação 1 libere o bloqueio de gravação na tabela LISTING para então proceder.

```
begin;
[waits]
copy listing from ;
end;
```

O mesmo comportamento ocorreria se uma ou ambas as transações contivessem um comando INSERT em vez de um comando COPY.

Operações DELETE simultâneas da mesma tabela com isolamento serializável

A transação 1 exclui linhas de uma tabela:

```
begin;
delete from listing where ...;
end;
```

A transação 2 começa simultaneamente e tenta excluir linhas da mesma tabela. Ela terá êxito, pois ela espera até que a transação 1 seja concluída antes de tentar excluir linhas.

```
begin
[waits]
delete from listing where ;
end;
```

O mesmo comportamento ocorreria se uma ou ambas as transações contivessem um comando UPDATE na mesma tabela em vez de um comando DELETE.

Transações simultâneas com uma combinação de operações de leitura e de gravação com isolamento serializável

Neste exemplo, a transação 1 exclui linhas da tabela USERS, recarrega a tabela, executa uma consulta COUNT(*) e ANALYZE antes de confirmar:

```
begin;
delete one row from USERS table;
copy ;
select count(*) from users;
analyze ;
end;
```

Enquanto isso, a transação 2 começa. Essa transação tenta copiar linhas adicionais na tabela USERS, analisar a tabela e executar a mesma consulta COUNT (*) da primeira transação:

```
begin;
[waits]
copy users from ...;
select count(*) from users;
analyze;
end;
```

A segunda transação terá êxito, pois ela deve aguardar a conclusão da primeira. Sua consulta COUNT retornará a contagem com base no carregamento que foi concluído.

Exemplos de gravação simultânea com isolamento de snapshot

Operações COPY simultâneas na mesma tabela com isolamento de snapshot

A transação 1 copia linhas na tabela LISTING:

```
begin;
copy listing from ...;
end;
```

A transação 2 começa simultaneamente em uma sessão separada e tenta copiar mais linhas na tabela LISTING. A Transação 2 pode progredir simultaneamente até que qualquer transação precise gravar dados na tabela de destino `listing`, momento em que as transações serão executadas sequencialmente.

```
begin;
//When the COPY statement from T1 needs to write data to the table, the COPY statement
 from T2 waits.
copy listing from ...;
end;
```

O mesmo comportamento ocorreria se uma ou ambas as transações contivessem um comando INSERT em vez de um comando COPY.

Operações DELETE simultâneas da mesma tabela com isolamento de snapshot

As operações simultâneas DELETE ou UPDATE da mesma tabela com isolamento de snapshot são executadas da mesma forma que as operações executadas com isolamento serializável.

Transações simultâneas com uma combinação de operações de leitura e de gravação com isolamento de snapshot

As transações simultâneas que são executadas com combinações de operações com isolamento de snapshot são executadas da mesma forma que as transações com combinações de operações executadas com isolamento serializável.

Solucionar problemas de erros de isolamento serializável

ERROR:1023 DETAIL: Violação de isolamento serializável em uma tabela no Redshift

Quando o Amazon Redshift detecta um erro de isolamento serializável, você vê uma mensagem de erro como a seguir.

```
ERROR:1023 DETAIL: Serializable isolation violation on table in Redshift
```

Para resolver um erro de isolamento serializável, você pode tentar um dos seguintes métodos:

- Tente executar a transação cancelada novamente.

  O Amazon Redshift detectou que um workload simultâneo não é serializável. Ele sugere lacunas na lógica da aplicação, que geralmente podem ser contornadas repetindo a transação que encontrou o erro. Se o problema persistir, tente um dos outros métodos.
- Mova todas as operações que não precisam estar na mesma transação atômica para fora da transação.

  Este método é aplicável quando operações individuais em duas transações diferentes fazem referências cruzadas de maneira que possa afetar os resultados. Por exemplo, as duas sessões a seguir iniciam uma transação.

  ```
  Session1_Redshift=# begin;
  ```

  ```
  Session2_Redshift=# begin;
  ```

  O resultado de uma instrução SELECT em uma transação pode ser afetado por uma instrução INSERT na outra. Ou seja, suponhamos que você esteja executando as seguintes instruções em série, em qualquer ordem. Em todo caso, o resultado é uma das instruções SELECT retornar uma linha a mais do que ocorreria caso as transações fossem executadas simultaneamente. Não há ordem na qual as operações possam ser executadas em série e ainda obter o mesmo resultado

da execução simultânea. Portanto, a última operação executada resulta em um erro de isolamento serializável.

```
Session1_Redshift=# select * from tab1;
Session1_Redshift=# insert into tab2 values (1);
```

```
Session2_Redshift=# insert into tab1 values (1);
Session2_Redshift=# select * from tab2;
```

Muitas vezes, o resultado das instruções SELECT não é relevante. Ou seja, a atomicidade das operações nas transações não é importante. Nesses casos, mova as instruções SELECT para fora das transações, conforme mostrado nos exemplos a seguir.

```
Session1_Redshift=# begin;
Session1_Redshift=# insert into tab1 values (1)
Session1_Redshift=# end;
Session1_Redshift=# select * from tab2;
```

```
Session2_Redshift # select * from tab1;
Session2_Redshift=# begin;
Session2_Redshift=# insert into tab2 values (1)
Session2_Redshift=# end;
```

Nesses exemplos, não há referências cruzadas nas transações. As duas instruções INSERT não afetam uma à outra. Nesses exemplos, há pelo menos uma ordem na qual as transações podem ser executadas em série e obter o mesmo resultado da execução simultânea. Isso significa que as transações são serializáveis.

- Bloqueie todas as tabelas nas sessões para forçar a serialização.

  O comando LOCK bloqueia as operações que podem resultar em erros de isolamento serializado. Ao usar o comando LOCK, faça o seguinte:

  - Bloqueie todas as tabelas afetadas pela transação, incluindo aquelas afetadas por instruções SELECT somente leitura na transação.
  - Bloqueie as tabelas na mesma ordem, independentemente da ordem na qual as operações são executadas.
  - Bloqueie todas as tabelas no início da transação, antes de executar qualquer operação.
- Use o isolamento de snapshot para transações simultâneas

Use um comando ALTER DATABASE com isolamento de snapshot. Para obter mais informações sobre o parâmetro SNAPSHOT para ALTER DATABASE, consulte Parâmetros.

ERROR:1018 DETAIL: A relação não existe

Ao executar operações simultâneas do Amazon Redshift em sessões diferentes, uma mensagem de erro semelhante a seguinte é exibida.

```
ERROR: 1018 DETAIL: Relation does not exist.
```

As transações no Amazon Redshift seguem o isolamento de snapshot. Após o início de uma transação, o Amazon Redshift obtém um snapshot do banco de dados. Para todo o ciclo de vida da transação, a transação opera no estado do banco de dados conforme refletido no snapshot. Se a transação lê de uma tabela que não existe no snapshot, ela lança a mensagem de erro 1018 mostrada anteriormente. Mesmo quando outra transação concorrente cria uma tabela após a transação ter obtido o snapshot, a transação não pode ler a partir da tabela recém-criada.

Para resolver este erro de isolamento de serialização, você pode tentar mover o início da transação para um ponto onde você sabe que a tabela existe.

Se a tabela é criada por outra transação, esse ponto é pelo menos depois que essa transação foi confirmada. Além disso, verifique se nenhuma transação simultânea foi confirmada e possa ter excluído a tabela.

```
session1 = # BEGIN;
session1 = # DROP TABLE A;
session1 = # COMMIT;
```

```
session2 = # BEGIN;
```

```
session3 = # BEGIN;
session3 = # CREATE TABLE A (id INT);
session3 = # COMMIT;
```

```
session2 = # SELECT * FROM A;
```

A última operação executada como operação de leitura pela sessão2 resulta em um erro de isolamento serializável. Esse erro ocorre quando session2 tira um snapshot e a tabela já foi descartada por uma session1 confirmada. Em outras palavras, mesmo que uma sessão3 simultânea tenha criado a tabela, session2 não vê a tabela porque ela não está no snapshot.

Para resolver esse erro, é possível reordenar as sessões da forma a seguir.

```
session1 = # BEGIN;
session1 = # DROP TABLE A;
session1 = # COMMIT;
```

```
session3 = # BEGIN;
session3 = # CREATE TABLE A (id INT);
session3 = # COMMIT;
```

```
session2 = # BEGIN;
session2 = # SELECT * FROM A;
```

Agora, quando session2 tira seu snapshot, session3 já foi confirmada e a tabela está no banco de dados. Session2 pode ler a partir da tabela sem qualquer erro.

Tutorial: Carregar dados do Amazon S3

Neste tutorial, você percorre o processo de carregamento de dados nas tabelas de banco de dados do Amazon Redshift a partir de arquivos de dados em um bucket do Amazon S3 do início ao fim.

Neste tutorial, você faz o seguinte:

- Baixa os arquivos de dados que usam formatos separados por vírgulas (CSV), delimitado por caracteres e de largura fixa.
- Cria um bucket do Amazon S3 e carrega os arquivos de dados para o bucket.
- Inicia um cluster do Amazon Redshift e cria tabelas de banco de dados.
- Usa os comandos COPY para carregar as tabelas dos arquivos de dados no Amazon S3.
- Soluciona erros de carga e modifica os comandos COPY para corrigir os erros.

Pré-requisitos

Você precisa dos seguintes pré-requisitos:

- Uma conta da AWS para iniciar um cluster do Amazon Redshift e criar um bucket no Amazon S3.
- Suas credenciais da AWS (função do IAM) para carregar dados de teste do Amazon S3. Se precisar de um novo perfil do IAM, acesse Criar perfis do IAM.
- Um cliente SQL, como o editor de consulta do console do Amazon Redshift.

Este tutorial foi projetado de maneira que possa ser seguido sozinho. Além deste tutorial, recomendamos concluir os seguintes tutoriais para obter uma compreensão mais completa de como projetar e usar bancos de dados do Amazon Redshift:

- O Guia de conceitos básicos do Amazon Redshift orienta você no processo de criação de um cluster do Amazon Redshift e de carregamento dos dados de exemplo.

Visão geral

Você pode adicionar dados às tabelas do Amazon Redshift usando um comando INSERT ou um comando COPY. Na escala e na velocidade de um data warehouse do Amazon Redshift, o comando COPY é muitas vezes mais rápido e eficiente do que os comandos INSERT.

O comando COPY usa a arquitetura de processamento massivamente paralelo (MPP) do Amazon Redshift para ler e carregar dados em paralelo de várias fontes de dados. Você pode carregar arquivos de dados no Amazon S3, Amazon EMR ou qualquer host remoto acessível por meio de uma conexão Secure Shell (SSH). Ou você pode carregar diretamente de uma tabela do Amazon DynamoDB.

Neste tutorial, você usa o comando COPY para carregar dados do Amazon S3. Muitos dos princípios apresentados aqui também se aplicam ao carregamento de outras fontes de dados.

Para saber mais sobre como usar o comando COPY, consulte estes recursos:

- Práticas recomendadas do Amazon Redshift para carregamento de dados
- Carregar dados do Amazon EMR
- Carregamento de dados de hosts remotos
- Carregar dados de uma tabela do Amazon DynamoDB

Etapa 1: criar um cluster

Se já tiver um cluster que quiser usar, você poderá ignorar esta etapa.

Para os exercícios deste tutorial, você usa um cluster de quatro nós.

Para criar um cluster

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.

   Usando o menu de navegação, escolha Painel de clusters provisionados.

   > **Important**
   >
   > Verifique se o você tem as permissões necessárias para executar as operações de cluster. Para obter informações sobre como conceder as permissões necessárias, consulte Autorizar o Amazon Redshift a acessar serviços da AWS.

2. Na parte superior direita, escolha a região da AWS onde você deseja criar o cluster. Para este tutorial, escolha Oeste dos EUA (Oregon).
3. No menu de navegação, escolha Clusters e Create cluster (Criar cluster). A página Create cluster (Criar cluster) é exibida.
4. Na página Create cluster (Criar cluster), insira parâmetros do cluster. Escolha seus valores para os parâmetros, sem alterar os seguintes valores:

   - Escolha **`dc2.large`** para o tipo de nó.
   - Escolha **`4`** em Number of nodes (Número de nós).
   - Na seção Cluster permissions (Permissões do cluster), escolha uma função do IAM em Available IAM roles (Funções do IAM disponíveis). Essa função deve ser uma que você criou anteriormente e que tem acesso ao Amazon S3. Depois, escolha Associate IAM role (Associar função do IAM) para adicioná-la à lista de Associated IAM roles (Funções do IAM associadas) do cluster.
5. Selecione Criar cluster.

Siga as etapas do Guia de conceitos básicos do Amazon Redshift para se conectar de um cliente SQL ao seu cluster e testar uma conexão. Você não precisa concluir as etapas de Conceitos básicos restantes para criar tabelas, fazer upload de dados e testar consultas de exemplo.

Etapa 2: Fazer download dos arquivos de dados

Nesta etapa, você baixará um conjunto de arquivos de dados de exemplo no computador. Na próxima etapa, você carrega os arquivos em um bucket do Amazon S3.

Para baixar os arquivos de dados

1. Baixe o arquivo compactado: LoadingDataSampleFiles.zip.
2. Extraia os arquivos para uma pasta no computador.
3. Verifique se a pasta contém os arquivos a seguir.

```
customer-fw-manifest
customer-fw.tbl-000
customer-fw.tbl-000.bak
customer-fw.tbl-001
customer-fw.tbl-002
customer-fw.tbl-003
customer-fw.tbl-004
customer-fw.tbl-005
customer-fw.tbl-006
customer-fw.tbl-007
customer-fw.tbl.log
dwdate-tab.tbl-000
dwdate-tab.tbl-001
dwdate-tab.tbl-002
dwdate-tab.tbl-003
dwdate-tab.tbl-004
dwdate-tab.tbl-005
dwdate-tab.tbl-006
dwdate-tab.tbl-007
part-csv.tbl-000
part-csv.tbl-001
part-csv.tbl-002
part-csv.tbl-003
part-csv.tbl-004
part-csv.tbl-005
part-csv.tbl-006
part-csv.tbl-007
```

Etapa 3: Carregar arquivos para um bucket do Amazon S3

Nesta etapa, você cria um bucket do Amazon S3 e carrega os arquivos de dados para o bucket.

Para carregar arquivos para um bucket do Amazon S3

1. Crie um bucket no Amazon S3.

   Para obter mais informações sobre como criar um bucket, consulte Criar um bucket, no Guia do usuário do Amazon Simple Storage Service.

   a. Faça login no Console de gerenciamento da AWS e abra o console do Amazon S3 em https://console.aws.amazon.com/s3/.

   b. Escolha Criar bucket.

   c. Escolha um Região da AWS.

      Crie o bucket na mesma região do cluster. Se o cluster estiver na região Oeste dos EUA (Oregon), escolha US West (Oregon) Region (us-west-2).

   d. Na caixa Nome do bucket da caixa de diálogo Criar um bucket, digite o nome de um bucket.

      O nome de bucket que você escolher deve ser único entre todos os nomes de bucket existentes no Amazon S3. Uma maneira de ajudar a garantir que os nomes dos buckets sejam exclusivos é prefixá-los com o nome da sua organização. Os nomes de bucket devem estar em conformidade com determinadas regras. Para obter mais informações, acesse Restrições e limitações de bucket no Guia do usuário do Amazon Simple Storage Service.

   e. Escolha os padrões recomendados para o restante das opções.

   f. Escolha Criar bucket.

      Quando o Amazon S3 cria seu bucket com sucesso, o console exibe seu bucket vazio no painel Buckets.

2. Crie uma pasta.

   a. Escolha o nome do novo bucket.

   b. Selecione o botão Criar pasta.

   c. Nomeie a nova pasta como **`load`**.

> **Note**
>
> O bucket que você criou não está em um sandbox. Neste exercício, você adiciona objetos a um bucket real. Um valor nominal é cobrado pelo tempo em que você armazena os objetos no bucket. Para obter mais informações sobre preços do Amazon S3, consulte Preço do Amazon S3.

3. Carregue arquivos de dados para o novo bucket do Amazon S3.

   a. Escolha o nome da pasta de dados.

   b. No assistente de upload, escolha Adicionar arquivos.

      Siga as instruções do console do Amazon S3 para carregar todos os arquivos que você baixou e extraiu.

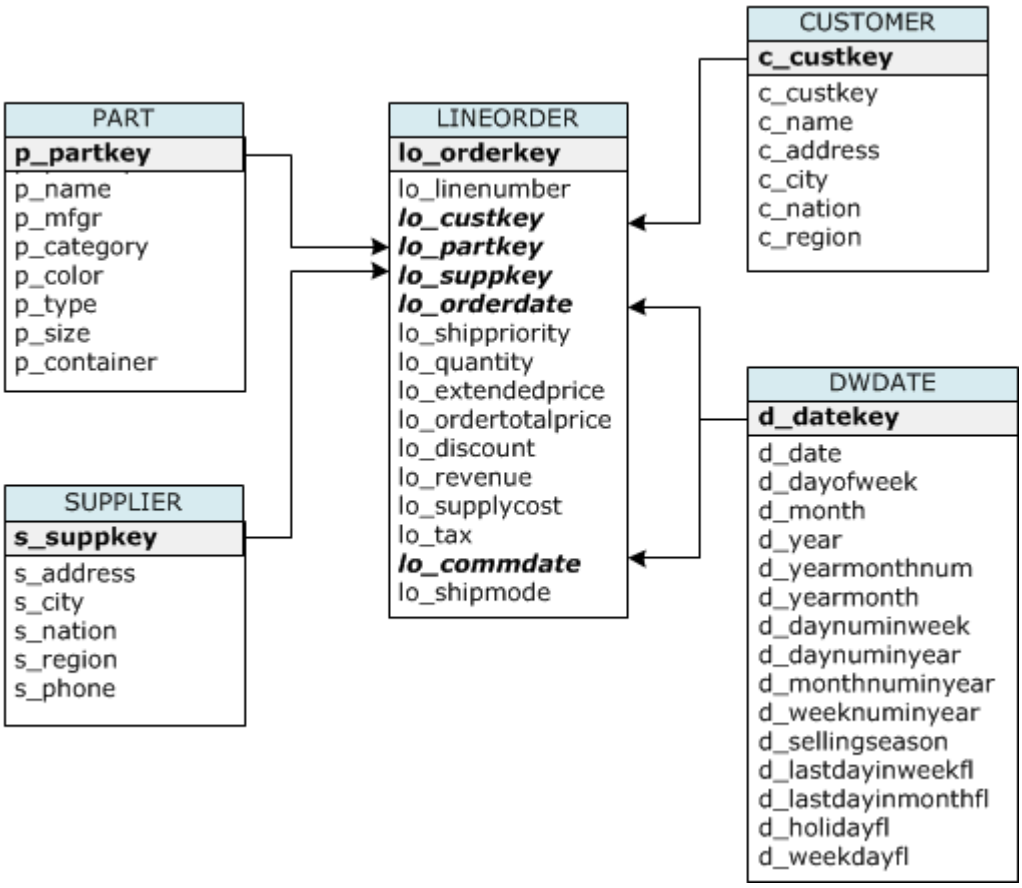
   c. Escolha Carregar.

Credenciais do usuário

O comando COPY do Amazon Redshift deve ter acesso para ler os objetos de arquivo no bucket do Amazon S3. Se você usar as mesmas credenciais de usuário para criar o bucket do Amazon S3 e para executar o comando COPY do Amazon Redshift, o comando COPY tem todas as permissões necessárias. Se quiser usar credenciais de usuário diferentes, você pode conceder acesso usando os controles de acesso do Amazon S3. O comando COPY do Amazon Redshift requer pelo menos as permissões ListBucket e GetObject para acessar os objetos de arquivo no bucket do Amazon S3. Para obter mais informações sobre como controlar o acesso aos recursos do Amazon S3, consulte Gerenciar permissões de acesso aos recursos do Amazon S3.

Etapa 4: Criar as tabelas de exemplo

Neste tutorial, você usará um conjunto de tabelas baseadas no esquema Star Schema Benchmark (SSB). O diagrama a seguir mostra o modelo de dados SSB.



As tabelas SSB talvez já existam no banco de dados atual. Nesse caso, descarte as tabelas para removê-las do banco de dados antes de criá-las usando os comandos CREATE TABLE na próxima etapa. As tabelas usadas neste tutorial podem ter atributos diferentes das tabelas existentes.

Para criar as tabelas de exemplo

1. Para remover tabelas SSB, execute os comandos a seguir no cliente SQL.

```
drop table part cascade;
drop table supplier;
drop table customer;
drop table dwdate;
drop table lineorder;
```

2. Execute os comandos CREATE TABLE a seguir em seu cliente SQL.

```
CREATE TABLE part
(
  p_partkey     INTEGER NOT NULL,
  p_name        VARCHAR(22) NOT NULL,
```

```
  p_mfgr        VARCHAR(6),
  p_category    VARCHAR(7) NOT NULL,
  p_brand1      VARCHAR(9) NOT NULL,
  p_color       VARCHAR(11) NOT NULL,
  p_type        VARCHAR(25) NOT NULL,
  p_size        INTEGER NOT NULL,
  p_container   VARCHAR(10) NOT NULL
);

CREATE TABLE supplier
(
  s_suppkey   INTEGER NOT NULL,
  s_name      VARCHAR(25) NOT NULL,
  s_address   VARCHAR(25) NOT NULL,
  s_city      VARCHAR(10) NOT NULL,
  s_nation    VARCHAR(15) NOT NULL,
  s_region    VARCHAR(12) NOT NULL,
  s_phone     VARCHAR(15) NOT NULL
);

CREATE TABLE customer
(
  c_custkey      INTEGER NOT NULL,
  c_name         VARCHAR(25) NOT NULL,
  c_address      VARCHAR(25) NOT NULL,
  c_city         VARCHAR(10) NOT NULL,
  c_nation       VARCHAR(15) NOT NULL,
  c_region       VARCHAR(12) NOT NULL,
  c_phone        VARCHAR(15) NOT NULL,
  c_mktsegment   VARCHAR(10) NOT NULL
);

CREATE TABLE dwdate
(
  d_datekey            INTEGER NOT NULL,
  d_date               VARCHAR(19) NOT NULL,
  d_dayofweek          VARCHAR(10) NOT NULL,
  d_month              VARCHAR(10) NOT NULL,
  d_year               INTEGER NOT NULL,
  d_yearmonthnum       INTEGER NOT NULL,
  d_yearmonth          VARCHAR(8) NOT NULL,
  d_daynuminweek       INTEGER NOT NULL,
  d_daynuminmonth      INTEGER NOT NULL,
  d_daynuminyear       INTEGER NOT NULL,
```

```
  d_monthnuminyear     INTEGER NOT NULL,
  d_weeknuminyear      INTEGER NOT NULL,
  d_sellingseason      VARCHAR(13) NOT NULL,
  d_lastdayinweekfl    VARCHAR(1) NOT NULL,
  d_lastdayinmonthfl   VARCHAR(1) NOT NULL,
  d_holidayfl          VARCHAR(1) NOT NULL,
  d_weekdayfl          VARCHAR(1) NOT NULL
);
CREATE TABLE lineorder
(
  lo_orderkey          INTEGER NOT NULL,
  lo_linenumber        INTEGER NOT NULL,
  lo_custkey           INTEGER NOT NULL,
  lo_partkey           INTEGER NOT NULL,
  lo_suppkey           INTEGER NOT NULL,
  lo_orderdate         INTEGER NOT NULL,
  lo_orderpriority     VARCHAR(15) NOT NULL,
  lo_shippriority      VARCHAR(1) NOT NULL,
  lo_quantity          INTEGER NOT NULL,
  lo_extendedprice     INTEGER NOT NULL,
  lo_ordertotalprice   INTEGER NOT NULL,
  lo_discount          INTEGER NOT NULL,
  lo_revenue           INTEGER NOT NULL,
  lo_supplycost        INTEGER NOT NULL,
  lo_tax               INTEGER NOT NULL,
  lo_commitdate        INTEGER NOT NULL,
  lo_shipmode          VARCHAR(10) NOT NULL
);
```

Etapa 5: Executar os comandos COPY

Você executa os comandos COPY para carregar cada uma das tabelas no esquema SSB. Os exemplos de comando COPY demonstram como carregar de formatos de arquivo diferentes usando várias opções de comando COPY e solucionando erros de carga.

Sintaxe do comando COPY

A seguir, a sintaxe básica do comando COPY.

```
COPY table_name [ column_list ] FROM data_source CREDENTIALS access_credentials
 [options]
```

Para executar um comando COPY, forneça os valores a seguir.

Nome da tabela

A tabela de destino do comando COPY. A tabela já deve existir no banco de dados. A tabela pode ser temporária ou persistente. O comando COPY acrescenta os novos dados de entrada a todas as linhas existentes na tabela.

Lista de colunas

Por padrão, COPY carrega campos dos dados de origem para as colunas da tabela na ordem. Você pode especificar uma lista de colunas, uma lista separada por vírgulas de nomes de coluna, a fim de mapear campos de dados para colunas específicas. Você não usa listas de colunas neste tutorial. Para obter mais informações, consulte Column List na referência do comando COPY.

Fonte de dados

Você pode usar o comando COPY para carregar dados de um bucket do Amazon S3, um cluster do Amazon EMR, um host remoto usando uma conexão SSH ou uma tabela do Amazon DynamoDB. Para este tutorial, você carrega de arquivos de dados em um bucket do Amazon S3. Ao carregar do Amazon S3, você deve fornecer o nome do bucket e a localização dos arquivos de dados. Para fazer isso, forneça o caminho de um objeto para os arquivos de dados ou a localização de um arquivo manifesto que lista explicitamente cada arquivo de dados e sua localização.

- Prefixo de chaves

  Um objeto armazenado no Amazon S3 é identificado exclusivamente por uma chave de objeto, que inclui o nome do bucket, os nomes das pastas, se houver, e o nome do objeto. Um prefixo de chaves se refere a um conjunto de objetos com o mesmo prefixo. O caminho do objeto é um prefixo de chaves usado pelo comando COPY para carregar todos os objetos que compartilham o prefixo de chaves. Por exemplo, o prefixo de chaves `custdata.txt` pode referenciar um único arquivo ou um conjunto de arquivos, inclusive `custdata.txt.001custdata.txt.002` e assim por diante.
- Arquivo manifesto

  Em alguns casos, pode ser necessário carregar os arquivos com prefixos diferentes, por exemplo, de vários buckets ou pastas. Em outros, pode ser necessário excluir arquivos que compartilham um prefixo. Nesses casos, é possível usar um arquivo manifesto. Um arquivo manifesto lista cada arquivo de carga e a chave de objeto exclusiva. Você usa um arquivo manifesto para carregar a tabela PART posteriormente neste tutorial.

Credenciais

Para acessar os recursos da AWS que contêm os dados a serem carregados, você deve fornecer as credenciais de acesso da AWS a um usuário com privilégios suficientes. Essas credenciais incluem um nome do recurso da Amazon (ARN) da função do IAM. Para carregar dados do Amazon S3, as credenciais devem incluir as permissões ListBucket e GetObject. Serão necessárias outras credenciais, se os dados estiverem criptografados. Para obter mais informações, consulte Parâmetros de autorização na referência do comando COPY. Para obter mais informações sobre como gerenciar o acesso, consulte Gerenciar permissões de acesso aos seus recursos do Amazon S3.

Opções

Você pode especificar vários parâmetros com o comando COPY para especificar formatos de arquivo, gerenciar formatos de dados, gerenciar erros e controlar outros recursos. Neste tutorial, você usa as seguintes opções e os seguintes recursos do comando COPY:

- Prefixo de chaves

  Para obter informações sobre como carregar de vários arquivos especificando um prefixo de chave, consulte Carregar a tabela PART usando NULL AS.

- Formato CSV

  Para obter informações sobre como carregar dados que estão no formato CSV, consulte Carregar a tabela PART usando NULL AS.

- NULL AS

  Para obter informações sobre como carregar PART usando a opção NULL AS, consulte Carregar a tabela PART usando NULL AS.

- Formato delimitado por caracteres

  Para obter informações sobre como usar a opção DELIMITER, consulte As opções DELIMITER e REGION.

- REGION

  Para obter informações sobre como usar a opção REGION, consulte As opções DELIMITER e REGION.

- Largura de formato fixo

Para obter informações sobre como carregar a tabela CUSTOMER de dados de largura fixa, consulte Carregar a tabela CUSTOMER usando MANIFEST.

- MAXERROR

  Para obter informações sobre como usar a opção MAXERROR, consulte Carregar a tabela CUSTOMER usando MANIFEST.

- ACCEPTINVCHARS

  Para obter informações sobre como usar a opção ACCEPTINVCHARS, consulte Carregar a tabela CUSTOMER usando MANIFEST.

- MANIFEST

  Para obter informações sobre como usar a opção MANIFEST, consulte Carregar a tabela CUSTOMER usando MANIFEST.

- DATEFORMAT

  Para obter informações sobre como usar a opção DATEFORMAT, consulte Carregar a tabela DWDATE usando DATEFORMAT.

- GZIP, LZOP e BZIP2

  Para obter informações sobre como compactar arquivos, consulte Carregar vários arquivos de dados.

- COMPUPDATE

  Para obter informações sobre como usar a opção COMPUPDATE, consulte Carregar vários arquivos de dados.

- Vários arquivos

  Para obter informações sobre como carregar vários arquivos, consulte Carregar vários arquivos de dados.

Carregar as tabelas SSB

Você usa os comandos COPY a seguir para carregar cada uma das tabelas no esquema SSB. O comando para cada tabela demonstra opções de COPY e técnicas para solução de problemas diferentes.

Para carregar as tabelas SSB, siga estas etapas:

1. Substituir o nome do bucket e as credenciais da AWS
2. Carregar a tabela PART usando NULL AS
3. Carregar a tabela CUSTOMER usando MANIFEST
4. Carregar a tabela DWDATE usando DATEFORMAT

Substituir o nome do bucket e as credenciais da AWS

Os comandos COPY neste tutorial são apresentados no formato a seguir.

```
copy table from 's3://<your-bucket-name>/load/key_prefix'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
options;
```

Para cada comando COPY, faça o seguinte:

1. Substitua *<your-bucket-name>* pelo nome de um bucket na mesma região do cluster.

   Essa etapa pressupõe que o bucket e o cluster estejam na mesma região. Você também pode especificar a região usando a opção REGION com o comando COPY.
2. Substitua *<aws-account-id>* e *<role-name>* por sua Conta da AWS e função do IAM. O segmento da string de credenciais entre aspas não deve conter espaços ou quebras de linha. O formato do ARN pode ser levemente diferente do formato da amostra. É recomendável copiar o ARN do perfil no console do IAM, para garantir que ele esteja correto, ao executar os comandos COPY.

Carregar a tabela PART usando NULL AS

Nesta etapa, você usa as opções CSV e NULL AS para carregar a tabela PART.

O comando COPY pode carregar dados de vários arquivos em paralelo, o que é muito mais rápido do que carregar de um único arquivo. Para demonstrar esse princípio, os dados de cada tabela neste tutorial estão divididos em oito arquivos, mesmo que os arquivos sejam muito pequenos. Em uma etapa posterior, você compara a diferença de tempo entre o carregamento de um único arquivo e carregar de vários arquivos. Para obter mais informações, consulte Carregar arquivos de dados.

Prefixo de chaves

Você pode carregar de vários arquivos especificando um prefixo de chaves para o conjunto de arquivos ou listando explicitamente os arquivos em um arquivo manifesto. Nesta etapa, você usa um prefixo de chaves. Em uma etapa posterior, você usa um arquivo manifesto. O prefixo de chaves `'s3://amzn-s3-demo-bucket/load/part-csv.tbl'` carrega o conjunto de arquivos a seguir na pasta `load`.

```
part-csv.tbl-000
part-csv.tbl-001
part-csv.tbl-002
part-csv.tbl-003
part-csv.tbl-004
part-csv.tbl-005
part-csv.tbl-006
part-csv.tbl-007
```

Formato CSV

CSV, que significa comma separated values, ou valores separados por vírgulas, é um formato comum usado para importar e exportar dados da planilha. O CSV é mais flexível que o formato delimitado por vírgulas porque permite incluir strings entre aspas nos campos. O caractere de aspa padrão para o formato COPY from CSV é uma aspa dupla ("), mas você pode especificar outro caractere de aspa usando a opção QUOTE AS. Ao usar aspas dentro do campo, escape o caractere com aspas adicionais.

O seguinte trecho de um arquivo de dados formatado em CSV para a tabela PART mostra strings entre aspas duplas (`"LARGE ANODIZED BRASS"`). Ele também mostra uma string entre aspas duplas dentro da string entre aspas (`"MEDIUM ""BURNISHED"" TIN"`).

```
15,dark sky,MFGR#3,MFGR#47,MFGR#3438,indigo,"LARGE ANODIZED BRASS",45,LG CASE
22,floral beige,MFGR#4,MFGR#44,MFGR#4421,medium,"PROMO, POLISHED BRASS",19,LG DRUM
23,bisque slate,MFGR#4,MFGR#41,MFGR#4137,firebrick,"MEDIUM ""BURNISHED"" TIN",42,JUMBO
 JAR
```

Os dados da tabela PART contêm caracteres que causam uma falha no comando COPY. Neste exercício, você soluciona problemas e corrige-os.

Para carregar dados que estejam em formato CSV, adicione `csv` ao comando COPY. Execute o comando a seguir para carregar a tabela PART.

```
copy part from 's3://<your-bucket-name>/load/part-csv.tbl'
```

```
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
csv;
```

Você poderá receber uma mensagem de erro semelhante à mensagem a seguir.

```
An error occurred when executing the SQL command:
copy part from 's3://amzn-s3-demo-bucket/load/part-csv.tbl'
credentials' ...

ERROR: Load into table 'part' failed.  Check 'stl_load_errors' system table for
 details. [SQL State=XX000]

Execution time: 1.46s

1 statement(s) failed.
1 statement(s) failed.
```

Para obter mais informações sobre o erro, consulte a tabela STL_LOAD_ERRORS. A consulta a seguir usar a função SUBSTRING para encurtar colunas para fins de legibilidade e usa LIMIT 10 para reduzir o número de linhas retornadas. Você pode ajustar os valores em `substring(filename,22,25)` para permitir o comprimento do nome do bucket.

```
select query, substring(filename,22,25) as filename,line_number as line,
substring(colname,0,12) as column, type, position as pos, substring(raw_line,0,30) as
 line_text,
substring(raw_field_value,0,15) as field_text,
substring(err_reason,0,45) as reason
from stl_load_errors
order by query desc
limit 10;
```

```
 query  |    filename      | line |  column   |    type    | pos |
--------+------------------------+-----------+------------+------------+-----+----
 333765 | part-csv.tbl-000 |    1 |           |            |   0 |

 line_text        | field_text |                   reason
------------------+------------+-----------------------------------------------
 15,NUL next,     |            | Missing newline: Unexpected character 0x2c f
```

NULL AS

Os arquivos de dados `part-csv.tbl` usam o caractere terminador NUL (`\x000` ou `\x0`) para indicar valores NULL.

> **Note**
>
> Apesar da ortografia muito semelhante, NUL e NULL não são iguais. NUL é um caractere UTF-8 com codepoint `x000` normalmente usado para indicar End Of Record (EOR – Fim do registro). NULL é um valor SQL que representa uma ausência de dados.

Por padrão, COPY trata um caractere terminador NUL como um caractere EOR e encerra o registro, o que normalmente acarreta em resultados inesperados ou em um erro. Não existe um único método padrão para indicar NULL em dados de texto. Assim, com a opção NULL AS do comando COPY, é possível especificar qual caractere deve ser substituído por NULL ao carregar a tabela. Neste exemplo, você deseja que COPY trate o caractere terminador NUL como um valor NULL.

> **Note**
>
> A coluna da tabela que recebe o valor NULL deve ser configurada como anulável. Ou seja, ela não deve incluir a restrição NOT NULL na especificação CREATE TABLE.

Para carregar PART usando a opção NULL AS, execute o comando COPY a seguir.

```
copy part from 's3://<your-bucket-name>/load/part-csv.tbl'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
csv
null as '\000';
```

Para verificar se o COPY carregou valores NULL, execute o comando a seguir para selecionar somente as linhas que contenham NULL.

```
select p_partkey, p_name, p_mfgr, p_category from part where p_mfgr is null;
```

```
 p_partkey |  p_name  | p_mfgr | p_category
-----------+----------+--------+------------
        15 | NUL next |        | MFGR#47
        81 | NUL next |        | MFGR#23
       133 | NUL next |        | MFGR#44
```

```
(2 rows)
```

As opções DELIMITER e REGION

As opções DELIMITER e REGION são importantes para entender como carregar os dados.

Formato delimitado por caracteres

Os campos em um arquivo delimitado por caracteres são separados por um caractere específico, como uma barra (|), uma vírgula (,) ou uma tabulação (\t). Os arquivos delimitados por caractere podem usar qualquer caractere ASCII único, inclusive um dos caracteres ASCII não imprimíveis, como o delimitador. Você especifica o caractere delimitador usando a opção DELIMITER. O delimitador padrão é um caractere de barra (|).

O trecho a seguir dos dados da tabela SUPPLIER usa o formato delimitado por barras.

```
1|1|257368|465569|41365|19950218|2-HIGH|0|17|2608718|9783671|4|2504369|92072|2|
19950331|TRUCK
1|2|257368|201928|8146|19950218|2-HIGH|0|36|6587676|9783671|9|5994785|109794|6|
19950416|MAIL
```

REGION

Sempre que possível, é recomendável localizar os dados de carga na mesma região da AWS do cluster do Amazon Redshift. Se os dados e o cluster estiverem na mesma região, você reduzirá a latência e evitará custos de transferência de dados entre regiões. Para obter mais informações, consulte Práticas recomendadas do Amazon Redshift para carregamento de dados.

Se for necessário carregar dados de uma região da AWS diferente, use a opção REGION para especificar a região da AWS na qual os dados de carga estão localizados. Se você especificar uma região, todos os dados de carga, incluindo os arquivos de manifesto, deverão estar na região especificada. Para obter mais informações, consulte REGION.

Por exemplo, se o cluster estiver na região Leste dos EUA (Norte da Virgínia) e o bucket do Amazon S3 estiver na região Oeste dos EUA (Oregon), o comando COPY a seguir mostrará como carregar a tabela SUPPLIER usando dados delimitados por pipe.

```
copy supplier from 's3://amzn-s3-demo-bucket/ssb/supplier.tbl'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
delimiter '|'
gzip
```

```
region 'us-west-2';
```

Carregar a tabela CUSTOMER usando MANIFEST

Nesta etapa, você usa as opções FIXEDWIDTH, MAXERROR, ACCEPTINVCHARS e MANIFEST para carregar a tabela CUSTOMER.

Os dados de exemplo deste exercício contêm caracteres que causam erros quando COPY tenta carregá-los. Você usa a opção MAXERRORS e a tabela do sistema STL_LOAD_ERRORS para solucionar erros de carga e usa as opções ACCEPTINVCHARS e MANIFEST para eliminar os erros.

Formato de largura fixa

O formato de largura fixa define cada campo como um número de caracteres fixo, em vez de separar campos com um delimitador. O trecho a seguir dos dados da tabela CUSTOMER usa o formato de largura fixa.

```
1   Customer#000000001   IVhzIApeRb            MOROCCO  0MOROCCO  AFRICA      25-705
2   Customer#000000002   XSTf4,NCwDVaWNe6tE    JORDAN   6JORDAN   MIDDLE EAST 23-453
3   Customer#000000003   MG9kdTD               ARGENTINA5ARGENTINAAMERICA     11-783
```

A ordem dos pares de rótulo/largura deve corresponder exatamente à ordem das colunas da tabela. Para obter mais informações, consulte FIXEDWIDTH.

A seguir, a string de especificação de largura fixa dos dados da tabela CUSTOMER.

```
fixedwidth 'c_custkey:10, c_name:25, c_address:25, c_city:10, c_nation:15,
c_region :12, c_phone:15,c_mktsegment:10'
```

Para carregar a tabela CUSTOMER de dados de largura fixa, execute o comando a seguir.

```
copy customer
from 's3://<your-bucket-name>/load/customer-fw.tbl'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
fixedwidth 'c_custkey:10, c_name:25, c_address:25, c_city:10, c_nation:15,
 c_region :12, c_phone:15,c_mktsegment:10';
```

Você deve receber uma mensagem de erro, semelhante à mensagem a seguir.

```
An error occurred when executing the SQL command:
copy customer
```

```
from 's3://amzn-s3-demo-bucket/load/customer-fw.tbl'
credentials'...

ERROR: Load into table 'customer' failed.  Check 'stl_load_errors' system table for
 details. [SQL State=XX000]

Execution time: 2.95s

1 statement(s) failed.
```

MAXERROR

Por padrão, a primeira vez em que COPY encontra um erro, o comando falha e retorna uma mensagem de erro. Para economizar tempo durante testes, você pode usar a opção MAXERROR para instruir COPY a ignorar um número especificado de erros antes de falhar. Como é possível esperar erros na primeira vez que você testar o carregamento dos dados da tabela CUSTOMER, adicione `maxerror 10` ao comando COPY.

Para testar usando as opções FIXEDWIDTH e MAXERROR, execute o comando a seguir.

```
copy customer
from 's3://<your-bucket-name>/load/customer-fw.tbl'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
fixedwidth 'c_custkey:10, c_name:25, c_address:25, c_city:10, c_nation:15,
 c_region :12, c_phone:15,c_mktsegment:10'
maxerror 10;
```

Desta vez, em vez de uma mensagem de erro, você recebe uma mensagem de aviso semelhante à que se segue.

```
Warnings:
Load into table 'customer' completed, 112497 record(s) loaded successfully.
Load into table 'customer' completed, 7 record(s) could not be loaded.  Check
 'stl_load_errors' system table for details.
```

O aviso indica que COPY encontrou sete erros. Para verificar erros, consulte a tabela STL_LOAD_ERRORS, conforme mostrado no exemplo a seguir.

```
select query, substring(filename,22,25) as filename,line_number as line,
substring(colname,0,12) as column, type, position as pos, substring(raw_line,0,30) as
 line_text,
substring(raw_field_value,0,15) as field_text,
```

```
substring(err_reason,0,45) as error_reason
from stl_load_errors
order by query desc, filename
limit 7;
```

Os resultados da consulta STL_LOAD_ERRORS devem ser semelhantes aos resultados a seguir.

```
 query  |         filename          | line |  column   |    type    | pos |
 line_text           | field_text |              error_reason
--------+---------------------------+------+-----------
+------------+-----+-------------------------------+------------
+-------------------------------------------
 334489 | customer-fw.tbl.log       |    2 | c_custkey | int4       |  -1 | customer-
fw.tbl               | customer-f | Invalid digit, Value 'c', Pos 0, Type: Integ
 334489 | customer-fw.tbl.log       |    6 | c_custkey | int4       |  -1 | Complete
                  | Complete   | Invalid digit, Value 'C', Pos 0, Type: Integ
 334489 | customer-fw.tbl.log       |    3 | c_custkey | int4       |  -1 | #Total rows
                  | #Total row | Invalid digit, Value '#', Pos 0, Type: Integ
 334489 | customer-fw.tbl.log       |    5 | c_custkey | int4       |  -1 | #Status
                  | #Status    | Invalid digit, Value '#', Pos 0, Type: Integ
 334489 | customer-fw.tbl.log       |    1 | c_custkey | int4       |  -1 | #Load file
                  | #Load file | Invalid digit, Value '#', Pos 0, Type: Integ
 334489 | customer-fw.tbl000        |    1 | c_address | varchar    |  34 | 1
 Customer#000000001  | .Mayag.ezR | String contains invalid or unsupported UTF8
 334489 | customer-fw.tbl000        |    1 | c_address | varchar    |  34 | 1
 Customer#000000001  | .Mayag.ezR | String contains invalid or unsupported UTF8
(7 rows)
```

Examinando os resultados, você pode ver que existem duas mensagens na coluna `error_reasons`:

- `Invalid digit, Value '#', Pos 0, Type: Integ`

  Esses erros são causados pelo arquivo `customer-fw.tbl.log`. O problema é que se trata de um arquivo de log, não um arquivo de dados, e não deve ser carregado. Você pode usar um arquivo manifesto para evitar carregar o arquivo errado.

- `String contains invalid or unsupported UTF8`

  O tipo de dados VARCHAR dá suporte a caracteres UTF-8 multibyte até três bytes. Se os dados de carga contiverem caracteres incompatíveis ou inválidos, você poderá usar a opção

ACCEPTINVCHARS para substituir cada caractere inválido por um caractere alternativo especificado.

Outro problema com a carga é mais difícil de detectar — a carga produziu resultados inesperados. Para investigar esse problema, execute o comando a seguir para consultar a tabela CUSTOMER.

```
select c_custkey, c_name, c_address
from customer
order by c_custkey
limit 10;
```

```
 c_custkey |          c_name           |         c_address
-----------+---------------------------+---------------------------
         2 | Customer#000000002        | XSTf4,NCwDVaWNe6tE
         2 | Customer#000000002        | XSTf4,NCwDVaWNe6tE
         3 | Customer#000000003        | MG9kdTD
         3 | Customer#000000003        | MG9kdTD
         4 | Customer#000000004        | XxVSJsL
         4 | Customer#000000004        | XxVSJsL
         5 | Customer#000000005        | KvpyuHCplrB84WgAi
         5 | Customer#000000005        | KvpyuHCplrB84WgAi
         6 | Customer#000000006        | sKZz0CsnMD7mp4Xd0YrBvx
         6 | Customer#000000006        | sKZz0CsnMD7mp4Xd0YrBvx
(10 rows)
```

As linhas devem ser exclusivas, mas há duplicações.

Outra maneira de verificar se há resultados inesperados é consultar o número de linhas que foram carregadas. Em nosso caso, 100.000 linhas devem ter sido carregadas, mas a mensagem de carga relatou o carregamento de 112.497. As linhas extra foram carregadas porque COPY carregou um arquivo estranho, `customer-fw.tbl0000.bak`.

Neste exercício, você usa um arquivo manifesto para evitar carregar os arquivos errados.

ACCEPTINVCHARS

Por padrão, quando encontra um caractere que não seja compatível com o tipo de dados da coluna, COPY ignora a linha retorna um erro. Para obter informações sobre caracteres UTF-8 inválidos, consulte Erros no carregamento de caracteres multibyte.

Você pode usar a opção MAXERRORS para ignorar erros e continuar carregando, a consulta STL_LOAD_ERRORS para localizar os caracteres inválidos e corrigir os arquivos de dados. Porém, MAXERRORS é mais bem usado na solução de problemas de carga e normalmente não deve ser usado em um ambiente de produção.

A opção ACCEPTINVCHARS normalmente é uma opção melhor para gerenciar caracteres inválidos. ACCEPTINVCHARS orienta COPY a substituir cada caractere inválido por um caractere válido especificado e continuar a operação de carga. Você pode especificar qualquer caractere ASCII válido, exceto NULL, como o caractere substituto. O caractere de substituição padrão é um ponto de interrogação (?). COPY substitui caracteres multibyte por uma string de substituição de comprimento igual. Por exemplo, um caractere de 4 bytes seria substituído por `'????'`.

COPY retorna o número de linhas que continham caracteres UTF-8 errados. Ele também adiciona uma entrada à tabela STL_REPLACEMENTS para cada linha afetada, até um máximo de 100 linhas por fatia de nó. Os caracteres UTF-8 inválidos adicionais também são substituídos, mas esses eventos substitutos não são registrados.

ACCEPTINVCHARS só é válido para colunas VARCHAR.

Para esta etapa, você adiciona o ACCEPTINVCHARS com o caractere de substituição `'^'`.

MANIFEST

Quando você COPY do Amazon S3 usando um prefixo das chave, há o risco de carregar tabelas indesejadas. Por exemplo, a pasta `'s3://amzn-s3-demo-bucket/load/` contém oito arquivos de dados que compartilham o prefixo de chaves `customer-fw.tbl`: `customer-fw.tbl0000`, `customer-fw.tbl0001` e assim por diante. Porém, a mesma pasta também contém arquivos estranhos `customer-fw.tbl.log` e `customer-fw.tbl-0001.bak`.

Para garantir que todos os arquivos corretos, e somente os arquivos corretos, sejam carregados, use um arquivo de manifesto. Manifesto é um arquivo de texto em formato JSON que lista explicitamente a chave de objeto exclusiva para cada arquivo de origem a ser carregado. Os objetos de arquivo podem estar em pastas ou em buckets diferentes, mas devem estar na mesma região. Para obter mais informações, consulte MANIFEST.

A seguir, o texto `customer-fw-manifest`.

```
{
  "entries": [
    {"url":"s3://<your-bucket-name>/load/customer-fw.tbl-000"},
    {"url":"s3://<your-bucket-name>/load/customer-fw.tbl-001"},
```

```
    {"url":"s3://<your-bucket-name>/load/customer-fw.tbl-002"},
    {"url":"s3://<your-bucket-name>/load/customer-fw.tbl-003"},
    {"url":"s3://<your-bucket-name>/load/customer-fw.tbl-004"},
    {"url":"s3://<your-bucket-name>/load/customer-fw.tbl-005"},
    {"url":"s3://<your-bucket-name>/load/customer-fw.tbl-006"},
    {"url":"s3://<your-bucket-name>/load/customer-fw.tbl-007"}
    ]
}
```

Para carregar os dados da tabela CUSTOMER usando o arquivo manifesto

1. Abra o arquivo `customer-fw-manifest` em um editor de texto.
2. Substitua *<your-bucket-name>* pelo nome do seu bucket.
3. Salve o arquivo.
4. Faça upload do arquivo na pasta de carga do bucket.
5. Execute o comando COPY a seguir.

```
copy customer from 's3://<your-bucket-name>/load/customer-fw-manifest'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
fixedwidth 'c_custkey:10, c_name:25, c_address:25, c_city:10, c_nation:15,
 c_region :12, c_phone:15,c_mktsegment:10'
maxerror 10
acceptinvchars as '^'
manifest;
```

Carregar a tabela DWDATE usando DATEFORMAT

Nesta etapa, você usa as opções DELIMITER e DATEFORMAT para carregar a tabela DWDATE.

Ao carregar as colunas DATE e TIMESTAMP, COPY espera o formato padrão, que é AAAA-MM-DD para datas e AAAA-MM-DD HH:MI SS para timestamps. Se os dados de carga não usarem um formato padrão, você poderá usar DATEFORMAT e TIMEFORMAT para especificar o formato.

O trecho a seguir mostra formatos de data na tabela DWDATE. Observe que os formatos de data na coluna dois são inconsistentes.

```
19920104 1992-01-04          Sunday  January 1992 199201 Jan1992 1 4 4 1...
19920112 January 12, 1992 Monday  January 1992 199201 Jan1992 2 12 12 1...
19920120 January 20, 1992 Tuesday     January 1992 199201 Jan1992 3 20 20 1...
```

DATEFORMAT

Você pode especificar somente um formato de data. Se os dados de carga contiverem formatos inconsistentes, possivelmente em colunas diferentes, ou se o formato não for conhecido no tempo de carregamento, você usará DATEFORMAT com o argumento `'auto'`. Quando `'auto'` for especificado, COPY reconhece qualquer formato de data ou hora válido e o converterá no formato padrão. A opção `'auto'` reconhece diversos formatos não compatíveis ao usar uma string DATEFORMAT e TIMEFORMAT. Para obter mais informações, consulte Usar o reconhecimento automático com DATEFORMAT e TIMEFORMAT.

Para carregar a tabela DWDATE, execute o comando COPY a seguir.

```
copy dwdate from 's3://<your-bucket-name>/load/dwdate-tab.tbl'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
delimiter '\t'
dateformat 'auto';
```

Carregar vários arquivos de dados

Você pode usar as opções GZIP e COMPUPDATE para carregar uma tabela.

Você pode carregar uma tabela usando um único arquivo de dados ou vários arquivos. Isso permite comparar os tempos de carregamento dos dois métodos.

GZIP, LZOP e BZIP2

Você pode compactar os arquivos usando os formatos de compactação gzip, lzop ou bzip2. Ao carregar de arquivos compactados, COPY descompacta os arquivos durante o processo de carga. Compactar os arquivos economiza espaço de armazenamento e encurta tempos de upload.

COMPUPDATE

Quando carrega uma tabela vazia sem codificações de compactação, COPY analisa os dados de carga para determinar as codificações ideais. Em seguida, ele altera a tabela para usar essas codificações antes de iniciar a carga. Esse processo de análise demora, mas ocorre, no máximo, uma vez por tabela. Para economizar tempo, você pode ignorar esta etapa desativando COMPUPDATE. Para permitir uma avaliação precisa dos tempos de COPY, você desativa COMPUPDATE para esta etapa.

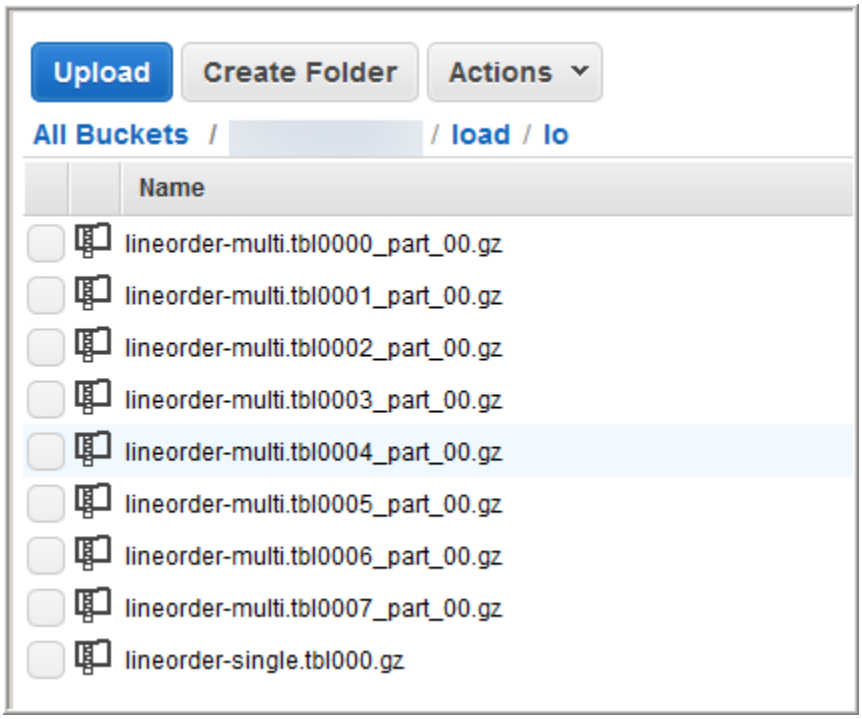
Vários arquivos

O comando COPY pode carregar dados de maneira muito eficiente ao carregar de vários arquivos em paralelo, em vez de carregar de um único arquivo. É possível dividir seus dados em arquivos de modo que o número de arquivos seja um múltiplo do número de fatias em seu cluster. Se você fizer isso, o Amazon Redshift dividirá o workload e distribuirá os dados uniformemente entre as fatias. O número de fatias por nó depende do tamanho do nó do cluster. Para obter mais informações sobre o número de fatias que cada tamanho de nó possui, consulte "Clusters e nós no Amazon Redshift" no Guia de gerenciamento de clusters do Amazon Redshift.

Por exemplo, neste tutorial, os nós de computação no cluster podem ter duas fatias cada; portanto, um cluster com quatro nós tem oito fatias. Em etapas anteriores, os dados da carga estavam contidos em oito arquivos, mesmo que os arquivos fossem muito pequenos. Nesta etapa, você compara a diferença de tempo entre o carregamento de um único arquivo grande e o carregamento de vários arquivos.

Até mesmo arquivos que contêm 15 milhões de registros e ocupam 1,2 GB são muito pequenos na escala do Amazon Redshift. Mas eles são suficientes para demonstrar a vantagem em termos de performance de carregar de vários arquivos.

A imagem a seguir mostra os arquivos de dados para a tabela LINEORDER.



Para avaliar a performance de COPY com vários arquivos

1. Em um teste de laboratório, o comando a seguir foi executado para fazer o COPY usando um único arquivo. Este comando mostra um bucket fictício.

```
copy lineorder from 's3://amzn-s3-demo-bucket/load/lo/lineorder-single.tbl'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
gzip
compupdate off
region 'us-east-1';
```

2. Os resultados são mostrados a seguir. Observe o tempo de execução.

```
Warnings:
Load into table 'lineorder' completed, 14996734 record(s) loaded successfully.

0 row(s) affected.
copy executed successfully

Execution time: 51.56s
```

3. Em seguida, o comando abaixo foi executado para fazer o COPY usando vários arquivos.

```
copy lineorder from 's3://amzn-s3-demo-bucket/load/lo/lineorder-multi.tbl'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
gzip
compupdate off
region 'us-east-1';
```

4. Os resultados são mostrados a seguir. Observe o tempo de execução.

```
Warnings:
Load into table 'lineorder' completed, 14996734 record(s) loaded successfully.

0 row(s) affected.
copy executed successfully

Execution time: 17.7s
```

5. Compare tempos de execução.

No nosso experimento, o tempo para carregar 15 milhões de registros diminuiu de 51,56 segundos para 17,7 segundos, uma redução de 65,7%.

Esses resultados se baseiam no uso de um cluster de quatro nós. Se o cluster tiver mais nós, o tempo economizado será multiplicado. Para clusters típicos do Amazon Redshift, com dezenas a centenas de nós, a diferença é ainda mais dramática. Se você tiver um único cluster de nó, haverá pouca diferença entre os tempos de execução.

Etapa 6: Vacuum e análise do banco de dados

Sempre que adicionar, excluir ou modificar um número significativo de linhas, você deve executar um comando VACUUM e depois um comando ANALYZE. Uma limpeza recupera o espaço de linhas excluídas e restaura a ordem de classificação. O comando ANALYZE atualiza os metadados de estatísticas, o que permite ao otimizador de consultas gerar planos de consulta mais precisos. Para obter mais informações, consulte Vacuum de tabelas.

Se você carregar os dados na ordem da chave de classificação, a limpeza será rápida. Neste tutorial, você adicionou um número significativo de linhas, mas as adicionou a tabelas vazias. Sendo esse o caso, não há necessidade de corrigir, e você não excluiu linhas. COPY atualizará automaticamente estatísticas de atualizações depois de carregar uma tabela vazia, logo, as estatísticas devem permanecer atualizadas. Porém, por uma questão de boas práticas de manutenção, você conclui este tutorial limpando e analisando o banco de dados.

Para limpar e analisar o banco de dados, execute os comandos a seguir.

```
vacuum;
analyze;
```

Etapa 7: limpar os recursos

O cluster continua acumulando cobranças enquanto está em execução. Ao concluir este tutorial, você deve retornar seu ambiente ao estado anterior seguindo as etapas na Etapa 5: Revogar o acesso e excluir sua amostra de cluster no Guia de conceitos básicos do Amazon Redshift.

Caso queira manter o cluster, mas recuperar o armazenamento usado pelas tabelas SSB, execute os comandos a seguir.

```
drop table part;
```

```
drop table supplier;
drop table customer;
drop table dwdate;
drop table lineorder;
```

Próximo

Resumo

Resumo

Neste tutorial, você carregou arquivos de dados no Amazon S3 e, em seguida, usou os comandos COPY para carregar os dados dos arquivos nas tabelas do Amazon Redshift.

Você carregou dados usando os seguintes formatos:

- Delimitado por caracteres
- CSV
- Largura fixa

Você usou a tabela do sistema STL_LOAD_ERRORS para solucionar erros de carga e usou as opções REGION, MANIFEST, MAXERROR, ACCEPTINVCHARS, DATEFORMAT e NULL AS para resolver os erros.

Você aplicou estas melhores práticas para carregar dados:

- Uso de um comando COPY para carregar dados
- Carregar arquivos de dados
- Uso de um único comando COPY para carregar a partir de vários arquivos
- Compactar arquivos de dados
- Verificação de arquivos de dados antes de depois de um carregamento

Para obter mais informações sobre as práticas recomendadas do Amazon Redshift, consulte os seguintes links:

- Práticas recomendadas do Amazon Redshift para carregamento de dados
- Práticas recomendadas do Amazon Redshift para projetar tabelas

- Práticas recomendadas do Amazon Redshift para criar consultas

Descarregar dados no Amazon Redshift

Para descarregar dados de tabelas de banco de dados para um conjunto de arquivos em um bucket do Amazon S3, você pode usar o comando UNLOAD com uma instrução SELECT. Você pode descarregar dados de texto em formato delimitado ou em formato de largura fixa, independente do formato de dados usado para carregá-los. Você também pode especificar se arquivos GZIP compactados devem ser criados.

Você pode limitar o acesso que os usuários têm ao seu bucket do Amazon S3 usando credenciais de segurança temporárias.

Tópicos

- Descarregar dados do Amazon S3
- Descarregamento de arquivos de dados criptografados
- Descarregamento de dados em formato delimitado ou de largura fixa
- Recarregamento de dados descarregados

Descarregar dados do Amazon S3

O Amazon Redshift divide os resultados de uma instrução selecionada em um conjunto de arquivos, um ou mais arquivos por fatia de nó, para simplificar o recarregamento paralelo dos dados. Como alternativa, você pode especificar que UNLOAD deve gravar os resultados em série para um ou mais arquivos adicionando a opção PARALLEL OFF. Você pode limitar o tamanho dos arquivos no Amazon S3 especificando o parâmetro MAXFILESIZE. O UNLOAD criptografa automaticamente os arquivos de dados usando a criptografia do lado do servidor Amazon S3 (SSE-S3).

Você pode usar qualquer instrução select no comando UNLOAD que o Amazon Redshift suporta, exceto para um select que usa uma cláusula LIMIT no select externo. Por exemplo, você pode usar uma instrução de seleção que inclua colunas específicas ou que use uma cláusula where para junção de várias tabelas. Se sua consulta contiver aspas (delimitando valores literais, por exemplo), você precisa escapá-las no texto da consulta (\'). Para obter mais informações, consulte a referência do comando SELECT. Para obter mais informações sobre o uso da cláusula LIMIT, consulte as Observações de uso para o comando UNLOAD.

Por exemplo, o comando UNLOAD a seguir envia o conteúdo da tabela VENUE para o bucket do Amazon S3 `s3://amzn-s3-demo-bucket/tickit/unload/`.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/tickit/unload/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Os nomes dos arquivos criados pelo exemplo anterior incluem o prefixo 'venue_'.

```
venue_0000_part_00
venue_0001_part_00
venue_0002_part_00
venue_0003_part_00
```

Por padrão, UNLOAD grava dados em paralelo a vários arquivos, de acordo com o número de fatias no cluster. Para gravar dados em um único arquivo, especifique a PARALLEL OFF. UNLOAD grava os dados em série, classificados absolutamente de acordo com a cláusula ORDER BY, se ela for usada. O tamanho máximo de um arquivo de dados é 6,2 GB. Se o tamanho dos dados for maior que o máximo, UNLOAD criará arquivos adicionais, de até 6,2 GB cada.

O seguinte exemplo grava os conteúdos de VENUE em um único arquivo. Somente um arquivo for necessário, pois o tamanho do arquivo é menor que 6,2 GB.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/tickit/unload/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
parallel off;
```

> **Note**
>
> O comando UNLOAD é projetado para usar processamento paralelo. Recomendamos deixar PARALLEL habilitado na maioria dos casos, especialmente se os arquivos serão usados para carregar tabelas usando um comando COPY.

Supondo que o tamanho total dos dados de VENUE seja 5 GB, o seguinte exemplo grava os conteúdos de VENUE em 50 arquivos, cada um com 100 MB de tamanho.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/tickit/unload/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
parallel off
```

```
maxfilesize 100 mb;
```

Se você incluir um prefixo na string do caminho do Amazon S3, UNLOAD usará esse prefixo para os nomes dos arquivos.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/tickit/unload/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Você pode criar um arquivo manifesto que liste os arquivos para descarregamento especificando a opção MANIFEST no comando UNLOAD. O manifesto é um arquivo de texto no formato JSON que lista explicitamente a URL de cada arquivo que foi gravado no Amazon S3.

O seguinte exemplo inclui a opção de manifesto:

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/tickit/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
manifest;
```

O seguinte exemplo mostra um manifesto para quatro arquivos de descarregamento.

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket/tickit/venue_0000_part_00"},
    {"url":"s3://amzn-s3-demo-bucket/tickit/venue_0001_part_00"},
    {"url":"s3://amzn-s3-demo-bucket/tickit/venue_0002_part_00"},
    {"url":"s3://amzn-s3-demo-bucket/tickit/venue_0003_part_00"}
  ]
}
```

O arquivo manifesto pode ser usado para carregar os mesmos arquivos usando COPY com a opção MANIFEST. Para obter mais informações, consulte Uso de um manifesto para especificar arquivos de dados.

Após a conclusão de uma operação UNLOAD, confirme se os dados foram descarregados corretamente navegando até o bucket do Amazon S3 onde UNLOAD gravou os arquivos. Você verá um ou mais arquivos numerados por fatia, começando com o número zero. Se você especificou a opção MANIFEST, você também verá um arquivo que termina com "". 'manifest'. Por exemplo:

```
amzn-s3-demo-bucket/tickit/venue_0000_part_00
amzn-s3-demo-bucket/tickit/venue_0001_part_00
amzn-s3-demo-bucket/tickit/venue_0002_part_00
amzn-s3-demo-bucket/tickit/venue_0003_part_00
amzn-s3-demo-bucket/tickit/venue_manifest
```

Você pode obter programaticamente uma lista dos arquivos que foram gravados no Amazon S3 chamando uma operação de listagem do Amazon S3 após a conclusão do UNLOAD. Você também pode consultar STL_UNLOAD_LOG.

A consulta a seguir retorna o nome do caminho para os arquivos que foram criados por um UNLOAD: A função PG_LAST_QUERY_ID retorna a consulta mais recente.

```
select query, substring(path,0,40) as path
from stl_unload_log
where query=2320
order by path;

query |             path
-------+--------------------------------------
  2320 | s3://amzn-s3-demo-bucket/venue0000_part_00
  2320 | s3://amzn-s3-demo-bucket/venue0001_part_00
  2320 | s3://amzn-s3-demo-bucket/venue0002_part_00
  2320 | s3://amzn-s3-demo-bucket/venue0003_part_00
(4 rows)
```

Se a quantidade de dados for muito grande, o Amazon Redshift pode dividir os arquivos em várias partes por fatia. Por exemplo:

```
venue_0000_part_00
venue_0000_part_01
venue_0000_part_02
venue_0001_part_00
venue_0001_part_01
venue_0001_part_02
...
```

O comando UNLOAD a seguir inclui uma string entre aspas na instrução select, portanto, as aspas são escapadas (=\'OH\' ').

```
unload ('select venuename, venuecity from venue where venuestate=\'OH\' ')
```

```
to 's3://amzn-s3-demo-bucket/tickit/venue/ '
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Por padrão, UNLOAD falhará em vez de substituir arquivos existentes no bucket de destino. Para substituir arquivos existentes, incluindo o arquivo manifesto, especifique a opção ALLOWOVERWRITE.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/venue_pipe_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
manifest
allowoverwrite;
```

Descarregamento de arquivos de dados criptografados

UNLOAD cria arquivos automaticamente usando a criptografia do lado do servidor do Amazon S3 (SSE-S3) com chaves de criptografia gerenciadas pela AWS. Você também pode especificar a criptografia no lado do servidor com uma chave do AWS Key Management Service (SSE-KMS) ou a criptografia do lado do cliente com uma chave gerenciada pelo cliente. Não é possível usar UNLOAD com a criptografia do lado do servidor do Amazon S3 utilizando uma chave fornecida pelo cliente. Para obter mais informações, consulte Proteger dados usando a criptografia no lado do servidor.

Para descarregar no Amazon S3 usando criptografia do lado do servidor com uma chave AWS KMS, use o parâmetro KMS_KEY_ID para fornecer o ID da chave, conforme mostrado no exemplo a seguir.

```
unload ('select venuename, venuecity from venue')
to 's3://amzn-s3-demo-bucket/encrypted/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
KMS_KEY_ID '1234abcd-12ab-34cd-56ef-1234567890ab'
encrypted;
```

Se desejar fornecer sua própria chave de criptografia, você pode criar arquivos de dados criptografados do lado do cliente no Amazon S3 usando o comando UNLOAD com a opção ENCRYPTED. UNLOAD usa o mesmo processo de criptografia de envelope que a criptografia do lado do cliente do Amazon S3 usa. Você pode usar o comando COPY com a opção ENCRYPTED para carregar os arquivos criptografados.

O processo funciona deste modo:

1. Crie uma chave AES de 256 bits codificada por base64 que será usada como sua chave de criptografia privada ou chave simétrica raiz.
2. Você emite um comando UNLOAD que inclua sua a chave simétrica raiz e a opção ENCRYPTED.
3. UNLOAD gera uma chave simétrica de uso único (chamada chave simétrica de envelope) e um vetor de inicialização (IV), que a usa para criptografar seus dados.
4. UNLOAD criptografa a chave simétrica de envelope usando sua chave simétrica raiz.
5. O UNLOAD então armazena os arquivos de dados criptografados no Amazon S3 e armazena a chave do envelope criptografado e IV como metadados de objeto com cada arquivo. A chave de envelope criptografada é armazenada como metadados de objeto `x-amz-meta-x-amz-key` e o IV é armazenado como metadados de objeto `x-amz-meta-x-amz-iv`.

Para obter mais informações sobre o processo de criptografia de envelope, consulte o artigo Criptografia de dados do lado do cliente com o AWS SDK for Java e o Amazon S3.

Para descarregar arquivos de dados criptografados, adicione a chave-valor raiz à string de credenciais e inclua a opção ENCRYPTED. Se você usar a opção MANIFEST, o arquivo manifesto também será criptografado.

```
unload ('select venuename, venuecity from venue')
to 's3://amzn-s3-demo-bucket/encrypted/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
master_symmetric_key '<root_key>'
manifest
encrypted;
```

Para descarregar arquivos de dados criptografados que estão compactador por GZIP, inclua a opção GZIP junto com a chave-valor raiz e a opção ENCRYPTED.

```
unload ('select venuename, venuecity from venue')
to 's3://amzn-s3-demo-bucket/encrypted/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
master_symmetric_key '<root_key>'
encrypted gzip;
```

Para carregar os arquivos de dados criptografados, adicione o parâmetro de MASTER_SYMMETRIC_KEY com a mesma chave-valor raiz e inclua a opção ENCRYPTED.

```
copy venue from 's3://amzn-s3-demo-bucket/encrypted/venue_'
```

```
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
master_symmetric_key '<root_key>'
encrypted;
```

Descarregamento de dados em formato delimitado ou de largura fixa

Você pode descarregar dados em formato delimitado ou de largura fixa. A saída padrão é delimitada por pipe (usando o caractere '|').

O seguinte exemplo especifica uma vírgula como o delimitador:

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/tickit/venue/comma'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter ',';
```

Os arquivos de saída resultantes são semelhantes a:

```
20,Air Canada Centre,Toronto,ON,0
60,Rexall Place,Edmonton,AB,0
100,U.S. Cellular Field,Chicago,IL,40615
200,Al Hirschfeld Theatre,New York City,NY,0
240,San Jose Repertory Theatre,San Jose,CA,0
300,Kennedy Center Opera House,Washington,DC,0
...
```

Para descarregar o mesmo conjunto de resultados para um arquivo delimitado por guia, emita o seguinte comando:

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/tickit/venue/tab'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter as '\t';
```

Como alternativa, você pode usar uma especificação FIXEDWIDTH. Essa especificação consiste em um identificador para cada coluna da tabela e a largura da coluna (número de caracteres). O comando UNLOAD falhará em vez de truncar os dados, portanto especifica uma largura que tenha no mínimo o mesmo comprimento da entrada mais longa para aquela coluna. O descarregamento

de dados de largura fixa funciona de modo similar ao descarregamento de dados delimitados, exceto que a saída resultante não contém caracteres delimitadores. Por exemplo:

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/tickit/venue/fw'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
fixedwidth '0:3,1:100,2:30,3:2,4:6';
```

A saída de largura fixa é semelhante a:

```
20 Air Canada Centre         Toronto      ON0
60 Rexall Place              Edmonton     AB0
100U.S. Cellular Field       Chicago      IL40615
200Al Hirschfeld Theatre     New York CityNY0
240San Jose Repertory TheatreSan Jose     CA0
300Kennedy Center Opera HouseWashington   DC0
```

Para mais detalhes sobre especificações FIXEDWIDTH, consulte o comando UNLOAD.

Recarregamento de dados descarregados

Para recarregar os resultados de uma operação de descarregamento, você pode usar o comando COPY.

O seguinte exemplo mostra um caso simples no qual a tabela VENUE é descarregada usando um arquivo manifesto, truncada e recarregada.

```
unload  ('select * from venue order by venueid')
to 's3://amzn-s3-demo-bucket/tickit/venue/reload_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
manifest
delimiter '|';

truncate venue;

copy venue
from 's3://amzn-s3-demo-bucket/tickit/venue/reload_manifest'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
manifest
delimiter '|';
```

Após seu recarregamento, a tabela VENUE tem a seguinte aparência:

```
select * from venue order by venueid limit 5;

 venueid |         venuename          |  venuecity  | venuestate | venueseats
---------+----------------------------+-------------+------------+-----------
       1 | Toyota Park                | Bridgeview  | IL         |          0
       2 | Columbus Crew Stadium      | Columbus    | OH         |          0
       3 | RFK Stadium                | Washington  | DC         |          0
       4 | CommunityAmerica Ballpark | Kansas City | KS         |          0
       5 | Gillette Stadium           | Foxborough  | MA         |      68756
(5 rows)
```

Funções definidas pelo usuário (UDFs) no Amazon Redshift

Você pode criar uma função escalar personalizada definida pelo usuário (UDF) usando uma cláusula SQL SELECT ou um programa Python. A nova função é armazenada no banco de dados e está disponível para qualquer usuário com privilégios suficientes para ser executada. Você executa um UDF escalar personalizado da mesma forma que executa funções existentes do Amazon Redshift.

Para UDFs Python, além da utilização da funcionalidade padrão Python, você pode importar seus próprios módulos Python personalizados. Para obter mais informações, consulte Suporte da linguagem Python para UDFs. O Python 3 não está disponível para UDFs do Python. Para obter suporte ao Python 3 para UDFs do Amazon Redshift, use UDFs escalares do Lambda.

Você também pode criar AWS Lambda UDFs que usam funções personalizadas definidas no Lambda como parte de suas consultas SQL. Com UDFs do Lambda, você pode escrever UDFs complexas e integrar componentes de terceiros. Eles também podem ajudá-lo a superar algumas das limitações dos atuais UDFs Python e SQL. Por exemplo, eles podem ajudá-lo a acessar recursos de rede e armazenamento e escrever mais instruções SQL completas. Você pode criar UDFs do Lambda em qualquer uma das linguagens de programação compatíveis com o Lambda, como Java, Go, PowerShell, Node.js, C#, Python e Ruby. Ou você pode usar um tempo de execução personalizado.

Por padrão, todos os usuários podem executar UDFs. Para obter mais informações sobre privilégios, consulte Segurança e permissões de UDFs.

Tópicos

- Segurança e permissões de UDFs
- Evitar conflitos na nomeação da UDF
- UDFs escalares de SQL
- UDFs escalares do Python
- UDFs escalares do Lambda
- Exemplos de caso de uso referentes a funções definidas pelo usuário (UDFs)

Segurança e permissões de UDFs

Para criar uma UDF, você deve ter permissão de uso na linguagem para SQL ou plpythonu (Python). Por padrão, USAGE ON LANGUAGE SQL é concedido a PUBLIC, mas você deve conceder explicitamente USAGE ON LANGUAGE PLPYTHONU a usuários ou grupos específicos.

Para revogar o uso na SQL, revogue primeiro o uso em PUBLIC. Depois, conceda o uso na SQL somente a usuários ou grupos específicos que tenham permissão para criar UDFs SQL. O exemplo a seguir revoga o uso na SQL em PUBLIC. Depois, ele concede o uso ao grupo de usuários `udf_devs`.

```
revoke usage on language sql from PUBLIC;
grant usage on language sql to group udf_devs;
```

Para executar uma UDF, você deve ter permissão de execução em cada função. Por padrão, a permissão de execução de novas UDFs é concedida a PUBLIC. Para restringir o uso, revogue a execução da função em PUBLIC. Depois, conceda o privilégio a pessoas ou grupos específicos.

O exemplo a seguir revoga a execução na função `f_py_greater` em PUBLIC. Depois, ele concede o uso ao grupo de usuários `udf_devs`.

```
revoke execute on function f_py_greater(a float, b float) from PUBLIC;
grant execute on function f_py_greater(a float, b float) to group udf_devs;
```

Os superusuários têm todos os privilégios por padrão.

Para obter mais informações, consulte GRANT e REVOKE.

Evitar conflitos na nomeação da UDF

Você pode evitar potenciais conflitos e resultados inesperados considerando suas convenções de nomenclatura de UDFs antes da implementação. Como os nomes de função podem ser sobrecarregados, eles podem colidir com nomes de função existentes e futuros do Amazon Redshift. Este tópico discute a sobrecarga e apresenta uma estratégia para evitar conflitos.

Sobrecarga de nomes de função

Uma função é identificada por seu nome e assinatura, que é o número de argumentos de entrada e tipos de dados dos argumentos. Duas funções no mesmo esquema podem ter o mesmo nome se tiverem assinaturas diferentes. Em outras palavras nomes de função podem ser sobrecarregados.

Quando você executa uma consulta, o mecanismo de consulta determina qual função deve ser chamada com base no número de argumentos fornecidos e nos tipos de dados dos argumentos. Você pode usar a sobrecarga para simular funções com um número variável de argumentos, até o limite permitido pelo comando CREATE FUNCTION.

Evitar conflitos com funções integradas do Amazon Redshift

Recomendamos que você nomeie todas as UDFs usando o prefixo `f_`. O Amazon Redshift reserva o prefixo `f_` exclusivamente para UDFs e, ao prefixar os nomes das suas UDFs com `f_`, você garante que o nome da UDF não entre em conflito com nomes de funções SQL integradas existentes ou futuras do Amazon Redshift. Por exemplo, ao nomear uma nova UDF `f_sum`, você evita conflito com a função SUM do Amazon Redshift. Da mesma forma, se você nomear uma nova função `f_fibonacci`, evitará conflito se o Amazon Redshift adicionar uma função chamada FIBONACCI em uma versão futura.

Você pode criar uma UDF com o mesmo nome e assinatura de uma função SQL integrada existente do Amazon Redshift sem que o nome da função seja sobrecarregado se a UDF e a função integrada existirem em esquemas diferentes. Como as funções integradas existem no esquema de catálogo de sistema, pg_catalog, você pode criar uma UDF com o mesmo nome em outro esquema, tal como um esquema público ou definido pelo usuário. Em alguns casos, é possível chamar uma função que não está explicitamente qualificada com um nome de esquema. Em caso afirmativo, o Amazon Redshift pesquisa primeiro o esquema pg_catalog por padrão. Assim, uma função interna é executada antes de uma nova UDF com o mesmo nome.

Você pode alterar esse comportamento configurando o caminho de pesquisa para colocar pg_catalog no final. Se você fizer isso, suas UDFs tomem precedência sobre funções integradas, mas a prática pode causar resultados inesperados. A adoção de uma estratégia de nomeação exclusiva, tal como a utilização do prefixo reservado `f_`, é uma prática mais confiável. Para obter mais informações, consulte SET e search_path.

UDFs escalares de SQL

Uma UDF SQL escalar incorpora uma cláusula SQL SELECT que é executada quando a função é chamada e retorna um valor único. O comando CREATE FUNCTION define os seguintes parâmetros:

- Argumentos de entrada (opcionais). Cada argumento deve ter um tipo de dados.
- Um tipo de dados de retorno.
- Uma cláusula SQL SELECT. Na cláusula SELECT, consulte os argumentos de entrada usando $1, $2 etc. de acordo com a ordem dos argumentos na definição de função.

Os tipos de dados de entrada e de retorno podem ser qualquer tipo de dados padrão do Amazon Redshift.

Não inclua uma cláusula FROM na cláusula SELECT. Em vez disso, inclua a cláusula FROM na instrução SQL que chama uma UDF SQL.

A cláusula SELECT não pode conter estes tipos de cláusulas:

- FROM
- INTO
- WHERE
- GROUP BY
- ORDER BY
- LIMIT

Exemplo de função SQL escalar

O seguinte exemplo cria uma função que compara dois números e retorna o maior valor. Para obter mais informações, consulte CREATE FUNCTION.

```
create function f_sql_greater (float, float)
  returns float
stable
as $$
  select case when $1 > $2 then $1
```

```
    else $2
  end
$$ language sql;
```

A consulta a seguir chama a nova função f_sql_greater para consultar a tabela SALES e retornar COMMISSION ou 20% de PRICEPAID, o que for maior.

```
select f_sql_greater(commission, pricepaid*0.20) from sales;
```

UDFs escalares do Python

Uma UDF Python escalar incorpora um programa Python que é executado quando a função é chamada e retorna um valor único. O comando CREATE FUNCTION define os seguintes parâmetros:

- Argumentos de entrada (opcionais). Cada argumento deve ter um nome e um tipo de dados.
- Um tipo de dados de retorno.
- Um programa Python executável.

Os tipos de dados de entrada e de retorno para UDFs Python podem ser qualquer um destes:

- SMALLINT
- INTEGER
- BIGINT
- DECIMAL
- REAL
- DOUBLE PRECISION
- BOOLEAN
- CHAR
- VARCHAR
- DATE
- TIMESTAMP
- ANYELEMENT

Os aliases desses tipos também são válidos. Para ver uma lista completa dos tipos de dados e os respectivos aliases, consulte Tipos de dados.

Quando UDFs Python podem usar o tipo de dados ANYELEMENT, o Amazon Redshift converte automaticamente em um tipo de dados padrão com base nos argumentos fornecidos em tempo de execução. Para obter mais informações, consulte Tipo de dados ANYELEMENT.

Quando uma consulta do Amazon Redshift chama uma UDF escalar, as seguintes etapas ocorrem no tempo de execução:

1. A função converte os argumentos de entrada para tipos de dados Python.

   Para obter um mapeamento de tipos de dados Amazon Redshift para tipos de dados Python, consulte Tipos de dados da UDF Python.
2. A função executa o programa Python, passando os argumentos de entrada convertidos.
3. O código Python retorna um único valor. O tipo de dados do valor de retorno deve corresponder ao tipo de dados de RETURNS especificado pela definição da função.
4. A função converte o valor de retorno do Python no tipo de dados Amazon Redshift especificado e, em seguida, retorna esse valor para a consulta.

> **Note**
>
> O Python 3 não está disponível para UDFs do Python. Para obter suporte ao Python 3 para UDFs do Amazon Redshift, use UDFs escalares do Lambda.

Exemplo de UDF Python escalar

O seguinte exemplo cria uma função que compara dois números e retorna o maior valor. Observe que o recuo do código entre os cifrões duplos ($$) é um requisito do Python. Para obter mais informações, consulte CREATE FUNCTION.

```
create function f_py_greater (a float, b float)
  returns float
stable
as $$
  if a > b:
    return a
```

```
  return b
$$ language plpythonu;
```

A seguinte consulta chama a nova função `f_greater` para consultar a tabela SALES e retornar a COMMISSION ou 20 por cento de PRICEPAID, o que for maior.

```
select f_py_greater (commission, pricepaid*0.20) from sales;
```

Tipos de dados da UDF Python

As UDFs Python podem usar qualquer tipo de dados padrão do Amazon Redshift para os argumentos de entrada e o valor de retorno da função. Além dos tipos de dados padrão, as UDFs suportam o tipo de dados ANYELEMENT, que o Amazon Redshift converte automaticamente em um tipo de dados padrão com base nos argumentos fornecidos no tempo de execução. UDFs escalares podem retornar um tipo de dado ANYELEMENT. Para obter mais informações, consulte Tipo de dados ANYELEMENT.

Durante a execução, o Amazon Redshift converte os argumentos dos tipos de dados do Amazon Redshift para tipos de dados Python para processamento. Em seguida, ele converte o valor de retorno do tipo de dados Python para o tipo de dados do Amazon Redshift correspondente. Para obter mais informações sobre tipos de dados do Amazon Redshift, consulte Tipos de dados.

A tabela a seguir mapeia tipos de dados Amazon Redshift para tipos de dados Python.

Tipo de dados do Amazon Redshift	Tipo de dados do Python
smallint	int
integer	
bigint	
curto	
longo	
decimal ou numérico	decimal
double	flutuação

Tipo de dados do Amazon Redshift	Tipo de dados do Python
real	
booleano	bool
char varchar	string
timestamp	datetime

Tipo de dados ANYELEMENT

ANYELEMENT é um tipo de dados polimórfico. Isso significa que se uma função for declarada usando ANYELEMENT para o tipo de dados de um argumento, a função pode aceitar qualquer tipo de dados Amazon Redshift padrão como entrada para esse argumento quando a função é chamada. O argumento de ANYELEMENT é definido para o tipo de dados realmente transmitido para ele quando a função é chamada.

Se uma função usa vários tipos de dados ANYELEMENT, todos devem se resolver para o mesmo tipo de dados real quando a função é chamada. Todos os tipos de dados de argumento ANYELEMENT são definidos como o tipo de dados real do primeiro argumento transmitido para um ANYELEMENT. Por exemplo, uma função declarada como `f_equal(anyelement, anyelement)` aceitará quaisquer dois valores de entrada, desde que eles sejam do mesmo tipo de dados.

Se o valor de retorno de uma função é declarado como ANYELEMENT, pelo menos um argumento de entrada deve ser ANYELEMENT. O tipo de dados real para o valor de retorno é o mesmo que o tipo de dados real fornecido para o argumento de entrada ANYELEMENT.

Suporte da linguagem Python para UDFs

Você pode criar uma UDF personalizada com base na linguagem de programação Python. A Biblioteca padrão Python 2.7 está disponível para uso em UDFs, com exceção dos seguintes módulos:

- ScrolledText
- Tix
- Tkinter

- tk
- turtle
- smtpd

Além da biblioteca padrão do Python, os seguintes módulos fazem parte da implementação do Amazon Redshift:

- numpy 1.8.2
- pandas 0.14.1
- python-dateutil 2.2
- pytz 2014.7
- scipy 0.12.1
- six 1.3.0
- wsgiref 0.1.2

Você também pode importar seus próprios módulos Python personalizados e disponibilizá-los para uso em UDFs executando um comando CREATE LIBRARY. Para obter mais informações, consulte Exemplo: importar módulos personalizados da biblioteca Python.

> **Important**
>
> O Amazon Redshift bloqueia todo o acesso à rede e acesso de gravação ao sistema de arquivos por meio de UDFs.

> **Note**
>
> O Python 3 não está disponível para UDFs do Python. Para obter suporte ao Python 3 para UDFs do Amazon Redshift, use UDFs escalares do Lambda.

Exemplo: importar módulos personalizados da biblioteca Python

Você define funções escalares usando a sintaxe de linguagem Python. Você pode usar os módulos Python Standard Library e os módulos pré-instalados do Amazon Redshift. Você também pode criar

seus próprios módulos personalizados da biblioteca Python e importar as bibliotecas para seus clusters ou usar bibliotecas existentes do Python ou de terceiros.

Você não pode criar uma biblioteca que contém um módulo com o mesmo nome de um módulo de biblioteca padrão Python ou um módulo Python pré-instalado do Amazon Redshift. Se uma biblioteca instalada pelo usuário existente usa o mesmo pacote Python que uma biblioteca criada por você, é necessário remover a biblioteca existente antes de instalar a nova biblioteca.

Você deve ser um superusuário ou ter privilégio `USAGE ON LANGUAGE plpythonu` para instalar bibliotecas personalizadas; entretanto, qualquer usuário com privilégios suficientes para criar funções pode usar as bibliotecas instaladas. Você pode consultar o catálogo de sistema PG_LIBRARY para visualizar informações sobre as bibliotecas instaladas em seu cluster.

Importar um módulo Python personalizado para o cluster

Esta seção fornece um exemplo de importação de um módulo Python personalizado para o seu cluster. Para executar as etapas desta seção, você deve ter um bucket do Amazon S3, onde você carrega o pacote de biblioteca. Então, você instala o pacote em seu cluster. Para obter mais informações sobre como criar buckets, consulte Criar um bucket no Guia do usuário do Amazon Simple Storage Service.

Neste exemplo, suponha que você crie UDFs para trabalhar com posições e distâncias nos seus dados. Conecte-se ao seu cluster Amazon Redshift a partir de uma ferramenta de cliente SQL e execute os comandos a seguir para criar as funções.

```
CREATE FUNCTION f_distance (x1 float, y1 float, x2 float, y2 float) RETURNS float
 IMMUTABLE as $$
    def distance(x1, y1, x2, y2):
        import math
        return math.sqrt((y2 - y1) ** 2 + (x2 - x1) ** 2)

    return distance(x1, y1, x2, y2)
$$ LANGUAGE plpythonu;

CREATE FUNCTION f_within_range (x1 float, y1 float, x2 float, y2 float) RETURNS bool
 IMMUTABLE as $$
    def distance(x1, y1, x2, y2):
        import math
        return math.sqrt((y2 - y1) ** 2 + (x2 - x1) ** 2)

    return distance(x1, y1, x2, y2) < 20
```

```
$$ LANGUAGE plpythonu;
```

Observe que algumas linhas de código estão duplicadas nas funções anteriores. Essa duplicação é necessária pois uma UDF não pode fazer referência ao conteúdo de outra UDF e ambas as funções exigem a mesma funcionalidade. Contudo, em vez de duplicar o código em várias funções, você pode criar uma biblioteca personalizada e configurar as funções para usá-la.

Para fazer isso, crie o pacote da biblioteca seguindo estas etapas:

1. Crie uma pasta chamada geometry. Essa pasta é o pacote de nível superior da biblioteca.
2. Na pasta geometry, crie um arquivo chamado `__init__.py`. Observe que o nome do arquivo contém dois caracteres duplos de sublinhado. Este arquivo indica para Python que o pacote pode ser inicializado.
3. Também na pasta geometry, crie uma pasta chamada trig. Essa pasta é o subpacote da biblioteca.
4. Na pasta trig, crie outro arquivo chamado `__init__.py` e um arquivo chamado `line.py`. Nesta pasta, `__init__.py` indica a Python que o subpacote pode ser inicializado e que `line.py` é o arquivo que contém o código da biblioteca.

   Suas pasta e estrutura de arquivo devem ser exatamente como se segue:

   ```
   geometry/
      __init__.py
      trig/
         __init__.py
         line.py
   ```

   Para mais informações sobre a estrutura de pacotes, acesse Módulos no tutorial na página da Python.
5. O seguinte código contém uma classe e funções de membro para a biblioteca. Copie-o e cole-o em `line.py`.

   ```
   class LineSegment:
     def __init__(self, x1, y1, x2, y2):
       self.x1 = x1
       self.y1 = y1
       self.x2 = x2
       self.y2 = y2
     def angle(self):
   ```

```
    import math
    return math.atan2(self.y2 - self.y1, self.x2 - self.x1)
  def distance(self):
    import math
    return math.sqrt((self.y2 - self.y1) ** 2 + (self.x2 - self.x1) ** 2)
```

Depois de criar o pacote, faça como mostrado a seguir para preparar o pacote e carregá-lo no Amazon S3.

1. Comprima o conteúdo da pasta geometry em um arquivo de .zip chamado geometry.zip. Não inclua a própria pasta geometry; inclua somente os conteúdo da pasta, conforme mostrado a seguir:

```
geometry.zip
   __init__.py
   trig/
      __init__.py
      line.py
```

2. Carregue geometry.zip para o seu bucket do Amazon S3.

> **Important**
>
> Se o bucket do Amazon S3 não estiver na mesma região que o cluster do Amazon Redshift, você deverá usar a opção REGION para especificar a região em que os dados estão localizados. Para obter mais informações, consulte CREATE LIBRARY.

3. A partir de sua ferramenta do cliente SQL, execute o seguinte comando para instalar a biblioteca. Substitua *<bucket_name>* pelo nome do bucket e *<access key id>* e *<secret key>* por uma chave de acesso e chave de acesso secreta de suas credenciais de usuário do AWS Identity and Access Management (IAM).

```
CREATE LIBRARY geometry LANGUAGE plpythonu FROM 's3://<bucket_name>/geometry.zip'
 CREDENTIALS 'aws_access_key_id=<access key id>;aws_secret_access_key=<secret key>';
```

Após instalar a biblioteca em seu cluster, você precisará configurar suas funções para que usem a biblioteca. Para fazer isso, execute os comandos a seguir.

```
CREATE OR REPLACE FUNCTION f_distance (x1 float, y1 float, x2 float, y2 float) RETURNS
 float IMMUTABLE as $$
    from trig.line import LineSegment

    return LineSegment(x1, y1, x2, y2).distance()
$$ LANGUAGE plpythonu;

CREATE OR REPLACE FUNCTION f_within_range (x1 float, y1 float, x2 float, y2 float)
 RETURNS bool IMMUTABLE as $$
    from trig.line import LineSegment

    return LineSegment(x1, y1, x2, y2).distance() < 20
$$ LANGUAGE plpythonu;
```

Nos comandos anteriores, `import trig/line` elimina o código duplicado das funções originais nesta seção. Você pode reutilizar a funcionalidade fornecida por essa biblioteca em várias UDFs. Observe que para importar o módulo, você somente precisa especificar o caminho para o subpacote e nome do módulo (`trig/line`).

Restrições de UDFs do Python

Dentro das restrições listadas neste tópico, você pode usar UDFs em qualquer lugar em que usar as funções escalares integradas do Amazon Redshift. Para obter mais informações, consulte Referência de funções SQL.

As UDFs do Amazon Redshift Python têm as seguintes restrições:

- As UDFs Python não podem acessar a rede nem ler ou gravar o sistema de arquivos.
- O tamanho total das bibliotecas Python instaladas pelo usuário não pode exceder 100 MB.
- O Amazon Redshift só pode executar uma UDF Python por vez para clusters provisionados usando o gerenciamento de workload (WLM) automático e para grupos de trabalho de tecnologia sem servidor. Se você tentar executar mais de uma UDF simultaneamente, o Amazon Redshift enfileirará as UDFs restantes do Python para serem executadas nas filas de gerenciamento de workload. As UDFs SQL não têm um limite de simultaneidade ao usar o WLM automático.
- Ao usar o WLM manual para clusters provisionados, o número de UDFs Python que podem ser executadas simultaneamente por cluster é limitado a um quarto do nível total de simultaneidade do cluster. Por exemplo, um cluster provisionado com uma simultaneidade de 15 pode executar no máximo três UDFs Python simultâneas.

- Ao usar UDFs Python, o Amazon Redshift não oferece suporte aos tipos de dados SUPER e HLLSKETCH.

Registro em log de erros e alertas em UDFs do Python

Você pode usar o módulo de log Python para criar mensagens de erro e alerta definidas pelo usuário em suas UDFs. Depois da execução da consulta, você poderá consultar a exibição do sistema SVL_UDF_LOG para recuperar mensagens registradas.

> **Note**
>
> O log de UDF consome recursos do cluster e pode afetar a performance do sistema. Recomendamos a implementação de log somente para desenvolvimento e solução de problemas.

Durante a execução da consulta, o handler do log grava mensagens na exibição de sistema de SVL_UDF_LOG, assim como o nome da função, nó e fatia correspondentes. O handler do log grava uma linha no SVL_UDF_LOG por mensagem e por fatia. As mensagens são truncadas para 4.096 bytes. O log da UDF é limitado a 500 linhas por fatia. Quando o log fica cheio, o handler do log descarta as mensagens mais antigas e adiciona uma mensagem de alerta em SVL_UDF_LOG.

> **Note**
>
> O manipulador de log UDF do Amazon Redshift escapa caracteres de nova linha (\n), barra vertical (|) e barra invertida (\) com um caractere barra invertida (\).

Por padrão, o nível de log de UDF é definido como WARNING. As mensagens com um nível de log WARNING, ERROR e CRITICAL são registradas. As mensagens com problemas de segurança INFO, DEBUG e NOTSET são ignoradas. Ao definir o nível de log de UDF, use o método do registrador Python. Por exemplo, defina o nível de log como INFO da forma a seguir.

```
logger.setLevel(logging.INFO)
```

Para mais informações sobre como usar o módulo de registro Python, consulte Instalação de registro para Python na documentação Python.

O exemplo a seguir cria uma função chamada f_pyerror que importa o módulo de log Python, instancia o registrador e registra um erro.

```
CREATE OR REPLACE FUNCTION f_pyerror()
RETURNS INTEGER
VOLATILE AS
$$
import logging

logger = logging.getLogger()
logger.setLevel(logging.INFO)
logger.info('Your info message here')
return 0
$$ language plpythonu;
```

O exemplo a seguir consulta SVL_UDF_LOG para visualizar a mensagem registrada no exemplo anterior.

```
select funcname, node, slice, trim(message) as message
from svl_udf_log;

  funcname  | query | node | slice |   message
------------+-------+------+-------+-----------------
  f_pyerror | 12345 |     1|     1 | Your info message here
```

UDFs escalares do Lambda

O Amazon Redshift pode usar funções personalizadas definidas no AWS Lambda como parte de consultas SQL. Você pode escrever UDFs escalares do Lambda em quaisquer linguagens de programação compatíveis com o Lambda, como Java, Go, PowerShell, Node.js, C#, Python e Ruby. Ou você pode usar um tempo de execução personalizado.

O comando CREATE EXTERNAL FUNCTION cria os seguintes parâmetros:

- (Opcional) Uma lista de argumentos com tipo de dados.
- Um tipo de dados de retorno.
- Um nome de função da função externa chamada pelo Amazon Redshift.
- Uma função do IAM que o cluster do Amazon Redshift está autorizado a assumir e chamar para o Lambda.

- Um nome de função do Lambda que o UDF do Lambda chama.

Para obter mais informações sobre CREATE EXTERNAL FUNCTION, consulte CREATE EXTERNAL FUNCTION.

Os tipos de dados de entrada e de retorno para UDFs Lambda podem ser qualquer um destes:

- SMALLINT
- INTEGER
- BIGINT
- DECIMAL
- REAL
- DOUBLE PRECISION
- CHAR
- VARCHAR
- BOOLEAN
- DATE
- TIMESTAMP

Os aliases desses tipos também são válidos. Para ver uma lista completa dos tipos de dados e os respectivos aliases, consulte Tipos de dados.

O Amazon Redshift garante que a função externa possa enviar e receber argumentos e resultados em lote.

As UDFs do Lambda são definidos e gerenciados no Lambda, e você pode controlar os privilégios de acesso para invocar esses UDFs no Amazon Redshift. Você pode invocar várias funções do Lambda na mesma consulta ou invocar a mesma função várias vezes.

Use UDFs do Lambda em quaisquer cláusulas das instruções SQL onde funções escalares são aceitas. Você também pode usar UDFs do Lambda em qualquer instrução SQL, como SELECT, UPDATE, INSERT ou DELETE.

> **Note**
>
> O uso de UDFs do Lambda pode incorrer em custos adicionais do serviço do Lambda. Se ele faz isso depende de fatores como o número de solicitações do Lambda (invocações UDF) e

a duração total da execução do programa Lambda. No entanto, não há custo adicional para usar UDFs do Lambda no Amazon Redshift. Para obter mais informações sobre preço do AWS Lambda, consulte AWS LambdaPreço do .

O número de solicitações do Lambda varia dependendo da cláusula de instrução SQL específica em que o UDF do Lambda é usado. Por exemplo, suponha que a função é usada em uma cláusula WHERE como o seguinte.

```
SELECT a, b FROM t1 WHERE lambda_multiply(a, b) = 64; SELECT a, b
FROM t1 WHERE a*b = lambda_multiply(2, 32)
```

Nesse caso, o Amazon Redshift chama a primeira instrução SELECT para cada uma e chama a segunda instrução SELECT apenas uma vez.

No entanto, usar um UDF na parte de projeção da consulta só pode invocar a função Lambda uma vez para cada linha qualificada ou agregada no conjunto de resultados.

Segurança e permissões de UDFs

Para criar uma UDF do Lambda, você deve ter permissões para uso no LANGUAGE EXFUNC. Você deve conceder explicitamente USAGE ON LANGUAGE EXFUNC ou revogar USAGE ON LANGUAGE EXFUNC a usuários específicos, grupos ou públicos.

O exemplo a seguir concede uso em EXFUNC para PUBLIC.

```
grant usage on language exfunc to PUBLIC;
```

O exemplo a seguir revoga o uso em exfunc de PUBLIC e, em seguida, concede o uso ao grupo de usuários lambda_udf_devs.

```
revoke usage on language exfunc from PUBLIC;
grant usage on language exfunc to group lambda_udf_devs;
```

Para executar uma UDF do Lambda, certifique-se de que você tem permissão para cada função chamada. Por padrão, a permissão para executar novos UDFs do Lambda é concedida a PUBLIC. Para restringir o uso, revogue a execução da função em PUBLIC. Em seguida, conceda o privilégio a usuários ou grupos específicos.

O exemplo a seguir revoga a execução na função exfunc_sum de PUBLIC. Em seguida, ele concede o uso ao grupo de usuários lambda_udf_devs.

```
revoke execute on function exfunc_sum(int, int) from PUBLIC;
grant execute on function exfunc_sum(int, int) to group lambda_udf_devs;
```

Os superusuários têm todos os privilégios por padrão.

Para obter mais informações sobre como conceder e revogar privilégios, consulte GRANT e REVOKE.

Configurar o parâmetro de autorização para UDFs do Lambda

O comando CREATE EXTERNAL FUNCTION requer autorização para invocar funções do Lambda no AWS Lambda. Para iniciar a autorização, especifique uma função do AWS Identity and Access Management (IAM) quando você executa o comando CREATE EXTERNAL FUNCTION. Para obter mais informações sobre funções do IAM, consulte Funções do IAM no Manual do usuário do IAM.

Se houver uma função do IAM existente com permissões para invocar funções do Lambda anexadas ao cluster, então você poderá substituir sua função de nome do recurso da Amazon (ARN) no parâmetro IAM_ROLE do comando. As seções a seguir descrevem as etapas para usar uma função do IAM no comando CREATE EXTERNAL FUNCTION.

Criar uma função do IAM para Lambda

A função do IAM requer permissão para invocar funções do Lambda. Ao criar a função do IAM, forneça a permissão de uma das seguintes maneiras:

- Anexe a política `AWSLambdaRole` na página Anexar política de permissões ao criar uma função do IAM. A política `AWSLambdaRole` concede ao permissões para invocar funções do Lambda, que é o requisito mínimo. Para obter mais informações e outras políticas, consulte Políticas do IAM baseadas em identidade para AWS Lambda no Guia do desenvolvedor do AWS Lambda.
- Crie sua própria política personalizada para anexar à sua função do IAM com a permissão `lambda:InvokeFunction` de todos os recursos ou uma determinada função do Lambda com o ARN dessa função. Para obter informações sobre como criar políticas, consulte Criar políticas do IAM no Manual do usuário do IAM.

  A política de exemplo a seguir permite invocar o Lambda em uma determinada função do Lambda.

  JSON

  ```
  {
  ```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Invoke",
            "Effect": "Allow",
            "Action": [
                "lambda:InvokeFunction"
            ],
            "Resource": "arn:aws:lambda:us-west-2:123456789012:function:my-
function"
        }
    ]
}
```

Para obter mais informações sobre recursos para funções do Lambda, consulte Recursos e condições para ações do Lambda na Referência de API do IAM.

Depois de criar sua política personalizada com as permissões necessárias, você pode anexar sua política à função do IAM na página Anexar política de permissões ao criar uma função do IAM.

Para ver as etapas de como criar um perfil do IAM, consulte "Autorizar o Amazon Redshift a acessar outros serviços da AWS em seu nome" no Guia de gerenciamento de clusters do Amazon Redshift.

Se você não quiser criar uma nova função do IAM, adicione as permissões mencionadas anteriormente à função do IAM existente.

Associar uma função do IAM ao cluster

Anexar a função do IAM ao seu cluster. Você pode adicionar uma função a um cluster ou visualizar as funções associadas a um cluster usando o Console de Gerenciamento do Amazon Redshift, a CLI ou a API. Para obter mais informações, consulte "Autorizar operações COPY, UNLOAD, CREATE EXTERNAL FUNCTION e CREATE EXTERNAL SCHEMA usando funções do IAM" no Guia de gerenciamento de clusters do Amazon Redshift.

Incluir a função do IAM no comando

Inclua o ARN da função do IAM no comando CREATE EXTERNAL FUNCTION. Quando você cria uma função do IAM, o IAM retorna um nome de recurso da Amazon (ARN) para a função. Para especificar uma função do IAM, forneça o ARN da função com o parâmetro `IAM_ROLE`. A seguir é mostrado a sintaxe do parâmetro `IAM_ROLE`.

```
IAM_ROLE 'arn:aws:iam::aws-account-id:role/role-name'
```

Para invocar funções do Lambda que residem em outras contas dentro da mesma região, consulte Encadeando funções do IAM no Amazon Redshift.

Usar a interface JSON entre o Amazon Redshift e AWS Lambda

O Amazon Redshift usa uma interface comum para todas as funções do Lambda com as quais o Amazon Redshift se comunica.

A tabela a seguir mostra a lista de campos de entrada que o Lambda designado funções que você pode esperar para a carga JSON.

Nome do campo	Descrição	Intervalo de valores
request_id	Um identificador universalmente exclusivo (UID) que identifica exclusiva mente cada solicitação de invocar.	Um UUID válido.
cluster	O nome do recurso da Amazon (ARN) completo do cluster.	Um ARN do cluster válido.
usuário	O nome do usuário que faz a chamada.	Um nome de usuário válido.
banco de dados	O nome do banco de dados no qual a consulta está sendo executada.	Um nome do banco de dados válido.
external_ function	O nome totalmente qualificado da função	Um nome de função totalmente qualificado válido.

Nome do campo	Descrição	Intervalo de valores
	externa que faz a chamada.	
query_id	O ID de consulta da consulta que está fazendo a chamada.	Um ID de consulta válido.
num_records	O número de argumentos na carga útil.	Um valor entre 1 e 2^64.
arguments	A carga útil de dados no formato especific ado.	Os dados em formato de array devem ser um array JSON. Cada elemento é um registro que é um array se o número de argumentos é maior do que 1. Usando um array, o Amazon Redshift preserva a ordem dos registros na carga útil.

A ordem do array JSON determina a ordem do processamento em lote. A função do Lambda deve processar os argumentos iterativamente e produzir o número exato de registros. Veja a seguir um exemplo de carga útil.

```
{
  "request_id" : "23FF1F97-F28A-44AA-AB67-266ED976BF40",
  "cluster" : "arn:aws:redshift:xxxx",
  "user" : "adminuser",
  "database" : "db1",
  "external_function": "public.foo",
  "query_id" : 5678234,
  "num_records" : 4,
  "arguments" : [
     [ 1, 2 ],
     [ 3, null],
     null,
     [ 4, 6]
   ]
 }
```

A saída de retorno da função do Lambda contém os campos a seguir.

Nome do campo	Descrição	Intervalo de valores
success	A indicação de sucesso ou falha para a função.	Um valor de `"true"` ou `"false"`.
error_msg	A mensagem de erro se o valor de sucesso for `"false"` (se a função falhar); caso contrário, este campo será ignorado.	Uma mensagem válida.
num_records	O número de registros na carga útil.	Um valor entre 1 e 2^64.
results	Os resultados da chamada no formato especificado.	N/D

Veja a seguir um exemplo de uma saída de função do Lambda.

```
{
  "success": true,   // true indicates the call succeeded
  "error_msg" : "my function isn't working",  // shall only exist when success != true
  "num_records": 4,      // number of records in this payload
  "results" : [
     1,
     4,
     null,
     7
   ]
}
```

Quando você chama funções do Lambda a partir de consultas SQL, o Amazon Redshift garante a segurança da conexão com as seguintes considerações:

- Permissões GRANT e REVOKE. Para ter mais informações sobre segurança e permissões de UDFs, consulte Segurança e permissões de UDFs.
- O Amazon Redshift envia apenas o conjunto mínimo de dados para a função do Lambda designada.
- O Amazon Redshift chama apenas a função do Lambda designada com a função do IAM designada.

Exemplos de caso de uso referentes a funções definidas pelo usuário (UDFs)

> **Note**
>
> A partir de 1.º de novembro de 2025, o Amazon Redshift não permitirá mais a criação de UDFs do Python. As UDFs do Python existentes continuarão a funcionar normalmente até 30 de junho de 2026. A partir de 1.º de julho de 2026, o Amazon Redshift não permitirá mais a criação de UDFs do Python. Recomendamos enfaticamente que você migre as UDFs existentes do Python para as UDFs do Lambda antes de 1.º de novembro de 2025. Para ter mais informações sobre como criar e usar UDFs do Lambda, consulte UDFs escalares do Lambda. Para ter informações sobre a conversão de UDFs do Python existentes em UDFs do Lambda, consulte a publicação de blog .

Você pode usar funções definidas pelo usuário para resolver problemas de negócios integrando o Amazon Redshift com outros componentes. Veja, a seguir, alguns exemplos de como outras pessoas usaram UDFs para seus casos de uso:

- Acessar componentes externos usando UDFs Lambda do Amazon Redshift: descreve como as UDFs Lambda do Amazon Redshift funcionam e mostra o passo a passo da criação de uma UDF Lambda.
- Traduzir e analisar texto usando funções de SQL com o Amazon Redshift, Amazon Translate e Amazon Comprehend: fornece UDFs pré-construídas do Amazon Redshift Lambda que você pode instalar com alguns cliques para traduzir, redigir e analisar campos de texto.
- Acessar o Amazon Location Service do Amazon Redshift: descreve como usar as UDFs Lambda do Amazon Redshift para integração com o Amazon Location Service.

- Tokenização de dados com o Amazon Redshift e o Protegrity: descreve como integrar as UDFs Lambda do Amazon Redshift com o produto Protegrity Serverless.
- UDFs do Amazon Redshift: uma coleção de UDFs SQL, Lambda e Python do Amazon Redshift.

Criar procedimentos armazenados no Amazon Redshift

Este tópico descreve como criar e usar procedimentos armazenados no Amazon Redshift. Um procedimento armazenado é uma coleção de instruções SQL que vários programas podem usar.

Você pode definir um procedimento armazenado do Amazon Redshift usando a linguagem procedural do PostgreSQL PL/pgSQL para realizar um conjunto de consultas SQL e operações lógicas. O procedimento é armazenado no banco de dados e está disponível para qualquer usuário com privilégios de banco de dados suficientes.

Ao contrário de uma função definida pelo usuário (UDF), um procedimento armazenado pode incorporar linguagem de definição de dados (DDL) e linguagem de manipulação de dados (DML) além de consultas SELECT. Um procedimento armazenado não precisa retornar um valor. Use a linguagem processual, incluindo loops e expressões condicionais, para controlar o fluxo lógico.

Para obter detalhes sobre os comandos do SQL para criar e gerenciar procedimentos armazenados, consulte os seguintes tópicos de comandos:

- CREATE PROCEDURE
- ALTER PROCEDURE
- DROP PROCEDURE
- SHOW PROCEDURE
- CALL
- GRANT
- REVOKE
- ALTER DEFAULT PRIVILEGES

Tópicos

- Visão geral dos procedimentos armazenados no Amazon Redshift
- Referência da linguagem PL/pgSQL

Visão geral dos procedimentos armazenados no Amazon Redshift

Este tópico apresenta detalhes sobre a finalidade e o uso dos procedimentos armazenados.

Procedimentos armazenados são comumente usados para encapsular lógica de transformação e validação de dados e lógica empresarial específica. A combinação de várias etapas de SQL em um procedimento armazenado permite reduzir a comunicação entre os aplicativos e o banco de dados.

Para obter controle de acesso granular, você pode criar procedimentos armazenados para executar funções sem precisar conceder ao usuário acesso às tabelas subjacentes. Por exemplo, somente o proprietário ou um superusuário pode truncar uma tabela, e um usuário precisa de privilégios de gravação para inserir dados em uma tabela. Em vez de conceder privilégios a um usuário nas tabelas subjacentes, é possível criar um procedimento armazenado que realiza a tarefa. Depois, conceda ao usuário privilégios para executar o procedimento armazenado.

Um procedimento armazenado com o atributo de segurança DEFINER é executado com os privilégios do proprietário do procedimento armazenado. Por padrão, um procedimento armazenado tem segurança INVOKER, o que significa que o procedimento usa os privilégios do usuário que chama o procedimento.

Para criar um procedimento armazenado, use o comando CREATE PROCEDURE. Para executar um procedimento, use o comando CALL. Exemplos são apresentados posteriormente nessa seção.

Note

Alguns clientes podem exibir o seguinte erro ao criar um procedimento armazenado do Amazon Redshift.

```
ERROR: 42601: [Amazon](500310) unterminated dollar-quoted string at or near "$$
```

Esse erro ocorre devido à impossibilidade do cliente analisar corretamente a instrução CREATE PROCEDURE com instruções delimitadas por ponto e vírgula e com o cifrão ($) entre aspas. Isso resulta em apenas uma parte do extrato enviado ao servidor Amazon Redshift. Geralmente esse erro pode ser resolvido usando a opção `Run as batch` ou `Execute selected` do cliente.
Por exemplo, ao usar um cliente Aginity, use a opção `Run entire script as batch`.
Ao usar SQL Workbench/J, recomendamos a versão 124. Ao usar SQL Workbench/J versão 125, considere a possibilidade de especificar um delimitador alternativo como solução temporária.
CREATE PROCEDURE contém instruções SQL delimitadas por ponto e vírgula (;). Definir um delimitador alternativo, como uma barra (/) e colocá-lo no final da instrução CREATE PROCEDURE envia a instrução inteira ao servidor do Amazon Redshift para processamento. Veja um exemplo a seguir.

```
CREATE OR REPLACE PROCEDURE test()
AS $$
BEGIN
  SELECT 1 a;
END;
$$
LANGUAGE plpgsql
;
/
```

É possível usar um cliente com suporte para analisar instruções CREATE PROCEDURE, como o editor de consultas no console do Amazon Redshift ou o TablePlus.

Tópicos

- Nomeação de procedimentos armazenados
- Segurança e privilégios para procedimentos armazenados
- Exibir um conjunto de resultados de um procedimento armazenado
- Gerenciamento de transações
- Retenção de erros
- Registro em log de procedimentos armazenados
- Limitações dos procedimentos armazenados

O exemplo a seguir mostra um procedimento sem argumentos de saída. Por padrão, os argumentos são de entrada (IN).

```
CREATE OR REPLACE PROCEDURE test_sp1(f1 int, f2 varchar)
AS $$
BEGIN
  RAISE INFO 'f1 = %, f2 = %', f1, f2;
END;
$$ LANGUAGE plpgsql;

call test_sp1(5, 'abc');
INFO: f1 = 5, f2 = abc
CALL
```

> **Note**
>
> Quando você escreve procedimentos armazenados, recomendamos a prática de proteger valores confidenciais:
> Não codifique nenhuma informação confidencial na lógica do procedimento armazenado. Por exemplo, não atribua uma senha de usuário em uma instrução CREATE USER no corpo de um procedimento armazenado. Isso representa um risco de segurança, pois valores codificados podem ser registrados como metadados de esquema nas tabelas de catálogos. Em vez disso, transmita valores confidenciais, como senhas, como argumentos ao procedimento armazenado por meio de parâmetros.
> Para obter mais informações sobre os procedimentos armazenados, consulte CREATE PROCEDURE e Criar procedimentos armazenados no Amazon Redshift. Para obter mais informações sobre tabelas de catálogos, consulte Tabelas de catálogo do sistema.

O exemplo a seguir mostra um procedimento com argumentos de saída. Os argumentos são de entrada (IN), entrada e saída (INOUT) e saída (OUT).

```
CREATE OR REPLACE PROCEDURE test_sp2(f1 IN int, f2 INOUT varchar(256), out_var OUT
 varchar(256))
AS $$
DECLARE
  loop_var int;
BEGIN
  IF f1 is null OR f2 is null THEN
    RAISE EXCEPTION 'input cannot be null';
  END IF;
  DROP TABLE if exists my_etl;
  CREATE TEMP TABLE my_etl(a int, b varchar);
    FOR loop_var IN 1..f1 LOOP
        insert into my_etl values (loop_var, f2);
        f2 := f2 || '+' || f2;
    END LOOP;
  SELECT INTO out_var count(*) from my_etl;
END;
$$ LANGUAGE plpgsql;


call test_sp2(2,'2019');
```

```
          f2           | column2
-----------------------+---------
 2019+2019+2019+2019 | 2
(1 row)
```

Nomeação de procedimentos armazenados

Este tópico apresenta detalhes sobre nomes de procedimentos armazenados.

Se você definir um procedimento com o mesmo nome e diferentes assinaturas ou tipos de dados de argumento de entrada, um novo procedimento será criado. Como resultado, o nome do procedimento será sobrecarregado. Para obter mais informações, consulte Sobrecarga de nomes de procedimento. O Amazon Redshift não habilita a sobrecarga de procedimentos com base em argumentos de saída. Não é possível ter dois procedimentos com o mesmo nome e tipos de dados de argumento de entrada, mas diferentes tipos de argumento de saída.

O proprietário ou um usuário avançado pode substituir o corpo de um procedimento armazenado por um novo com a mesma assinatura. Para alterar a assinatura ou os tipos de retorno de um procedimento armazenado, descarte o procedimento armazenado e crie-o novamente. Para obter mais informações, consulte DROP PROCEDURE e CREATE PROCEDURE.

Você pode evitar potenciais conflitos e resultados inesperados considerando suas convenções de nomenclatura para procedimentos armazenados antes de implementá-los. Como você pode sobrecarregar os nomes de procedimentos, eles podem entrar em conflito com nomes de procedimentos existentes e futuros do Amazon Redshift.

Sobrecarga de nomes de procedimento

Um procedimento é identificado por seu nome e assinatura, que é o número de argumentos de entrada e tipos de dados dos argumentos. Dois procedimentos no mesmo esquema podem ter o mesmo nome se tiverem assinaturas diferentes. Em outras palavras, é possível sobrecarregar nomes de procedimento.

Ao executar um procedimento, o mecanismo de consulta determina qual procedimento chamar com base no número de argumentos fornecidos e nos tipos de dados dos argumentos. Você pode usar o sobrecarregamento para simular procedimentos com um número variável de argumentos até o limite permitido pelo comando CREATE PROCEDURE. Para obter mais informações, consulte CREATE PROCEDURE.

Evitar conflitos de nomenclatura

Recomendamos nomear todos os procedimentos usando o prefixo `sp_`. O Amazon Redshift reserva o prefixo `sp_` para procedimentos armazenados. Ao prefixar seus nomes de procedimento com `sp_`, você garante que seu nome de procedimento não entrará em conflito com nenhum nome de procedimento existente ou futuro do Amazon Redshift.

Segurança e privilégios para procedimentos armazenados

Este tópico descreve as credenciais de banco de dados necessárias para criar e executar procedimentos armazenados.

Por padrão, todos os usuários têm privilégios para criar um procedimento. Para criar um procedimento, é necessário ter o privilégio USAGE na linguagem PL/pgSQL, que é concedida para PUBLIC, por padrão. Por padrão, somente superusuários e proprietários têm privilégio para chamar um procedimento. Usuários avançados podem executar REVOKE USAGE em PL/pgSQL de um usuário se quiserem impedir que o usuário crie um procedimento armazenado.

Para chamar um procedimento, é necessário receber privilégio para realizar EXECUTE no procedimento. Por padrão, o privilégio para realizar EXECUTE em novos procedimentos é concedido ao proprietário do procedimento e a superusuários. Para obter mais informações, consulte GRANT.

O usuário que cria um procedimento é o proprietário, por padrão. O proprietário possui os privilégios CREATE, DROP e EXECUTE sobre o procedimento, por padrão. Os usuários avançados têm todos os privilégios.

O atributo SECURITY controla os privilégios de um procedimento para acessar objetos do banco de dados. Ao criar um procedimento armazenado, é possível definir o atributo SECURITY como DEFINER ou INVOKER. Esse atributo determina quais privilégios são usados ao executar as instruções no corpo do procedimento armazenado. Se você especificar SECURITY INVOKER, o procedimento usará os privilégios do usuário que invoca o procedimento. Se você especificar SECURITY DEFINER, o procedimento usará os privilégios do proprietário do procedimento. INVOKER é o padrão.

Como um procedimento SECURITY DEFINER é executado com os privilégios do usuário que é proprietário dele, você deve garantir que o procedimento não seja mal utilizado. Para garantir que procedimentos SECURITY DEFINER não sejam mal utilizados, faça o seguinte:

- Conceda EXECUTE em procedimentos SECURITY DEFINER para usuários específicos em vez de PUBLIC.

- Qualifique todos os objetos do banco de dados que o procedimento precisa acessar com os nomes de esquema. Por exemplo, use `myschema.mytable`, em vez de apenas `mytable`.
- Se não for possível qualificar um nome de objeto pelo seu esquema, defina `search_path` ao criar o procedimento usando a opção SET. Defina `search_path` para excluir quaisquer esquemas que sejam graváveis por usuários não confiáveis. Essa abordagem impede que qualquer chamador desse procedimento crie objetos (por exemplo, tabelas ou visualizações) que mascaram objetos destinados ao uso pelo procedimento. Para obter mais informações sobre a opção SET, consulte CREATE PROCEDURE.

O exemplo a seguir define `search_path` como `admin` para garantir que a tabela `user_creds` seja acessada pelo esquema `admin`, e não pelo esquema público ou por qualquer outro esquema no `search_path` do chamador.

```
CREATE OR REPLACE PROCEDURE sp_get_credentials(userid int, o_creds OUT varchar)
AS $$
BEGIN
  SELECT creds INTO o_creds
  FROM user_creds
  WHERE user_id = $1;
END;
$$ LANGUAGE plpgsql
SECURITY DEFINER
-- Set a secure search_path
SET search_path = admin;
```

Exibir um conjunto de resultados de um procedimento armazenado

Este tópico descreve como os procedimentos armazenados exibem dados.

É possível retornar um conjunto de resultados usando um cursor ou uma tabela temporária.

Retorno de um cursor

Para retornar um cursor, crie um procedimento com um argumento INOUT definido com um tipo de dados `refcursor`. Ao chamar o procedimento, dê um nome ao cursor. Depois, você pode buscar os resultados do cursor pelo nome.

O exemplo a seguir cria um procedimento chamado `get_result_set` com um argumento INOUT chamado `rs_out` usando o tipo de dados `refcursor`. O procedimento abre o cursor usando uma instrução SELECT.

```
CREATE OR REPLACE PROCEDURE get_result_set (param IN integer, rs_out INOUT refcursor)
AS $$
BEGIN
  OPEN rs_out FOR SELECT * FROM fact_tbl where id >= param;
END;
$$ LANGUAGE plpgsql;
```

O comando CALL a seguir abre o cursor com o nome `mycursor`. Use cursores somente nas transações.

```
BEGIN;
CALL get_result_set(1, 'mycursor');
```

Depois que o cursor for aberto, você pode buscar a partir do cursor, conforme mostram os exemplos a seguir.

```
FETCH ALL FROM mycursor;

    id | secondary_id | name
-------+--------------+---------
     1 |            1 | Joe
     1 |            2 | Ed
     2 |            1 | Mary
     1 |            3 | Mike
(4 rows)
```

Ao final, a transação é confirmada ou revertida.

```
COMMIT;
```

Um cursor retornado por um procedimento armazenado está sujeito às mesmas restrições e considerações de performance, conforme descrito em DECLARE CURSOR. Para obter mais informações, consulte Restrições de cursor.

O exemplo a seguir mostra a chamada do procedimento armazenado `get_result_set` usando um tipo de dados `refcursor` a partir do JDBC. O `'mycursor'` literal (o nome do cursor) é enviado para a `prepareStatement`. Depois, os resultados são obtidos do `ResultSet`.

```
static void refcursor_example(Connection conn) throws SQLException {
```

```
    conn.setAutoCommit(false);
    PreparedStatement proc = conn.prepareStatement("CALL get_result_set(1,
 'mycursor')");
    proc.execute();
    ResultSet rs = statement.executeQuery("fetch all from mycursor");
    while (rs.next()) {
      int n = rs.getInt(1);
      System.out.println("n " + n);
    }
```

Uso de uma tabela temporária

Para retornar resultados, você pode retornar um identificador para uma tabela temporária que contém linhas de resultados. O cliente pode fornecer um nome como um parâmetro para o procedimento armazenado. Dentro do procedimento armazenado, SQL dinâmico pode ser usado para operar na tabela temporária. Por exemplo:

```
CREATE PROCEDURE get_result_set(param IN integer, tmp_name INOUT varchar(256)) as $$
DECLARE
  row record;
BEGIN
  EXECUTE 'drop table if exists ' || tmp_name;
  EXECUTE 'create temp table ' || tmp_name || ' as select * from fact_tbl where id <= '
 || param;
END;
$$ LANGUAGE plpgsql;

CALL get_result_set(2, 'myresult');
 tmp_name
-----------
 myresult
(1 row)

SELECT * from myresult;
 id | secondary_id | name
----+--------------+------
  1 |            1 | Joe
  2 |            1 | Mary
  1 |            2 | Ed
  1 |            3 | Mike
(4 rows)
```

Gerenciamento de transações

Você pode criar um procedimento armazenado com comportamento padrão de gerenciamento de transações ou comportamento não atômico.

Gerenciamento de transações de procedimentos armazenados no modo padrão

O comportamento de confirmação automática do modo de transações padrão faz com que cada comando SQL executado separadamente seja confirmado individualmente. Uma chamada para um procedimento armazenado é tratada como um único comando SQL. As instruções SQL dentro de um procedimento se comportam como se estivessem em um bloco de transações iniciado implicitamente quando a chamada começa, e que termina quando a chamada é encerrada. Uma chamada aninhada para outro procedimento é tratada como qualquer outra instrução SQL e opera dentro do contexto da mesma transação que o chamador. Para obter mais informações sobre o comportamento de confirmação automática, consulte Níveis de isolamento no Amazon Redshift.

Porém, suponha que você chame um procedimento armazenado a partir de um bloco de transações especificado pelo usuário (definido por BEGIN... COMMIT). Neste caso, todas as instruções do procedimento armazenado são executadas no contexto da transação especificada pelo usuário. O procedimento não é confirmado implicitamente na saída. O chamador controla a confirmação ou a reversão do procedimento.

Se um erro for encontrado durante a execução de um procedimento armazenado, todas as alterações feitas na transação atual são revertidas.

Você pode usar as seguintes instruções de controle de transação em um procedimento armazenado:

- COMMIT - Confirma todo o trabalho realizado na transação atual e inicia implicitamente uma nova transação. Para obter mais informações, consulte COMMIT.
- ROLLBACK - Reverte o trabalho realizado na transação atual e inicia implicitamente uma nova transação. Para obter mais informações, consulte ROLLBACK.

TRUNCATE é outra instrução que pode ser emitida em um procedimento armazenado e influencia o gerenciamento de transações. No Amazon Redshift, TRUNCATE emite uma confirmação implicitamente. Esse comportamento permanece o mesmo no contexto dos procedimentos armazenados. Quando uma instrução TRUNCATE for emitida a partir de um procedimento armazenado, ela confirma a transação atual e inicia uma nova. Para obter mais informações, consulte TRUNCATE.

Todas as instruções que seguem uma instrução COMMIT, ROLLBACK ou TRUNCATE no contexto de uma nova transação. Elas fazem isso até uma instrução COMMIT, ROLLBACK ou TRUNCATE ser encontrada ou o procedimento armazenado ser encerrado.

Ao usar uma instrução COMMIT, ROLLBACK ou TRUNCATE em um procedimento armazenado, as seguintes restrições são aplicadas:

- Se o procedimento armazenado for chamado em um bloco de transação, ele não poderá emitir uma instrução COMMIT, ROLLBACK ou TRUNCATE. Essa restrição se aplica ao corpo do próprio procedimento armazenado e a qualquer chamada de procedimento aninhada.
- Se o procedimento armazenado for criado com opções `SET config`, ele não poderá emitir uma instrução COMMIT, ROLLBACK ou TRUNCATE. Essa restrição se aplica ao corpo do próprio procedimento armazenado e a qualquer chamada de procedimento aninhada.
- Qualquer cursor aberto (explícita ou implicitamente) será fechado automaticamente quando uma instrução COMMIT, ROLLBACK ou TRUNCATE for processada. Para obter as restrições sobre cursores explícitos ou implícitos, consulte Limitações dos procedimentos armazenados.

Além disso, não é possível executar COMMIT ou ROLLBACK com SQL dinâmico. No entanto, é possível executar TRUNCATE com SQL dinâmico. Para obter mais informações, consulte SQL dinâmico.

Ao trabalhar com procedimentos armazenados, considere que as instruções BEGIN e END em PL/pgSQL são somente para agrupamento. Elas não iniciam ou encerram uma transação. Para obter mais informações, consulte Bloquear.

O exemplo a seguir demonstra o comportamento da transação ao chamar um procedimento armazenado a partir de um bloco de transações explícito. As duas instruções de inserção emitidas fora do procedimento armazenado e aquela emitida dentro dele fazem parte da mesma transação (3382). A transação é confirmada quando o usuário emite a confirmação explícita.

```
CREATE OR REPLACE PROCEDURE sp_insert_table_a(a int) LANGUAGE plpgsql
AS $$
BEGIN
  INSERT INTO test_table_a values (a);
END;
$$;

Begin;
  insert into test_table_a values (1);
```

```
  Call sp_insert_table_a(2);
  insert into test_table_a values (3);
Commit;

select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

 userid | xid  | pid |  type   |               stmt_text
--------+------+-----+---------+----------------------------------------
    103 | 3382 | 599 | UTILITY | Begin;
    103 | 3382 | 599 | QUERY   | insert into test_table_a values (1);
    103 | 3382 | 599 | UTILITY | Call sp_insert_table_a(2);
    103 | 3382 | 599 | QUERY   | INSERT INTO test_table_a values ( $1 )
    103 | 3382 | 599 | QUERY   | insert into test_table_a values (3);
    103 | 3382 | 599 | UTILITY | COMMIT
```

Por outro lado, pense em quando as mesmas instruções forem emitidas fora de um bloco de transações explícito e a sessão tiver a confirmação automática definida como ON. Nesse caso, cada instrução é executada em sua própria transação.

```
insert into test_table_a values (1);
Call sp_insert_table_a(2);
insert into test_table_a values (3);

select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

 userid | xid  | pid |  type   |
            stmt_text
--------+------+-----+---------
+----------------------------------------------------------------------------------------------------
    103 | 3388 | 599 | QUERY   | insert into test_table_a values (1);
    103 | 3388 | 599 | UTILITY | COMMIT
    103 | 3389 | 599 | UTILITY | Call sp_insert_table_a(2);
    103 | 3389 | 599 | QUERY   | INSERT INTO test_table_a values ( $1 )
    103 | 3389 | 599 | UTILITY | COMMIT
    103 | 3390 | 599 | QUERY   | insert into test_table_a values (3);
    103 | 3390 | 599 | UTILITY | COMMIT
```

O exemplo a seguir emite uma instrução TRUNCATE depois de inserir na `test_table_a`. A instrução TRUNCATE emite uma confirmação implícita que confirma a transação atual (3335) e inicia uma nova (3336). A nova transação é confirmada quando o procedimento é encerrado.

```
CREATE OR REPLACE PROCEDURE sp_truncate_proc(a int, b int) LANGUAGE plpgsql
AS $$
BEGIN
  INSERT INTO test_table_a values (a);
  TRUNCATE test_table_b;
  INSERT INTO test_table_b values (b);
END;
$$;

Call sp_truncate_proc(1,2);

select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

 userid | xid  |  pid  |  type   |
                                        stmt_text
--------+------+-------+---------
+-----------------------------------------------------------------------------------------------
    103 | 3335 | 23636 | UTILITY | Call sp_truncate_proc(1,2);
    103 | 3335 | 23636 | QUERY   | INSERT INTO test_table_a values ( $1 )
    103 | 3335 | 23636 | UTILITY | TRUNCATE test_table_b
    103 | 3335 | 23636 | UTILITY | COMMIT
    103 | 3336 | 23636 | QUERY   | INSERT INTO test_table_b values ( $1 )
    103 | 3336 | 23636 | UTILITY | COMMIT
```

O exemplo a seguir emite uma TRUNCATE a partir de uma chamada aninhada. A TRUNCATE confirma todo o trabalho feito até agora nos procedimentos externo e interno em uma transação (3344). Ela inicia uma nova transação (3345). A nova transação é confirmada quando o procedimento externo é encerrado.

```
CREATE OR REPLACE PROCEDURE sp_inner(c int, d int) LANGUAGE plpgsql
AS $$
BEGIN
  INSERT INTO inner_table values (c);
  TRUNCATE outer_table;
  INSERT INTO inner_table values (d);
END;
```

```
$$;

CREATE OR REPLACE PROCEDURE sp_outer(a int, b int, c int, d int) LANGUAGE plpgsql
AS $$
BEGIN
  INSERT INTO outer_table values (a);
  Call sp_inner(c, d);
  INSERT INTO outer_table values (b);
END;
$$;

Call sp_outer(1, 2, 3, 4);

select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

 userid | xid  |  pid  |  type   |
                                        stmt_text
--------+------+-------+---------
+-----------------------------------------------------------------------------------------------------
    103 | 3344 | 23636 | UTILITY | Call sp_outer(1, 2, 3, 4);
    103 | 3344 | 23636 | QUERY   | INSERT INTO outer_table values ( $1 )
    103 | 3344 | 23636 | UTILITY | CALL sp_inner( $1 , $2 )
    103 | 3344 | 23636 | QUERY   | INSERT INTO inner_table values ( $1 )
    103 | 3344 | 23636 | UTILITY | TRUNCATE outer_table
    103 | 3344 | 23636 | UTILITY | COMMIT
    103 | 3345 | 23636 | QUERY   | INSERT INTO inner_table values ( $1 )
    103 | 3345 | 23636 | QUERY   | INSERT INTO outer_table values ( $1 )
    103 | 3345 | 23636 | UTILITY | COMMIT
```

O exemplo a seguir mostra que o cursor `cur1` foi fechado quando a instrução TRUNCATE foi confirmada.

```
CREATE OR REPLACE PROCEDURE sp_open_cursor_truncate()
LANGUAGE plpgsql
AS $$
DECLARE
  rec RECORD;
  cur1 cursor for select * from test_table_a order by 1;
BEGIN
  open cur1;
  TRUNCATE table test_table_b;
```

```
  Loop
    fetch cur1 into rec;
    raise info '%', rec.c1;
    exit when not found;
  End Loop;
END
$$;

call sp_open_cursor_truncate();
ERROR: cursor "cur1" does not exist
CONTEXT: PL/pgSQL function "sp_open_cursor_truncate" line 8 at fetch
```

O exemplo a seguir emite uma instrução TRUNCATE e não pode ser chamado dentro de um bloco de transações explícito.

```
CREATE OR REPLACE PROCEDURE sp_truncate_atomic() LANGUAGE plpgsql
AS $$
BEGIN
  TRUNCATE test_table_b;
END;
$$;

Begin;
  Call sp_truncate_atomic();
ERROR: TRUNCATE cannot be invoked from a procedure that is executing in an atomic
 context.
HINT: Try calling the procedure as a top-level call i.e. not from within an explicit
 transaction block.
Or, if this procedure (or one of its ancestors in the call chain) was created with SET
 config options, recreate the procedure without them.
CONTEXT: SQL statement "TRUNCATE test_table_b"
PL/pgSQL function "sp_truncate_atomic" line 2 at SQL statement
```

O exemplo a seguir mostra que um usuário que não é um superusuário ou proprietário de uma tabela pode emitir uma instrução TRUNCATE na tabela. O usuário faz isso usando um procedimento `Security Definer` armazenado. O exemplo mostra as seguintes ações:

- O user1 cria a tabela `test_tbl`.
- O user1 cria o procedimento armazenado `sp_truncate_test_tbl`.
- O user1 concede privilégio `EXECUTE` no procedimento armazenado para user2.

- O user2 executa o procedimento armazenado para truncar a tabela `test_tbl`. O exemplo mostra a contagem de linhas antes e depois do comando TRUNCATE.

```
set session_authorization to user1;
create table test_tbl(id int, name varchar(20));
insert into test_tbl values (1,'john'), (2, 'mary');
CREATE OR REPLACE PROCEDURE sp_truncate_test_tbl() LANGUAGE plpgsql
AS $$
DECLARE
  tbl_rows int;
BEGIN
  select count(*) into tbl_rows from test_tbl;
  RAISE INFO 'RowCount before Truncate: %', tbl_rows;
  TRUNCATE test_tbl;
  select count(*) into tbl_rows from test_tbl;
  RAISE INFO 'RowCount after Truncate: %', tbl_rows;
END;
$$ SECURITY DEFINER;
grant execute on procedure sp_truncate_test_tbl() to user2;
reset session_authorization;


set session_authorization to user2;
call sp_truncate_test_tbl();
INFO:  RowCount before Truncate: 2
INFO:  RowCount after Truncate: 0
CALL
reset session_authorization;
```

O exemplo a seguir emite COMMIT duas vezes. O primeiro COMMIT confirma todo o trabalho feito na transação 10363 e começa implicitamente a transação 10364. A transação 10364 será confirmada pela segunda instrução COMMIT.

```
CREATE OR REPLACE PROCEDURE sp_commit(a int, b int) LANGUAGE plpgsql
AS $$
BEGIN
  INSERT INTO test_table values (a);
  COMMIT;
  INSERT INTO test_table values (b);
  COMMIT;
END;
```

```
$$;

call sp_commit(1,2);

select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;
 userid |  xid  | pid  |  type   |
                               stmt_text
--------+-------+------+---------
+-----------------------------------------------------------------------------------------------
    100 | 10363 | 3089 | UTILITY | call sp_commit(1,2);
    100 | 10363 | 3089 | QUERY   | INSERT INTO test_table values ( $1 )
    100 | 10363 | 3089 | UTILITY | COMMIT
    100 | 10364 | 3089 | QUERY   | INSERT INTO test_table values ( $1 )
    100 | 10364 | 3089 | UTILITY | COMMIT
```

O exemplo a seguir emitirá uma instrução ROLLBACK se `sum_vals` for maior do que 2. A primeira instrução ROLLBACK reverte todo o trabalho feito na transação 10377 e começa uma nova transação 10378. A transação 10378 é confirmada quando o procedimento é encerrado.

```
CREATE OR REPLACE PROCEDURE sp_rollback(a int, b int) LANGUAGE plpgsql
AS $$
DECLARE
  sum_vals int;
BEGIN
  INSERT INTO test_table values (a);
  SELECT sum(c1) into sum_vals from test_table;
  IF sum_vals > 2 THEN
    ROLLBACK;
  END IF;

  INSERT INTO test_table values (b);
END;
$$;

call sp_rollback(1, 2);

select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;
```

```
userid |  xid  | pid  |  type   |
                              stmt_text
--------+-------+------+---------
+-----------------------------------------------------------------------------------------------
    100 | 10377 | 3089 | UTILITY | call sp_rollback(1, 2);
    100 | 10377 | 3089 | QUERY   | INSERT INTO test_table values ( $1 )
    100 | 10377 | 3089 | QUERY   | SELECT sum(c1) from test_table
    100 | 10377 | 3089 | QUERY   | Undoing 1 transactions on table 133646 with current
 xid 10377 : 10377
    100 | 10378 | 3089 | QUERY   | INSERT INTO test_table values ( $1 )
    100 | 10378 | 3089 | UTILITY | COMMIT
```

Gerenciamento de transações de procedimentos armazenados no modo não atômico

Um procedimento armazenado criado no modo NONATOMIC tem um comportamento de controle de transações diferente de um procedimento criado no modo padrão. Semelhante ao comportamento de confirmação automática dos comandos SQL fora dos procedimentos armazenados, cada instrução SQL dentro de um procedimento NONATOMIC é executada em sua própria transação e é confirmada automaticamente. Se um usuário iniciar um bloco de transação explícito em um procedimento armazenado NONATOMIC, as instruções SQL dentro do bloco não serão confirmadas automaticamente. O bloco de transação controla a confirmação ou a reversão das instruções contidas nele.

Em procedimentos armazenados NONATOMIC, você pode abrir um bloco de transação explícito dentro do procedimento usando a instrução START TRANSACTION. No entanto, se já houver um bloco de transação aberto, essa instrução não fará nada porque o Amazon Redshift não é compatível com subtransações. A transação anterior continua.

Ao trabalhar com loops FOR do cursor dentro de um procedimento NONATOMIC, certifique-se de abrir um bloco de transação explícito antes de percorrer os resultados de uma consulta. Caso contrário, o cursor será fechado quando a instrução SQL dentro do loop for confirmada automaticamente.

Algumas considerações ao usar o comportamento do modo NONATOMIC são as seguintes:

- Cada instrução SQL dentro do procedimento armazenado será confirmada automaticamente se não houver um bloco de transação aberto e se a confirmação automática estiver ativada na sessão.

- Você pode emitir uma instrução COMMIT/ROLLBACK/TRUNCATE para encerrar a transação se o procedimento armazenado for chamado de dentro de um bloco de transação. Isso não é possível no modo padrão.
- Você pode emitir uma instrução START TRANSACTION para iniciar um bloco de transação dentro do procedimento armazenado.

Os exemplos a seguir demonstram o comportamento da transação ao trabalhar com procedimentos armazenados NONATOMIC. A sessão para todos os exemplos a seguir está com a confirmação automática ativada.

No exemplo a seguir, um procedimento armazenado NONATOMIC tem duas instruções INSERT. Quando o procedimento é chamado fora de um bloco de transação, cada instrução INSERT dentro do procedimento é confirmada automaticamente.

```
CREATE TABLE test_table_a(v int);
CREATE TABLE test_table_b(v int);

CREATE OR REPLACE PROCEDURE sp_nonatomic_insert_table_a(a int, b int) NONATOMIC AS
$$
BEGIN
    INSERT INTO test_table_a values (a);
    INSERT INTO test_table_b values (b);
END;
$$
LANGUAGE plpgsql;

Call sp_nonatomic_insert_table_a(1,2);

Select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

 userid | xid  |    pid     |  type   |               stmt_text
--------+------+------------+---------+----------------------------------------
      1 | 1792 | 1073807554 | UTILITY | Call sp_nonatomic_insert_table_a(1,2);
      1 | 1792 | 1073807554 | QUERY   | INSERT INTO test_table_a values ( $1 )
      1 | 1792 | 1073807554 | UTILITY | COMMIT
      1 | 1793 | 1073807554 | QUERY   | INSERT INTO test_table_b values ( $1 )
      1 | 1793 | 1073807554 | UTILITY | COMMIT
(5 rows)
```

No entanto, quando o procedimento é chamado de dentro de um bloco BEGIN…COMMIT, todas as instruções fazem parte da mesma transação (xid=1799).

```
Begin;
  INSERT INTO test_table_a values (10);
  Call sp_nonatomic_insert_table_a(20,30);
  INSERT INTO test_table_b values (40);
Commit;

Select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

 userid | xid  |    pid     |  type   |                stmt_text
--------+------+------------+---------+-----------------------------------------
      1 | 1799 | 1073914035 | UTILITY | Begin;
      1 | 1799 | 1073914035 | QUERY   | INSERT INTO test_table_a values (10);
      1 | 1799 | 1073914035 | UTILITY | Call sp_nonatomic_insert_table_a(20,30);
      1 | 1799 | 1073914035 | QUERY   | INSERT INTO test_table_a values ( $1 )
      1 | 1799 | 1073914035 | QUERY   | INSERT INTO test_table_b values ( $1 )
      1 | 1799 | 1073914035 | QUERY   | INSERT INTO test_table_b values (40);
      1 | 1799 | 1073914035 | UTILITY | COMMIT
(7 rows)
```

Neste exemplo, duas instruções INSERT estão entre START TRANSACTION...COMMIT. Quando o procedimento é chamado fora de um bloco de transação, as duas instruções INSERT estão na mesma transação (xid=1866).

```
CREATE OR REPLACE PROCEDURE sp_nonatomic_txn_block(a int, b int) NONATOMIC AS
$$
BEGIN
    START TRANSACTION;
    INSERT INTO test_table_a values (a);
    INSERT INTO test_table_b values (b);
    COMMIT;
END;
$$
LANGUAGE plpgsql;

Call sp_nonatomic_txn_block(1,2);

Select userid, xid, pid, type, trim(text) as stmt_text
```

```
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

 userid | xid  |    pid     |  type   |               stmt_text
--------+------+------------+---------+----------------------------------------
      1 | 1865 | 1073823998 | UTILITY | Call sp_nonatomic_txn_block(1,2);
      1 | 1866 | 1073823998 | QUERY   | INSERT INTO test_table_a values ( $1 )
      1 | 1866 | 1073823998 | QUERY   | INSERT INTO test_table_b values ( $1 )
      1 | 1866 | 1073823998 | UTILITY | COMMIT
(4 rows)
```

Quando o procedimento é chamado de dentro de um bloco BEGIN...COMMIT, o comando START TRANSACTION dentro do procedimento não faz nada porque já existe uma transação aberta. O comando COMMIT dentro do procedimento confirma a transação atual (xid=1876) e inicia uma nova.

```
Begin;
  INSERT INTO test_table_a values (10);
  Call sp_nonatomic_txn_block(20,30);
  INSERT INTO test_table_b values (40);
Commit;

Select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

 userid | xid  |    pid     |  type   |               stmt_text
--------+------+------------+---------+----------------------------------------
      1 | 1876 | 1073832133 | UTILITY | Begin;
      1 | 1876 | 1073832133 | QUERY   | INSERT INTO test_table_a values (10);
      1 | 1876 | 1073832133 | UTILITY | Call sp_nonatomic_txn_block(20,30);
      1 | 1876 | 1073832133 | QUERY   | INSERT INTO test_table_a values ( $1 )
      1 | 1876 | 1073832133 | QUERY   | INSERT INTO test_table_b values ( $1 )
      1 | 1876 | 1073832133 | UTILITY | COMMIT
      1 | 1878 | 1073832133 | QUERY   | INSERT INTO test_table_b values (40);
      1 | 1878 | 1073832133 | UTILITY | COMMIT
(8 rows)
```

Este exemplo mostra como trabalhar com loops de cursor. A tabela test_table_a tem três valores. O objetivo é percorrer os três valores e inseri-los na tabela test_table_b. Se um procedimento armazenado NONATOMIC for criado conforme mostrado a seguir, será gerado um erro informando que o cursor “cur1" não existe depois da execução da instrução INSERT no primeiro loop. Isso ocorre porque a confirmação automática de INSERT fecha o cursor aberto.

```
insert into test_table_a values (1), (2), (3);

CREATE OR REPLACE PROCEDURE sp_nonatomic_cursor() NONATOMIC
LANGUAGE plpgsql
AS $$
DECLARE
  rec RECORD;
  cur1 cursor for select * from test_table_a order by 1;
BEGIN
  open cur1;
  Loop
    fetch cur1 into rec;
    exit when not found;
    raise info '%', rec.v;
    insert into test_table_b values (rec.v);
  End Loop;
END
$$;

CALL sp_nonatomic_cursor();

INFO:  1
ERROR:  cursor "cur1" does not exist
CONTEXT:  PL/pgSQL function "sp_nonatomic_cursor" line 7 at fetch
```

Para que o loop do cursor funcione, coloque-o entre START TRANSACTION...COMMIT.

```
insert into test_table_a values (1), (2), (3);

CREATE OR REPLACE PROCEDURE sp_nonatomic_cursor() NONATOMIC
LANGUAGE plpgsql
AS $$
DECLARE
  rec RECORD;
  cur1 cursor for select * from test_table_a order by 1;
BEGIN
  START TRANSACTION;
  open cur1;
  Loop
    fetch cur1 into rec;
    exit when not found;
    raise info '%', rec.v;
    insert into test_table_b values (rec.v);
```

```
  End Loop;
  COMMIT;
END
$$;

CALL sp_nonatomic_cursor();

INFO:  1
INFO:  2
INFO:  3
CALL
```

Retenção de erros

Este tópico descreve como o Amazon Redshift lida com erros.

Quando uma consulta ou comando em um procedimento armazenado gera um erro, as consultas subsequentes não são executadas e a transação é revertida. Mas você pode tratar os erros usando um bloco EXCEPTION.

> **Note**
>
> O comportamento padrão é que um erro fará com que as consultas subsequentes não sejam executadas, mesmo quando não há outras condições de geração de erros no procedimento armazenado.

```
[ <<label>> ]
[ DECLARE
  declarations ]
BEGIN
  statements
EXCEPTION
  WHEN OTHERS THEN
    statements
END;
```

Quando ocorre uma exceção e você adiciona um bloco de tratamento de exceções, é possível gravar instruções RAISE e a maioria das outras instruções PL/PGSQL. Por exemplo, é possível levantar uma exceção com uma mensagem personalizada ou inserir um registro em uma tabela de registro.

Ao inserir o bloco de tratamento de exceções, a transação atual é revertida, e cria-se uma nova transação para executar as instruções no bloco. Se as instruções no bloco forem executadas sem erros, a transação será confirmada, e a exceção será lançada novamente. Por fim, o procedimento armazenado é encerrado.

A única condição compatível em um bloco de exceção é OTHERS, que corresponde a todos os tipos de erros, exceto o cancelamento de consulta. Além disso, se ocorrer um erro em um bloco de tratamento de exceções, ele poderá ser detectado por um bloco de tratamento de exceções externo.

Quando ocorre um erro dentro do procedimento NONATOMIC, o erro não será relançado se for tratado por um bloco de exceções. Consulte a instrução PL/pgSQL `RAISE` para lançar uma exceção capturada pelo bloco de tratamento de exceções. Essa instrução só é válida em blocos de tratamento de exceções. Para obter mais informações, consulte RAISE.

Controle do que acontecerá depois de um erro em um procedimento armazenado, com o manipulador CONTINUE

O manipulador `CONTINUE` é um tipo de manipulador de exceções que controla o fluxo de execuções em um procedimento armazenado NONATOMIC. Ao usá-lo, você pode capturar e processar exceções sem encerrar o bloco de instruções existente. Normalmente, quando ocorre um erro em um procedimento armazenado, o fluxo é interrompido, e o erro é retornado ao chamador. No entanto, em alguns casos de uso, a condição de erro não é grave o suficiente para justificar a interrupção do fluxo. Convém processar o erro normalmente, usando a lógica de tratamento de erros de sua preferência em uma transação à parte e, em seguida, continuar executando as instruções que seguem o erro. Esta é a sintaxe.

```
[ DECLARE
  declarations ]
BEGIN
  statements
EXCEPTION
  [ CONTINUE_HANDLER | EXIT_HANDLER ] WHEN OTHERS THEN
    handler_statements
END;
```

Há várias tabelas do sistema disponíveis para ajudar você a coletar informações sobre vários tipos de erros. Para obter mais informações, consulte STL_LOAD_ERRORS, STL_ERROR e SYS_STREAM_SCAN_ERRORS. Também há tabelas de sistema adicionais que você pode usar para solucionar erros. Mais informações sobre elas podem ser encontradas em Referência de visualizações e tabelas do sistema.

Exemplo

O exemplo a seguir mostra como gravar instruções no bloco de tratamento de exceções. O procedimento armazenado está usando o comportamento padrão de gerenciamento de transações.

```
CREATE TABLE employee (firstname varchar, lastname varchar);
INSERT INTO employee VALUES ('Tomas','Smith');
CREATE TABLE employee_error_log (message varchar);

CREATE OR REPLACE PROCEDURE update_employee_sp() AS
$$
BEGIN
    UPDATE employee SET firstname = 'Adam' WHERE lastname = 'Smith';
    EXECUTE 'select invalid';
EXCEPTION WHEN OTHERS THEN
    RAISE INFO 'An exception occurred.';
    INSERT INTO employee_error_log VALUES ('Error message: ' || SQLERRM);
END;
$$
LANGUAGE plpgsql;

CALL update_employee_sp();

INFO:  An exception occurred.
ERROR:  column "invalid" does not exist
CONTEXT:  SQL statement "select invalid"
PL/pgSQL function "update_employee_sp" line 3 at execute statement
```

Neste exemplo, se você chamar `update_employee_sp`, a mensagem informativa Ocorreu uma exceção. será gerada e a mensagem de erro será inserida no log `employee_error_log` da tabela de registro. A exceção original é lançada novamente antes que o procedimento armazenado seja encerrado. As consultas a seguir mostram registros resultantes da execução do exemplo.

```
SELECT * from employee;

firstname | lastname
----------+----------
 Tomas     | Smith

SELECT * from employee_error_log;

          message
```

```
-----------------------------------------------
 Error message: column "invalid" does not exist
```

Para obter mais informações sobre RAISE, inclusive ajuda para formatação e uma lista dos níveis adicionais, consulte Instruções da PL/pgSQL compatíveis.

O exemplo a seguir mostra como gravar instruções no bloco de tratamento de exceções. O procedimento armazenado está usando o comportamento NONATOMIC de gerenciamento de transações. Neste exemplo, nenhum erro é revertido ao chamador depois da conclusão da chamada do procedimento. A instrução UPDATE não é revertida devido ao erro na próxima instrução. A mensagem informativa é gerada e a mensagem de erro é inserida na tabela de log.

```
CREATE TABLE employee (firstname varchar, lastname varchar);
INSERT INTO employee VALUES ('Tomas','Smith');
CREATE TABLE employee_error_log (message varchar);

-- Create the SP in NONATOMIC mode
CREATE OR REPLACE PROCEDURE update_employee_sp_2() NONATOMIC AS
$$
BEGIN
    UPDATE employee SET firstname = 'Adam' WHERE lastname = 'Smith';
    EXECUTE 'select invalid';
EXCEPTION WHEN OTHERS THEN
    RAISE INFO 'An exception occurred.';
    INSERT INTO employee_error_log VALUES ('Error message: ' || SQLERRM);
END;
$$
LANGUAGE plpgsql;

CALL update_employee_sp_2();
INFO:  An exception occurred.
CALL

SELECT * from employee;

 firstname | lastname
-----------+----------
 Adam      | Smith
(1 row)

SELECT * from employee_error_log;

                    message
```

```
-----------------------------------------------
 Error message: column "invalid" does not exist
(1 row)
```

Este exemplo mostra como criar um procedimento armazenado em dois sub-blocos. Quando o procedimento armazenado é chamado, o erro do primeiro sub-bloco é tratado por seu bloco de tratamento de exceções. Após a conclusão do primeiro sub-bloco, o procedimento continua executando o segundo sub-bloco. Pelo resultado, podemos ver que nenhum erro é gerado quando a chamada do procedimento é concluída. As operações UPDATE e INSERT na tabela “employee” são confirmadas. As mensagens de erro dos dois blocos de exceção são inseridas na tabela de log.

```
CREATE TABLE employee (firstname varchar, lastname varchar);
INSERT INTO employee VALUES ('Tomas','Smith');
CREATE TABLE employee_error_log (message varchar);

CREATE OR REPLACE PROCEDURE update_employee_sp_3() NONATOMIC AS
$$
BEGIN
    BEGIN
        UPDATE employee SET firstname = 'Adam' WHERE lastname = 'Smith';
        EXECUTE 'select invalid1';
    EXCEPTION WHEN OTHERS THEN
        RAISE INFO 'An exception occurred in the first block.';
        INSERT INTO employee_error_log VALUES ('Error message: ' || SQLERRM);
    END;
    BEGIN
        INSERT INTO employee VALUES ('Edie','Robertson');
        EXECUTE 'select invalid2';
    EXCEPTION WHEN OTHERS THEN
        RAISE INFO 'An exception occurred in the second block.';
        INSERT INTO employee_error_log VALUES ('Error message: ' || SQLERRM);
    END;
END;
$$
LANGUAGE plpgsql;

CALL update_employee_sp_3();
INFO:  An exception occurred in the first block.
INFO:  An exception occurred in the second block.
CALL

SELECT * from employee;
```

```
 firstname | lastname
-----------+-----------
 Adam      | Smith
 Edie      | Robertson
(2 rows)

SELECT * from employee_error_log;

                     message
-----------------------------------------------
 Error message: column "invalid1" does not exist
 Error message: column "invalid2" does not exist
(2 rows)
```

O exemplo a seguir mostra como usar o manipulador de exceções CONTINUE. Esse exemplo cria duas tabelas e as usa em um procedimento armazenado. O manipulador CONTINUE controla o fluxo de execução em um procedimento armazenado com comportamento de gerenciamento de transações NONATOMIC.

```
CREATE TABLE tbl_1 (a int);
CREATE TABLE tbl_error_logging(info varchar, err_state varchar, err_msg varchar);

CREATE OR REPLACE PROCEDURE sp_exc_handling_1() NONATOMIC AS
$$
BEGIN
    INSERT INTO tbl_1 VALUES (1);
    -- Expect an error for the insert statement following, because of the invalid value
    INSERT INTO tbl_1 VALUES ("val");
    INSERT INTO tbl_1 VALUES (2);
EXCEPTION CONTINUE_HANDLER WHEN OTHERS THEN
    INSERT INTO tbl_error_logging VALUES ('Encountered error', SQLSTATE, SQLERRM);
END;
$$ LANGUAGE plpgsql;
```

Chame o procedimento armazenado:

```
CALL sp_exc_handling_1();
```

O fluxo avança assim:

1. Ocorre um erro porque é feita uma tentativa de inserir um tipo de dados incompatível em uma coluna. O controle passa para o bloco EXCEPTION. Quando o bloco de tratamento de exceções é acessado, a transação atual é revertida e uma nova transação implícita é criada para executar as instruções nela.
2. Se as instruções em CONTINUE_HANDLER forem executadas sem erros, o controle passará imediatamente para a instrução depois da instrução que estiver causando a exceção. (Se uma instrução em CONTINUE_HANDLER gerar uma nova exceção, você poderá tratá-la com um manipulador de exceções dentro do bloco EXCEPTION.)

Depois de você chamar o procedimento armazenado de amostra, as tabelas conterão os seguintes registros:

- Se você executar `SELECT * FROM tbl_1;`, ele retornará dois registros. Eles contêm os valores 1 e 2.
- Se você executar `SELECT * FROM tbl_error_logging;`, ele retornará um registro com estes valores: Erro encontrado, 42703 e A coluna "val" não existe em tbl_1.

O exemplo adicional do tratamento de erros a seguir usa um manipulador EXIT e um manipulador CONTINUE. Ele cria duas tabelas: uma tabela de dados e uma tabela de logs. Ele também cria um procedimento armazenado que demonstra o tratamento de erros:

```
CREATE TABLE tbl_1 (a int);
CREATE TABLE tbl_error_logging(info varchar, err_state varchar, err_msg varchar);

CREATE OR REPLACE PROCEDURE sp_exc_handling_2() NONATOMIC AS
$$
BEGIN
    INSERT INTO tbl_1 VALUES (1);
    BEGIN
        INSERT INTO tbl_1 VALUES (100);
        -- Expect an error for the insert statement following, because of the invalid
 value
        INSERT INTO tbl_1 VALUES ("val");
        INSERT INTO tbl_1 VALUES (101);
    EXCEPTION EXIT_HANDLER WHEN OTHERS THEN
        INSERT INTO tbl_error_logging VALUES ('Encountered error', SQLSTATE, SQLERRM);
    END;
    INSERT INTO tbl_1 VALUES (2);
    -- Expect an error for the insert statement following, because of the invalid value
```

```
    INSERT INTO tbl_1 VALUES ("val");
    INSERT INTO tbl_1 VALUES (3);
EXCEPTION CONTINUE_HANDLER WHEN OTHERS THEN
    INSERT INTO tbl_error_logging VALUES ('Encountered error', SQLSTATE, SQLERRM);
END;
$$ LANGUAGE plpgsql;
```

Depois de criar o procedimento armazenado, chame-o com o seguinte:

```
CALL sp_exc_handling_2();
```

Quando ocorre um erro no bloco de exceção interno, delimitado pelo conjunto interno de BEGIN e END, ele é tratado pelo manipulador EXIT. Qualquer erro ocorrido no bloco externo é tratado pelo manipulador CONTINUE.

Depois de você chamar o procedimento armazenado de amostra, as tabelas conterão os seguintes registros:

- Se você executar `SELECT * FROM tbl_1;`, ele retornará quatro registros, com os valores 1, 2, 3 e 100.
- Se você executar `SELECT * FROM tbl_error_logging;`, ele retornará dois registros. Eles têm estes valores: Erro encontrado, 42703 e A coluna "val" não existe em tbl_1.

Se a tabela tbl_error_logging não existir, ela vai gerar uma exceção.

O exemplo a seguir mostra como usar o manipulador de exceções CONTINUE com o loop FOR. Esse exemplo cria três tabelas e as usa em um loop FOR dentro de um procedimento armazenado. O loop FOR é uma variante do conjunto de resultados, o que significa que itera nos resultados de uma consulta:

```
CREATE TABLE tbl_1 (a int);
INSERT INTO tbl_1 VALUES (1), (2), (3);
CREATE TABLE tbl_2 (a int);
CREATE TABLE tbl_error_logging(info varchar, err_state varchar, err_msg varchar);

CREATE OR REPLACE PROCEDURE sp_exc_handling_loop() NONATOMIC AS
$$
DECLARE
 rec RECORD;
```

```
BEGIN
    FOR rec IN SELECT a FROM tbl_1
    LOOP
        IF rec.a = 2 THEN
            -- Expect an error for the insert statement following, because of the
 invalid value
            INSERT INTO tbl_2 VALUES("val");
        ELSE
            INSERT INTO tbl_2 VALUES (rec.a);
        END IF;
    END LOOP;
EXCEPTION CONTINUE_HANDLER WHEN OTHERS THEN
    INSERT INTO tbl_error_logging VALUES ('Encountered error', SQLSTATE, SQLERRM);
END;
$$ LANGUAGE plpgsql;
```

Chame o procedimento armazenado:

```
CALL sp_exc_handling_loop();
```

Depois de você chamar o procedimento armazenado de amostra, as tabelas conterão os seguintes registros:

- Se você executar `SELECT * FROM tbl_2;`, ele retornará dois registros. Eles contêm os valores 1 e 3.
- Se você executar `SELECT * FROM tbl_error_logging;`, ele retornará um registro com estes valores: Erro encontrado, 42703 e A coluna "val" não existe em tbl_2.

Observações sobre o uso referente ao manipulador CONTINUE:

- As palavras-chave CONTINUE_HANDLER e EXIT_HANDLER só podem ser usadas em procedimentos armazenados NONATOMIC.
- As palavras-chave CONTINUE_HANDLER e EXIT_HANDLER são opcionais. EXIT_HANDLER é o padrão.

Registro em log de procedimentos armazenados

Este tópico descreve os procedimentos armazenados e as visualizações que o Amazon Redshift usa para o registro em log de procedimentos armazenados.

Detalhes sobre procedimentos armazenados são registrados em log nas seguintes tabelas e visualizações do sistema:

- SVL_STORED_PROC_CALL - Os detalhes são registrados sobre a hora de início e de término da chamada de procedimento armazenado e se a chamada é encerrada antes da conclusão. Para obter mais informações, consulte SVL_STORED_PROC_CALL.
- SVL_STORED_PROC_MESSAGES - Mensagens em procedimentos armazenados emitidos pela consulta RAISE são registradas com o nível de registro em log correspondente. Para obter mais informações, consulte SVL_STORED_PROC_MESSAGES.
- SVL_QLOG - O ID da consulta da chamada de procedimento é registrado para cada consulta chamada a partir de um procedimento armazenado. Para obter mais informações, consulte SVL_QLOG.
- STL_UTILITYTEXT - Chamadas de procedimento armazenado são registradas em log após serem concluídas. Para obter mais informações, consulte STL_UTILITYTEXT.
- PG_PROC_INFO - Esta visualização do catálogo do sistema mostra informações sobre procedimentos armazenados. Para obter mais informações, consulte PG_PROC_INFO.

Limitações dos procedimentos armazenados

Este tópico descreve as limitações dos procedimentos armazenados do Amazon Redshift.

As considerações a seguir se aplicam ao usar procedimentos armazenados do Amazon Redshift.

Diferenças entre Amazon Redshift e PostgreSQL para suporte a procedimentos armazenados

A seguir estão as diferenças entre o suporte a procedimentos armazenados no Amazon Redshift e PostgreSQL:

- O Amazon Redshift não oferece suporte a subtransações e, portanto, tem suporte limitado para blocos de tratamento de exceção.

Considerações e limitações

Veja a seguir as considerações sobre os procedimentos armazenados no Amazon Redshift:

- O número máximo de procedimentos armazenados para um banco de dados é 10.000.

- O tamanho máximo do código-fonte para um procedimento é 2 MB.
- O número máximo de cursores explícitos e implícitos que podem ser abertos simultaneamente em uma sessão do usuário é 1. Loops FOR que iteram sobre o conjunto de resultados de uma instrução SQL abrem cursores implícitos. Não há suporte para cursores aninhados.
- Os cursores explícitos e implícitos têm as mesmas restrições no tamanho do conjunto de resultados que os cursores padrão do Amazon Redshift. Para obter mais informações, consulte Restrições de cursor.
- O número máximo de níveis para chamadas aninhadas é 16.
- O número máximo de parâmetros de procedimento é 32 para argumentos de entrada e 32 para argumentos de saída.
- O número máximo de variáveis em um procedimento armazenado é 1.024.
- Qualquer comando SQL que requer seu próprio contexto de transação não é compatível dentro de um procedimento armazenado. Os exemplos incluem:
  - PREPARE
  - CREATE/DROP DATABASE
  - CREATE EXTERNAL TABLE
  - VACUUM
  - SET LOCAL
  - ALTER TABLE APPEND
- A chamada do método `registerOutParameter` pelo driver do Java Database Connectivity (JDBC) não é compatível com o tipo de dados `refcursor`. Para obter um exemplo de uso do tipo de dados `refcursor`, consulte Exibir um conjunto de resultados de um procedimento armazenado.

Referência da linguagem PL/pgSQL

Os procedimentos armazenados no Amazon Redshift são baseados na linguagem procedural PostgreSQL PL/pgSQL, com algumas diferenças importantes. Nesta referência, você pode encontrar detalhes da sintaxe PL/pgSQL implementada pelo Amazon Redshift. Para obter mais informações sobre PL/pgSQL, consulte PL pgSQL - Linguagem procedural SQL na documentação do PostgreSQL.

Tópicos

- Convenções de referência da PL/pgSQL

- Estrutura da PL/pgSQL
- Instruções da PL/pgSQL compatíveis

Convenções de referência da PL/pgSQL

Nesta seção, encontre as convenções que são usadas para escrever a sintaxe para a linguagem de procedimento armazenado PL/pgSQL.

Caractere	Descrição
CAPS	Palavras em letras maiúsculas são palavras-chave.
[ ]	Parênteses denotam argumentos opcionais. Vários argumentos em parênteses indicam que você pode escolher qualquer número de argumento s. Além disso, os argumentos entre colchetes em linhas separadas indicam que o analisador Amazon Redshift espera que os argumentos estejam na ordem em que estão listados na sintaxe.
{ }	As chaves indicam que será necessário escolher um dos argumentos nelas.
|	Barras verticais indicam que você pode escolher entre os argumentos.
vermelho itálico	As palavras em vermelho itálico indicam espaços reservados. Insira o valor apropriado no lugar da palavra em vermelho itálico.
. . .	Uma elipse indica que você pode repetir o elemento anterior.
'	Palavras entre aspas simples indicam que você deve digitar as aspas.

Estrutura da PL/pgSQL

PL/pgSQL é uma linguagem procedural com muitas das mesmas construções de outras linguagens procedurais.

Tópicos

- Bloquear
- Declaração de variáveis

- Declaração de alias
- Variáveis integradas
- Tipos de registro

Bloquear

PL/pgSQL é uma linguagem estruturada em blocos. O corpo completo de um procedimento é definido em um bloco, que contém declarações variáveis e instruções em PL/pgSQL. Uma instrução também pode ser um bloco aninhado, ou um sub-bloco.

Finalize as declarações e as instruções com ponto-e-vírgula. Coloque um ponto-e-vírgula após a palavra-chave END de um bloco ou sub-bloco. Não use ponto-e-vírgula depois das palavras DECLARE e BEGIN.

Todas as palavras-chave e os identificadores podem ser escritos com mistura entre maiúsculas e minúsculas. Os identificadores são convertidos implicitamente para minúsculas a menos que estejam entre aspas.

Um hífen duplo (--) inicia um comentário que se estende até o final da linha. Um /* inicia um comentário de bloco que se estende até a próxima ocorrência de */. Não é possível aninhar comentários de bloco. No entanto, é possível incluir comentários de hífen duplo em um comentário de bloco e um hífen duplo pode ocultar os delimitadores /* e */ do comentário de bloco.

Qualquer instrução na seção de instruções de um bloco pode ser um sub-bloco. Use sub-blocos para agrupamento lógico ou para localizar variáveis em um pequeno grupo de instruções.

```
[ <<label>> ]
[ DECLARE
  declarations ]
BEGIN
  statements
END [ label ];
```

As variáveis declaradas na seção de declarações que antecede um bloco são inicializadas com seus valores padrão toda vez que o bloco for inserido. Em outras palavras, não são inicializadas apenas uma vez por chamada da função.

Por exemplo:

```
CREATE PROCEDURE update_value() AS $$
DECLARE
  value integer := 20;
BEGIN
  RAISE NOTICE 'Value here is %', value;  -- Value here is 20
  value := 50;
  --
  -- Create a subblock
  --
  DECLARE
    value integer := 80;
  BEGIN
    RAISE NOTICE 'Value here is %', value;  -- Value here is 80
  END;

  RAISE NOTICE 'Value here is %', value;  -- Value here is 50
END;
$$ LANGUAGE plpgsql;
```

Use um rótulo para identificar o bloco a ser usado em uma instrução EXIT ou para qualificar os nomes das variáveis declaradas no bloco.

Não confunda o uso de BEGIN/END para instruções de agrupamento na PL/pgSQL com os comandos de banco de dados para controle de transações. BEGIN e END na PL/pgSQL destinam-se apenas para o agrupamento. Elas não iniciam ou encerram uma transação.

Declaração de variáveis

Declare todas as variáveis em um bloco, exceto as variáveis de loop, na seção DECLARE do bloco. As variáveis podem usar qualquer tipo de dados válido do Amazon Redshift. Para obter os tipos de dados compatíveis, consulte Tipos de dados.

As variáveis PL/pgSQL podem ser qualquer tipo de dados compatível com o Amazon Redshift, mais `RECORD` e `refcursor`. Para obter mais informações sobre o `RECORD`, consulte Tipos de registro. Para obter mais informações sobre o `refcursor`, consulte Cursores.

```
DECLARE
name [ CONSTANT ] type [ NOT NULL ] [ { DEFAULT | := } expression ];
```

Veja a seguir exemplos de declarações de variáveis.

```
customerID integer;
numberofitems numeric(6);
link varchar;
onerow RECORD;
```

A variável de loop de um loop FOR iterando sobre uma gama de inteiros é automaticamente declarada como uma variável inteira.

A cláusula DEFAULT, se fornecida, especifica o valor inicial atribuído à variável quando o bloco é inserido. Se a cláusula DEFAULT não for fornecida, a variável será inicializada com o valor SQL NULL. A opção CONSTANT impede que a variável seja atribuída, para que seu valor permaneça constante ao longo da duração do bloco. Se NOT NULL for especificado, uma atribuição de um valor nulo resultará em um erro de tempo de execução. Todas as variáveis declaradas como NOT NULL devem ter um valor padrão não nulo especificado.

O valor padrão é avaliado toda vez que o bloco for inserido. Por exemplo, atribuir `now()` a uma variável do tipo `timestamp` faz com que a variável tenha o horário da chamada de função atual, não o horário em que a função foi pré-compilada.

```
quantity INTEGER DEFAULT 32;
url VARCHAR := 'http://mysite.com';
user_id CONSTANT INTEGER := 10;
```

`refcursor` é o tipo de dados das variáveis de cursor dentro dos procedimento armazenados. Um valor `refcursor` pode ser retornado dentro de um procedimento armazenado. Para ter mais informações, consulte Exibir um conjunto de resultados de um procedimento armazenado.

Declaração de alias

Se a assinatura do procedimento armazenado omitir o nome do argumento, você pode declarar um alias para o argumento.

```
name ALIAS FOR $n;
```

Variáveis integradas

As seguintes variáveis integradas são compatíveis:

- FOUND

- SQLSTATE
- SQLERRM
- GET DIAGNOSTICS integer_var := ROW_COUNT;

FOUND é uma variável especial do tipo booliana. FOUND começa como false em cada chamada de procedimento. FOUND é definida pelos seguintes tipos de instruções:

- SELECT INTO

  Define FOUND como true se retornar uma linha e false se não retornar linhas.
- UPDATE, INSERT e DELETE

  Define FOUND como true se pelo menos uma linha for afetada e como false se nenhuma linha for afetada.
- FETCH

  Define FOUND como true se retornar uma linha e false se não retornar linhas.
- Instrução FOR

  Define FOUND como true se a instrução FOR iterar uma ou mais vezes, caso contrário, false. Isso se aplica às três variantes da instrução FOR: loops FOR de inteiros, loops FOR de conjunto de registros e loops FOR de conjunto de registros dinâmico.

  FOUND é definida ao sair do loop FOR. Dentro do runtime do loop, FOUND não é modificada pela instrução FOR. No entanto, ela pode ser alterada pela execução de outras instruções no corpo do loop.

Por exemplo:

```
CREATE TABLE employee(empname varchar);
CREATE OR REPLACE PROCEDURE show_found()
AS  $$
DECLARE
  myrec record;
BEGIN
  SELECT INTO myrec * FROM employee WHERE empname = 'John';
  IF NOT FOUND THEN
    RAISE EXCEPTION 'employee John not found';
  END IF;
```

```
END;
$$ LANGUAGE plpgsql;
```

Dentro de um tratador de exceções, a variável especial SQLSTATE contém o código de erro que corresponde à exceção gerada. A variável especial SQLERRM contém a mensagem de erro associada à exceção. Essas variáveis não são definidas fora dos tratadores de exceções e exibem um erro se forem usadas.

Por exemplo:

```
CREATE OR REPLACE PROCEDURE sqlstate_sqlerrm() AS
$$
BEGIN
  UPDATE employee SET firstname = 'Adam' WHERE lastname = 'Smith';
  EXECUTE 'select invalid';
  EXCEPTION WHEN OTHERS THEN
  RAISE INFO 'error message SQLERRM %', SQLERRM;
  RAISE INFO 'error message SQLSTATE %', SQLSTATE;
END;
$$ LANGUAGE plpgsql;
```

ROW_COUNT é usada com o comando GET DIAGNOSTICS. Ela mostra o número de linhas processadas pelo último comando SQL enviado para o mecanismo SQL.

Por exemplo:

```
CREATE OR REPLACE PROCEDURE sp_row_count() AS
$$
DECLARE
  integer_var int;
BEGIN
  INSERT INTO tbl_row_count VALUES(1);
  GET DIAGNOSTICS integer_var := ROW_COUNT;
  RAISE INFO 'rows inserted = %', integer_var;
END;
$$ LANGUAGE plpgsql;
```

Tipos de registro

Um tipo RECORD não é um tipo de dados real, apenas um espaço reservado. Variáveis de tipo de registro assumem a estrutura real da linha à qual são atribuídas durante o comando SELECT ou

FOR. A subestrutura de uma variável de registro pode mudar toda vez que receber um valor. Até uma variável de registro receber uma atribuição, ela não possui subestrutura. Qualquer tentativa de acessar um campo nela gerará um erro de tempo de execução.

```
name RECORD;
```

Por exemplo:

```
CREATE TABLE tbl_record(a int, b int);
INSERT INTO tbl_record VALUES(1, 2);
CREATE OR REPLACE PROCEDURE record_example()
LANGUAGE plpgsql
AS $$
DECLARE
  rec RECORD;
BEGIN
  FOR rec IN SELECT a FROM tbl_record
  LOOP
    RAISE INFO 'a = %', rec.a;
  END LOOP;
END;
$$;
```

Instruções da PL/pgSQL compatíveis

As instruções da PL/pgSQL aumentam comandos SQL com construções processuais, incluindo loops e expressões condicionais, a fim de controlar o fluxo lógico. A maioria dos comandos SQL pode ser usada, incluindo linguagem de manipulação de dados (DML), como COPY, UNLOAD e INSERT, e linguagem de definição de dados (DDL), como CREATE TABLE. Para obter a lista dos comandos SQL abrangentes, consulte Comandos SQL. Além disso, as seguintes instruções PL/pgSQL são compatíveis com o Amazon Redshift.

Tópicos

- Atribuição
- SELECT INTO
- No-op
- SQL dinâmico
- Return

- Condicionais: IF
- Condicionais: CASE
- Loops
- Cursores
- RAISE
- Controle da transação

Atribuição

A instrução de atribuição atribui um valor a uma variável. A expressão deve retornar um único valor.

```
identifier := expression;
```

Usar o = não padrão para a atribuição, em vez do `:=`, também é aceito.

Se o tipo de dados da expressão não corresponder ao tipo de dados da variável ou a variável tiver um tamanho ou uma precisão, o valor do resultado será convertido implicitamente.

Veja a seguir alguns exemplos.

```
customer_number := 20;
tip := subtotal * 0.15;
```

SELECT INTO

A instrução SELECT INTO atribui o resultado de várias colunas (mas somente uma linha) a uma variável de registro ou uma lista de variáveis escalares.

```
SELECT INTO target select_expressions FROM ...;
```

Na sintaxe anterior, *target* pode ser uma variável de registro ou uma lista separada por vírgulas de variáveis simples e campos de registro. A lista *select_expressions* e o restante do comando são iguais ao SQL regular.

Se uma lista de variáveis for usada como *target*, os valores selecionados deverão corresponder exatamente à estrutura do destino, caso contrário, ocorrerá um erro de tempo de execução. Quando

uma variável de registro for o destino, ela se configura automaticamente para o tipo de linha das colunas do resultado da consulta.

A cláusula INTO pode aparecer praticamente em qualquer lugar na instrução SELECT. Normalmente ela aparece logo após a cláusula SELECT ou imediatamente antes da cláusula FROM. Ou seja, ela aparece imediatamente antes ou logo após a lista *select_expressions*.

Se a consulta não retornar nenhuma linha, valores NULL serão atribuídos a *target*. Se a consulta retornar várias linhas, a primeira linha será atribuída a *target* e o restante é descartado. A menos que a instrução contenha um ORDER BY, a primeira linha é não determinística.

Para determinar se a atribuição retornou pelo menos uma fila, use a variável especial FOUND.

```
SELECT INTO customer_rec * FROM cust WHERE custname = lname;
IF NOT FOUND THEN
  RAISE EXCEPTION 'employee % not found', lname;
END IF;
```

Para testar se um resultado de registro é nulo, use o condicional IS NULL. Não é possível determinar se outras linhas adicionais foram descartadas. O exemplo a seguir lida com o caso em que nenhuma linha foi retornada.

```
CREATE OR REPLACE PROCEDURE select_into_null(return_webpage OUT varchar(256))
AS $$
DECLARE
  customer_rec RECORD;
BEGIN
  SELECT INTO customer_rec * FROM users WHERE user_id=3;
  IF customer_rec.webpage IS NULL THEN
    -- user entered no webpage, return "http://"
    return_webpage = 'http://';
  END IF;
END;
$$ LANGUAGE plpgsql;
```

No-op

A instrução no-op (`NULL;`) é uma instrução de espaço reservado que não faz nada. Uma instrução no-op pode indicar que uma ramificação de uma cadeia IF-THEN-ELSE está vazia.

```
NULL;
```

SQL dinâmico

Para gerar comandos dinâmicos que podem involver diferentes tabelas ou diferentes tipos de dados toda vez que forem executados a partir de um procedimento armazenado em PL/pgSQL, use a instrução EXECUTE.

```
EXECUTE command-string [ INTO target ];
```

Acima, *command-string* é uma expressão que gera uma string (do tipo texto) que contém o comando a ser executado. Esse valor de *command-string* é enviado ao mecanismo SQL. Nenhuma substituição das variáveis em PL/pgSQL é feita na string de comando. Os valores das variáveis deve ser inserido na string de comando à medida que é construída.

> **Note**
>
> Não é possível usar as instruções COMMIT e ROLLBACK em SQL dinâmico. Para obter informações sobre como usar as instruções COMMIT e ROLLBACK em um procedimento armazenado, consulte Gerenciamento de transações.

Ao trabalhar com comandos dinâmicos, é usual precisar lidar com o escape de aspas simples. Recomendamos colocar texto fixo entre aspas no corpo da função usando cotação de dólar. Valores dinâmicos a serem inseridos em uma consulta construída exigem tratamento especial, pois eles mesmos podem conter aspas. O exemplo a seguir assume cotação de dólar para a função como um todo, para que as aspas não precisem ser duplicadas.

```
EXECUTE 'UPDATE tbl SET '
  || quote_ident(colname)
  || ' = '
  || quote_literal(newvalue)
  || ' WHERE key = '
  || quote_literal(keyvalue);
```

O exemplo anterior mostra as funções `quote_ident(text)` e `quote_literal(text)`. Esse exemplo envia variáveis que contém identificadores de coluna e tabela para a função `quote_ident`. Ele também envia variáveis que contém strings literais no comando construído para a função `quote_literal`. Ambas as funções executam as etapas apropriadas para retornar o texto de entrada entre aspas duplas ou simples, respectivamente, com quaisquer caracteres especiais incorporados com escape adequado.

A cotação de dólar só é útil para a cotação de texto fixo. Não grave o exemplo anterior no formato a seguir.

```
EXECUTE 'UPDATE tbl SET '
  || quote_ident(colname)
  || ' = $$'
  || newvalue
  || '$$ WHERE key = '
  || quote_literal(keyvalue);
```

Não faça isso pois o exemplo trava se o conteúdo de `newvalue` conter $$. O mesmo problema se aplica a qualquer outro delimitador de cotação de dólar que venha a escolher. Para citar com segurança um texto não conhecido previamente, use a função `quote_literal`.

Return

A instrução RETURN retorna para o chamador a partir de um procedimento armazenado.

```
RETURN;
```

Por exemplo:

```
CREATE OR REPLACE PROCEDURE return_example(a int)
AS $$
BEGIN
  FOR b in 1..10 LOOP
    IF b < a THEN
      RAISE INFO 'b = %', b;
    ELSE
      RETURN;
    END IF;
  END LOOP;
END;
$$ LANGUAGE plpgsql;
```

Condicionais: IF

A instrução condicional IF pode assumir as seguintes formas na linguagem PL/pgSQL que o Amazon Redshift usa:

- IF ... THEN

```
IF boolean-expression THEN
  statements
END IF;
```

Por exemplo:

```
IF v_user_id <> 0 THEN
  UPDATE users SET email = v_email WHERE user_id = v_user_id;
END IF;
```

- IF ... THEN ... ELSE

```
IF boolean-expression THEN
  statements
ELSE
  statements
END IF;
```

Por exemplo:

```
IF parentid IS NULL OR parentid = ''
THEN
  return_name = fullname;
  RETURN;
ELSE
  return_name = hp_true_filename(parentid) || '/' || fullname;
  RETURN;
END IF;
```

- IF ... THEN ... ELSIF ... THEN ... ELSE

A palavra-chave ELSIF também pode ser escrita como ELSEIF.

```
IF boolean-expression THEN
  statements
[ ELSIF boolean-expression THEN
  statements
[ ELSIF boolean-expression THEN
  statements
    ...] ]
[ ELSE
```

```
  statements ]
END IF;
```

Por exemplo:

```
IF number = 0 THEN
  result := 'zero';
ELSIF number > 0 THEN
  result := 'positive';
ELSIF number < 0 THEN
  result := 'negative';
ELSE
  -- the only other possibility is that number is null
  result := 'NULL';
END IF;
```

Condicionais: CASE

A instrução condicional CASE pode assumir as seguintes formas na linguagem PL/pgSQL que o Amazon Redshift usa:

- CASE simples

```
CASE search-expression
WHEN expression [, expression [ ... ]] THEN
  statements
[ WHEN expression [, expression [ ... ]] THEN
  statements
  ... ]
[ ELSE
  statements ]
END CASE;
```

Uma instrução CASE simples oferece execução condicional com base na igualdade dos operandos.

O valor *search-expression* é avaliado uma vez e comparado sucessivamente com cada *expression* nas cláusulas WHEN. Se uma correspondência for encontrada, as *statements* correspondentes são executadas e o controle passa para a próxima instrução após END CASE. As expressões WHEN subsequentes não são avaliadas. Se nenhuma correspondência for

encontrada, as *statements* ELSE são executadas. No entanto, se ELSE não estiver presente, uma exceção CASE_NOT_FOUND será gerada.

Por exemplo:

```
CASE x
WHEN 1, 2 THEN
  msg := 'one or two';
ELSE
  msg := 'other value than one or two';
END CASE;
```

- CASE pesquisado

```
CASE
WHEN boolean-expression THEN
  statements
[ WHEN boolean-expression THEN
  statements
  ... ]
[ ELSE
  statements ]
END CASE;
```

A formato pesquisado de CASE oferece execução condicional baseada na veracidade de expressões boolianas.

A *boolean-expression* de cada cláusula WHEN é, por sua vez, avaliada até uma verdadeira ser encontrada. Depois, as instruções correspondentes são executadas e o controle passa para a próxima instrução após END CASE. As *expressions* WHEN subsequentes não são avaliadas. Se nenhum resultado verdadeiro for encontrado, as *statements* ELSE são executadas. No entanto, se ELSE não estiver presente, uma exceção CASE_NOT_FOUND será gerada.

Por exemplo:

```
CASE
WHEN x BETWEEN 0 AND 10 THEN
  msg := 'value is between zero and ten';
WHEN x BETWEEN 11 AND 20 THEN
  msg := 'value is between eleven and twenty';
END CASE;
```

Loops

As instruções de loop podem assumir os seguintes formatos na linguagem PL/pgSQL que o Amazon Redshift usa:

- Loop simples

```
[<<label>>]
LOOP
  statements
END LOOP [ label ];
```

Um loop simples define um loop não condicional que é repetido indefinidamente até ser encerrado por uma instrução EXIT ou RETURN. O rótulo opcional pode ser usado pelas instruções EXIT e CONTINUE dentro de loops aninhados para especificar a qual loop as instruções EXIT e CONTINUE se referem.

Por exemplo:

```
CREATE OR REPLACE PROCEDURE simple_loop()
LANGUAGE plpgsql
AS $$
BEGIN
  <<simple_while>>
  LOOP
    RAISE INFO 'I am raised once';
    EXIT simple_while;
    RAISE INFO 'I am not raised';
  END LOOP;
  RAISE INFO 'I am raised once as well';
END;
$$;
```

- Loop de saída

```
EXIT [ label ] [ WHEN expression ];
```

Se *label* não estiver presente, o loop mais interno será encerrado e a instrução após END LOOP será executada na sequência. Se *label* estiver presente, deverá ser o rótulo do loop atual ou de

algum outro nível de loop ou bloco aninhado. Depois, o loop ou bloco nomeado será encerrado e o controle continuará com a instrução após END do loop ou bloco correspondente.

Se WHEN for especificada, a saída do loop ocorrerá apenas se a *expression* for verdadeira. Caso contrário, o controle passa para a instrução após EXIT.

Você pode usar EXIT com todos os tipos de loops, seu uso não se limita a loops não condicionais.

Quando usado com um bloco BEGIN, EXIT passa o controle para a próxima instrução após o final do bloco. Um rótulo pode ser usado para essa finalidade. Um EXIT sem rótulo nunca corresponderá a um bloco BEGIN.

Por exemplo:

```
CREATE OR REPLACE PROCEDURE simple_loop_when(x int)
LANGUAGE plpgsql
AS $$
DECLARE i INTEGER := 0;
BEGIN
  <<simple_loop_when>>
  LOOP
    RAISE INFO 'i %', i;
    i := i + 1;
    EXIT simple_loop_when WHEN (i >= x);
  END LOOP;
END;
$$;
```

- Loop de continuação

```
CONTINUE [ label ] [ WHEN expression ];
```

Se *label* não for fornecido, a execução pulará para a próxima iteração do loop mais interno. Ou seja, todas as instruções restantes no corpo do loop serão ignoradas. Depois, o controle retornará para a expressão de controle do loop (se houver) a fim de determinar se há necessidade de outra iteração do loop. Se *label* estiver presente, ele especifica o rótulo do loop cuja execução será continuada.

Se WHEN for especificado, a próxima iteração do loop começará somente se a *expression* for verdadeira. Caso contrário, o controle passa para a instrução após CONTINUE.

Você pode usar CONTINUE com todos os tipos de loops, seu uso não se limita a loops não condicionais.

```
CONTINUE mylabel;
```

- Loop WHILE

```
[<<label>>]
WHILE expression LOOP
  statements
END LOOP [ label ];
```

A instrução WHILE repete uma sequência de instruções até a *boolean-expression* ser avaliada como verdadeira. A expressão é verificada imediatamente antes de cada entrada no corpo do loop.

Por exemplo:

```
WHILE amount_owed > 0 AND gift_certificate_balance > 0 LOOP
  -- some computations here
END LOOP;

WHILE NOT done LOOP
  -- some computations here
END LOOP;
```

- Loop FOR (variante inteiro)

```
[<<label>>]
FOR name IN [ REVERSE ] expression .. expression LOOP
  statements
END LOOP [ label ];
```

O loop FOR (variante inteiro) cria um loop que itera sobre um intervalo de valores inteiros. O nome da variável é definido automaticamente como o tipo inteiro e exite somente dentro do loop. Qualquer definição existente do nome da variável será ignorada dentro do loop. As duas expressões que fornecem os limites superior e inferior do intervalo são avaliadas uma vez ao entrar no loop. Se você especificar REVERSE, o valor da etapa será subtraído, em vez de adicionado, após cada iteração.

Se o limite inferior for maior que o limite superior (ou menor no caso REVERSE), o corpo do loop não será executado. Nenhum erro será gerado.

Se um rótulo for anexado ao loop FOR, você pode referenciar a variável de loop inteira com um nome qualificado, usando esse rótulo.

Por exemplo:

```
FOR i IN 1..10 LOOP
  -- i will take on the values 1,2,3,4,5,6,7,8,9,10 within the loop
END LOOP;

FOR i IN REVERSE 10..1 LOOP
  -- i will take on the values 10,9,8,7,6,5,4,3,2,1 within the loop
END LOOP;
```

- Loop FOR (variante conjunto de resultados)

```
[<<label>>]
FOR target IN query LOOP
  statements
END LOOP [ label ];
```

O *target* é uma variável de registro ou lista separada por vírgulas de variáveis escalares. O destino recebe todas as linhas resultantes da consulta e o corpo do loop é executado para cada linha.

O loop FOR (variante conjunto de resultados) habilita a iteração de um procedimento armazenado pelos resultados de uma consulta e manipula os dados de acordo.

Por exemplo:

```
CREATE PROCEDURE cs_refresh_reports() AS $$
DECLARE
  reports RECORD;
BEGIN
  FOR reports IN SELECT * FROM cs_reports ORDER BY sort_key LOOP
    -- Now "reports" has one record from cs_reports
    EXECUTE 'INSERT INTO ' || quote_ident(reports.report_name) || ' ' ||
 reports.report_query;
  END LOOP;
```

```
  RETURN;
END;
$$ LANGUAGE plpgsql;
```

- Loop FOR com SQL dinâmico

```
[<<label>>]
FOR record_or_row IN EXECUTE text_expression LOOP
  statements
END LOOP;
```

Um loop FOR com SQL dinâmico habilita a iteração de um procedimento armazenado pelos resultados de uma consulta dinâmica e manipula os dados de acordo.

Por exemplo:

```
CREATE OR REPLACE PROCEDURE for_loop_dynamic_sql(x int)
LANGUAGE plpgsql
AS $$
DECLARE
  rec RECORD;
  query text;
BEGIN
  query := 'SELECT * FROM tbl_dynamic_sql LIMIT ' || x;
  FOR rec IN EXECUTE query
  LOOP
    RAISE INFO 'a %', rec.a;
  END LOOP;
END;
$$;
```

Cursores

Em vez de executar uma consulta completa de uma só vez, é possível definir um cursor. Um cursor encapsula uma consulta e lê o resultado da consulta algumas linhas por vez. Um dos motivos para fazer isso é evitar o estouro de memória quando o resultado contiver um grande número de linhas. Outro motivo é retornar a referência a um cursor criado por um procedimento armazenado, o que permite ao chamador ler as linhas. Essa abordagem oferece uma maneira eficiente de retornar grandes conjuntos de linhas de procedimentos armazenados.

Para usar cursores em um procedimento armazenado NONATOMIC, coloque o loop do cursor entre START TRANSACTION...COMMIT.

Para configurar um cursor, primeiro declare uma variável de cursor. Todo o acesso a cursores na PL/pgSQL passa por variáveis de cursor, que sempre são do tipo de dados especial `refcursor`. Um tipo de dados `refcursor` apenas mantém uma referência a um cursor.

Você pode criar uma variável de cursor declarando-a como uma variável do tipo `refcursor`. Ou, use a sintaxe de declaração de cursor a seguir.

```
name CURSOR [ ( arguments ) ] FOR query ;
```

No trecho acima, *`arguments`* (se especificado) é uma lista separada por vírgulas de pares *`name datatype`* em que cada um define nomes a serem substituídos por valores de parâmetros em *`query`*. Os valores reais que substituirão esses nomes serão especificados posteriormente, quando o cursor for aberto.

Veja a seguir alguns exemplos.

```
DECLARE
  curs1 refcursor;
  curs2 CURSOR FOR SELECT * FROM tenk1;
  curs3 CURSOR (key integer) IS SELECT * FROM tenk1 WHERE unique1 = key;
```

Essas três variáveis têm o tipo de dados `refcursor`, mas a primeira pode ser usada com qualquer consulta. Por outro lado, a segunda tem uma consulta totalmente especificada já associada a ela, e a última tem uma consulta parametrizada associada a ela. O valor `key` é substituído por um valor de parâmetro inteiro quando o cursor é aberto. Diz-se que a variável `curs1` é não associada pois ela não está vinculada a nenhuma consulta específica.

Antes de usar um cursor para recuperar linhas, ele deve ser aberto. PL/pgSQL possui três formas da instrução OPEN, das quais duas usam variáveis de cursor não associadas e a terceira usa uma variável de cursor associada:

- Abrir para seleção: a variável de cursor é aberta e recebe a consulta especificada para execução. O cursor não pode já estar aberto. Além disso, ele deverá ter sido declarado como um cursor não associado (isto é, como uma variável `refcursor` simples). A consulta SELECT é tratada da mesma maneira que outras instruções SELECT na PL/pgSQL.

```
OPEN cursor_name FOR SELECT ...;
```

Por exemplo:

```
OPEN curs1 FOR SELECT * FROM foo WHERE key = mykey;
```

- Abrir para execução: a variável de cursor é aberta e recebe a consulta especificada para execução. O cursor não pode já estar aberto. Além disso, ele deverá ter sido declarado como um cursor não associado (isto é, como uma variável `refcursor` simples). A consulta é especificada como uma expressão de string da mesma maneira que no comando EXECUTE. Essa abordagem oferece flexibilidade para que a consulta possa variar de uma execução para a próxima.

  ```
  OPEN cursor_name FOR EXECUTE query_string;
  ```

  Por exemplo:

  ```
  OPEN curs1 FOR EXECUTE 'SELECT * FROM ' || quote_ident($1);
  ```

- Abrir um cursor associado: essa forma de OPEN é usada para abrir uma variável de cursor cuja consulta foi associada a ela no momento em que foi declarada. O cursor não pode já estar aberto. Uma lista das expressões de valor de argumento reais deverá aparecer se, e somente se, o cursor tiver sido declarado para receber argumentos. Esses valores são substituídos na consulta.

  ```
  OPEN bound_cursor_name [ ( argument_values ) ];
  ```

  Por exemplo:

  ```
  OPEN curs2;
  OPEN curs3(42);
  ```

Depois que um cursor tiver sido aberto, trabalhe com ele usando as instruções descritas a seguir. Essas instruções não precisam ocorrer no mesmo procedimento armazenado que abriu o cursor. Você pode retornar um valor `refcursor` fora de um procedimento armazenado e deixar o chamador operar no cursor. Todos os portais são implicitamente fechados ao final da transação. Portanto, você pode usar um valor `refcursor` para fazer referência a um cursor aberto somente até o final da transação.

- FETCH recupera a próxima linha do cursor em um destino. Esse destino pode ser uma variável de linha, uma variável de registro ou uma lista separada por vírgulas de variáveis simples, igual à SELECT INTO. Como na SELECT INTO, é possível verificar a variável especial FOUND para conferir se uma linha foi obtida.

  ```
  FETCH cursor INTO target;
  ```

  Por exemplo:

  ```
  FETCH curs1 INTO rowvar;
  ```

- CLOSE fecha o portal subjacente a um cursor aberto. Use essa instrução para liberar recursos antes do fim da transação. Você também pode usar essa instrução a fim de liberar a variável de cursor para que possa ser aberta novamente.

  ```
  CLOSE cursor;
  ```

  Por exemplo:

  ```
  CLOSE curs1;
  ```

RAISE

Use a instrução `RAISE level` para relatar mensagens e gerar erros.

```
RAISE level 'format' [, variable [, ...]];
```

Os níveis possíveis são NOTICE, INFO, LOG, WARNING e EXCEPTION. EXCEPTION gera um erro, que normalmente cancela a transação atual. Os outros níveis geram apenas mensagens com diferentes níveis de prioridade.

Dentro da string de formato, % é substituído pela representação de string do próximo argumento opcional. Escreva %% para emitir um % literal. No momento, argumentos opcionais devem ser variáveis simples, não expressões, e o formato deve ser uma string literal simples.

No exemplo a seguir, o valor de `v_job_id` substitui o % na string.

```
RAISE NOTICE 'Calling cs_create_job(%)', v_job_id;
```

Use a instrução RAISE para relançar a exceção capturada por um bloco de tratamento de exceções. Essa instrução só é válida em blocos de tratamento de exceções de procedimentos armazenados no modo NONATOMIC.

```
RAISE;
```

Controle da transação

Você pode trabalhar com instruções de controle de transação na linguagem PL/pgSQL que o Amazon Redshift usa. Para obter informações sobre como usar as instruções COMMIT, ROLLBACK e TRUNCATE em um procedimento armazenado, consulte Gerenciamento de transações.

Nos procedimentos armazenados no modo NONATOMIC, use START TRANSACTION para iniciar um bloco de transação.

```
START TRANSACTION;
```

> **Note**
>
> A instrução PL/pgSQL START TRANSACTION é diferente do comando SQL START TRANSACTION das seguintes formas:
>
> - Nos procedimentos armazenados, START TRANSACTION não é sinônimo de BEGIN.
> - A instrução PL/pgSQL não é compatível com palavras-chave opcionais de nível de isolamento e permissão de acesso.

Visões materializadas no Amazon Redshift

Esta seção descreve como criar e utilizar visões materializadas no Amazon Redshift. Visão materializada é um objeto de banco de dados que armazena os resultados de uma consulta, os quais podem ser usados para melhorar a performance e a eficiência.

Em um ambiente de data warehouse, as aplicações, com frequência, precisam executar consultas complexas em tabelas grandes. Um exemplo são as instruções SELECT que executam junções de várias tabelas e agregações em tabelas que contêm bilhões de linhas. O processamento dessas consultas pode ser dispendioso em termos de recursos do sistema e do tempo necessário para calcular os resultados.

As visões materializadas no Amazon Redshift fornecem uma maneira de resolver esses problemas. Uma visão materializada contém um conjunto de resultados pré-computados, com base em uma consulta SQL a uma ou mais tabelas base. É possível emitir instruções SELECT para consultar uma visão materializada, da mesma maneira como você pode consultar outras tabelas ou visualizações no banco de dados. O Amazon Redshift retorna os resultados pré-computados da visão materializada, sem necessidade de acessar as tabelas base. Do ponto de vista do usuário, os resultados da consulta são retornados muito mais rapidamente em comparação a quando os mesmos dados são recuperados das tabelas base.

As visões materializadas são especialmente úteis para acelerar consultas que são previsíveis e repetidas. Em vez de realizar consultas com uso intensivo de recursos em grandes tabelas (como agregados ou junções múltiplas), as aplicações podem consultar uma visão materializada e recuperar um conjunto de resultados pré-computado. Por exemplo, considere o cenário em que um conjunto de consultas é usado para preencher painéis, como o Amazon Quick. Esse caso de uso é ideal para uma visão materializada, porque as consultas são previsíveis e repetidas várias vezes.

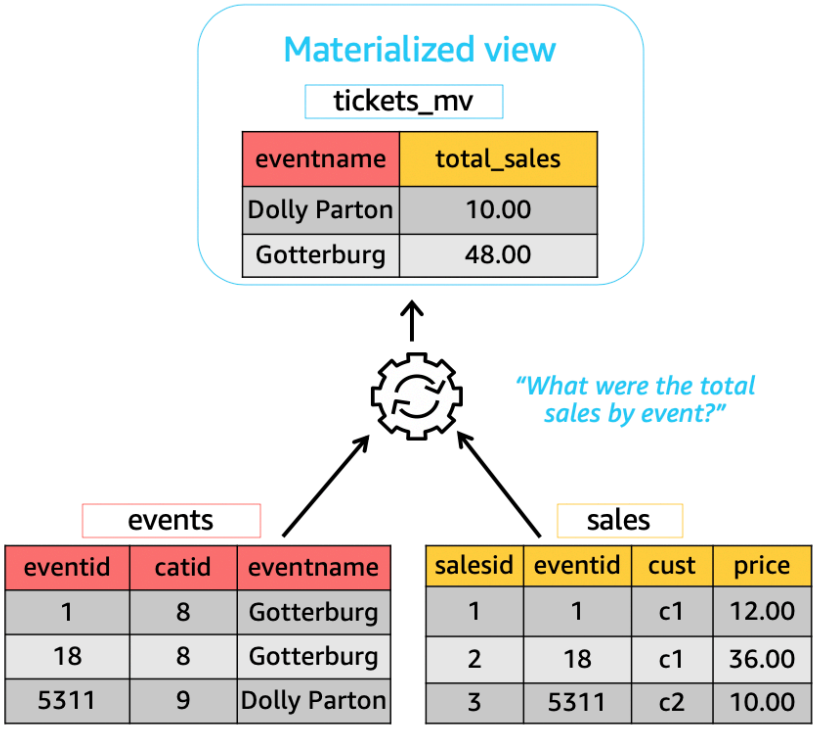
Você pode definir uma visão materializada em termos de outras visões materializadas. Use visões materializadas em visões materializadas para expandir a capacidade de visões materializadas. Nessa abordagem, uma visão materializada existente desempenha a mesma função que uma tabela base para a consulta recuperar dados.

Essa abordagem é especialmente útil para reutilizar junções pré-calculadas para diferentes opções agregadas ou GROUP BY. Por exemplo, tome uma visão materializada que une informações do cliente (contendo milhões de linhas) com informações detalhadas da ordem do item (contendo bilhões de linhas). Esta é uma consulta cara para calcular repetidamente sob demanda. Você pode

usar diferentes opções GROUP BY para as views materializadas criadas na parte superior desta visão materializada e juntar-se a outras tabelas. Fazer isso economiza tempo de computação, caso contrário, usado para executar a junção subjacente cara a cada vez. O STV_MV_DEPS mostra as dependências de uma visão materializada em outras visões materializadas.

Também é possível criar visualizações materializadas que armazenam seus dados como tabelas do Apache Iceberg no Amazon S3 ou em Buckets de Tabelas do Amazon S3 e são registradas no AWS Glue Data Catalog. Essas visualizações materializadas do Iceberg podem ser acessadas por qualquer mecanismo compatível com o formato do Iceberg, incluindo Apache Spark, Amazon Athena, Trino e outros. As visualizações materializadas do Iceberg comportam a atualização incremental e podem ser atualizadas a partir de qualquer cluster do Amazon Redshift com as permissões apropriadas. Para obter mais informações, consulte Visualizações materializadas armazenadas como tabelas do Apache Iceberg.

Quando você cria uma visão materializada, o Amazon Redshift executa a instrução SQL especificada pelo usuário para reunir os dados da tabela ou tabelas base e armazena o conjunto de resultados. A ilustração a seguir fornece uma visão geral da visão materializada `tickets_mv` que uma consulta SQL define usando duas tabelas base, `events` e `sales`.



Em seguida, você pode usar essas visões materializadas em consultas para acelerá-las. Além disso, o Amazon Redshift pode regravar automaticamente essas consultas para usar visões materializadas, mesmo quando a consulta não faz referência explícita a uma visualizações materializada. A

regravação automática de consultas é especialmente poderosa para melhorar a performance quando você não pode alterar suas consultas para usar visões materializadas.

Para atualizar os dados na visão materializada, você pode usar a instrução REFRESH MATERIALIZED VIEW a qualquer momento para atualizar manualmente as visões materializadas. Quando você faz isso, o Amazon Redshift identifica as alterações que ocorreram na tabela ou tabelas base e aplica essas alterações à visão materializada. Como a regravação automática de consultas requer visões materializadas para serem atualizadas, como um proprietário de visão materializada, certifique-se de atualizar visões materializadas sempre que uma tabela base for alterada.

O Amazon Redshift fornece alguns métodos para manter as visões materializadas atualizadas para regravação automática. Você pode configurar visões materializadas com a opção de atualização automática para atualizar visões materializadas quando as tabelas base de visões materializadas forem atualizadas. Essa operação de atualização automática é executada em um momento em que os recursos do cluster estão disponíveis para minimizar interrupções em outros workloads. Como a programação da atualização automática depende do workload, você pode ter mais controle sobre quando o Amazon Redshift atualiza suas visões materializadas. Você pode programar um trabalho de atualização de visão materializada usando a API do programador do Amazon Redshift e a integração do console. Para obter mais informações sobre programação de consultas, consulte Programação de consulta no console do Amazon Redshift.

Fazer isso é especialmente útil quando há um requisito de Acordo de Nível de Serviço (SLA) para dados atualizados de uma visão materializada. Você também pode atualizar manualmente quaisquer visões materializadas que podem ser atualizadas automaticamente. Para obter informações sobre como criar visões materializadas, consulte CREATE MATERIALIZED VIEW.

Você pode emitir instruções SELECT para consultar uma visão materializada. Para obter informações sobre como consultar visões materializadas, consulte Visões materializadas. Eventualmente o conjunto de resultados se torna obsoleto quando os dados são inseridos, atualizados e excluídos nas tabelas base. Você pode recarregar a visão materializada a qualquer momento para atualizá-la com as alterações mais recentes das tabelas base. Para obter informações sobre como atualizar visões materializadas, consulte REFRESH MATERIALIZED VIEW.

Para obter detalhes sobre os comandos SQL usados para criar e gerenciar visões materializadas, consulte os seguintes tópicos de comandos:

- CREATE MATERIALIZED VIEW
- ALTER MATERIALIZED VIEW

- REFRESH MATERIALIZED VIEW
- DROP MATERIALIZED VIEW

Para obter informações sobre tabelas e visualizações do sistema para monitorar visões materializadas, consulte os seguintes tópicos:

- STV_MV_INFO
- STL_MV_STATE
- SVL_MV_REFRESH_STATUS
- STV_MV_DEPS

Tópicos

- Visões materializadas
- Regravação automática de consulta para usar visões materializadas
- Visões materializadas em tabelas externas de data lake no Amazon Redshift Spectrum
- Visualizações materializadas armazenadas como tabelas do Apache Iceberg
- Atualizar uma visualização materializada
- Visões materializadas automatizadas
- Como usar uma função definida pelo usuário (UDF) em uma visão materializada
- Ingestão de streaming para uma visão materializada

Visões materializadas

Você pode usar uma visão materializada em qualquer consulta SQL fazendo referência ao nome da visão materializada como a fonte de dados, como uma exibição em tabela ou padrão.

Quando uma consulta acessa uma visão materializada, ela vê apenas os dados que estão armazenados na atualização mais recente da visão materializada. Portanto, talvez a consulta não veja todas as alterações mais recentes das tabelas base correspondentes da visão materializada.

Se outros usuários quiserem consultar a exibição materializada, o proprietário da exibição materializada deverá conceder a permissão SELECT a esses outros usuários. Os outros usuários não precisam ter a permissão SELECT nas tabelas base subjacentes. O proprietário da visão materializada também pode revogar a permissão SELECT de outros usuários para impedi-los de

consultar a exibição materializada. Os outros usuários ainda precisam da permissão USAGE nos esquemas que contêm as tabelas base da visão materializada.

Se o proprietário da visão materializada não tiver mais a permissão SELECT local nas tabelas base subjacentes:

- O proprietário não poderá mais consultar a visão materializada.
- Os outros usuários que tenham a permissão SELECT na exibição materializada não poderão mais consultar a exibição materializada.

Isso é limitado às permissões locais. As alterações nas permissões gerenciadas pelo Lake Formation não são verificadas ao consultar a visão materializada. Isso significa que, se uma tabela base subjacente for gerenciada pelo Lake Formation e as permissões de seleção na tabela forem revogadas no Lake Formation, você ainda poderá consultar a visão materializada.

O exemplo a seguir consulta a visão materializada `tickets_mv`. Para obter mais informações sobre o comando SQL usado para criar uma visão materializada, consulte CREATE MATERIALIZED VIEW.

```
SELECT sold
FROM tickets_mv
WHERE catgroup = 'Concerts';
```

Como os resultados da consulta são pré-computados, não há necessidade de acessar as tabelas subjacentes (`category`, `event` e `sales`). O Amazon Redshift pode retornar os resultados diretamente do `tickets_mv`.

Regravação automática de consulta para usar visões materializadas

Você pode usar a regravação automática de consultas de visões materializadas no Amazon Redshift para que o Amazon Redshift regrave as consultas para usar visões materializadas. Isso acelera os workloads de consulta, mesmo para consultas que não fazem referência explícita a uma visão materializada. Ao regravar consultas, o Amazon RedShift usa somente visões materializadas atualizadas.

Observações de uso

Para verificar se a regravação automática de consultas é usada para uma consulta, você pode inspecionar o plano de consulta ou STL_EXPLAIN. A seguir é mostrado uma instrução SELECT e a saída EXPLAIN do plano de consulta original.

```
SELECT catgroup, SUM(qtysold) AS sold
FROM category c, event e, sales s
WHERE c.catid = e.catid AND e.eventid = s.eventid
GROUP BY 1;

EXPLAIN
 XN HashAggregate  (cost=920021.24..920021.24 rows=1 width=35)
   ->  XN Hash Join DS_BCAST_INNER  (cost=440004.53..920021.22 rows=4 width=35)
         Hash Cond: ("outer".eventid = "inner".eventid)
         ->  XN Seq Scan on sales s  (cost=0.00..7.40 rows=740 width=6)
         ->  XN Hash  (cost=440004.52..440004.52 rows=1 width=37)
               ->  XN Hash Join DS_BCAST_INNER  (cost=0.01..440004.52 rows=1 width=37)
                     Hash Cond: ("outer".catid = "inner".catid)
                     ->  XN Seq Scan on event e  (cost=0.00..2.00 rows=200 width=6)
                     ->  XN Hash  (cost=0.01..0.01 rows=1 width=35)
                           ->  XN Seq Scan on category c  (cost=0.00..0.01 rows=1
 width=35)
```

A seguir é mostrado a saída EXPLAIN após uma regravação automática bem-sucedida. Essa saída inclui uma varredura na visão materializada no plano de consulta que substitui partes do plano de consulta original.

```
* EXPLAIN
     XN HashAggregate  (cost=11.85..12.35 rows=200 width=41)
       ->  XN Seq Scan on mv_tbl__tickets_mv__0 derived_table1  (cost=0.00..7.90
 rows=790 width=41)
```

Somente visões materializadas atualizadas (novas) são consideradas para regravação automática de consultas, independentemente da estratégia de atualização, como automática, programada ou manual. Assim, a consulta original retorna resultados atualizados. Quando uma visualizações materializada é explicitamente referenciada em consultas, o Amazon Redshift acessa os dados armazenados atualmente na visualizações materializada. Esses dados podem não refletir as últimas alterações das tabelas base da visão materializada.

É possível usar a regravação automática de consultas de visões materializadas criadas na versão de cluster 1.0.20949 ou posterior.

Você pode interromper a regravação automática de consulta no nível da sessão usando SET mv_enable_aqmv_for_session como FALSE.

Como funciona a regravação automática de consulta de visões materializadas

Com base na otimização interna, o Amazon Redshift pode decidir invocar a regravação automática de consultas de visões materializadas de maneira transparente para oferecer a execução de consultas mais adequada e os menores tempos de consulta possíveis.

Por exemplo, suponha que o usuário A crie uma visão materializada M1 na tabela T1 com a consulta `SELECT * FROM T1`. O usuário A tem todos os privilégios SELECT em T1. O usuário A concede acesso à M1 a todos os usuários, incluindo o usuário B. No entanto, quando o usuário B tenta consultar T1 diretamente, o acesso é negado. Isso ocorre porque o usuário B não tem privilégios SELECT em T1.

Algum tempo depois, o usuário B tenta consultar novamente T1, mas dessa vez obtém resultados de T1. Isso ocorre porque a regravação automática de consultas usando a visão materializada regravou a consulta do usuário B na tabela T1 (`SELECT <cols> FROM T1`) em uma consulta na visão materializada M1 (`SELECT <cols> FROM M1`).

Limitações

A seguir estão as limitações para usar a regravação automática de consultas de visões materializadas:

- A regravação automática de consulta funciona com visões materializadas que não fazem referência ou incluem qualquer um dos seguintes itens:
  - Subconsultas
  - Esquerda, direita ou junções externas completas
  - Operações de conjunto
  - Todas as funções agregadas, exceto SUM, COUNT, MIN e MAX e AVG; (Essas são as únicas funções agregadas que funcionam com a reescrita automática de consultas.)
  - Todas as funções agregadas com DISTINCT
  - Todas as funções da janela

  - Cláusulas SELECT DISTINCT ou HAVING
  - Tabelas externas ou compartilhadas
  - Outras visões materializadas
- A reescrita automática de consulta regrava consultas SELECT que se referem a tabelas definidas pelo usuário do Amazon Redshift. O Amazon Redshift não regrava as seguintes consultas:
  - Instruções CREATE TABLE AS
  - Instruções SELECT INTO
  - Consultas em catálogos ou tabelas de sistema
  - Consultas com junções externas ou uma cláusula SELECT DISTINCT
- Se uma consulta não for regravada automaticamente, confira se você tem o privilégio SELECT na visão materializada especificada e se a opção mv_enable_aqmv_for_session está definida como TRUE.

  Você também pode verificar se suas visões materializadas são elegíveis para regravação automática de consultas inspecionando STV_MV_INFO. Para obter mais informações, consulte STV_MV_INFO.

Visões materializadas em tabelas externas de data lake no Amazon Redshift Spectrum

As visões materializadas podem fornecer manutenção incremental em tabelas externas de data lake. Em uma atualização incremental, o Amazon Redshift atualiza os dados na visão materializada apenas com as alterações dos dados nas tabelas básicas desde a última atualização. A manutenção incremental é mais econômica do que recalcular totalmente a visão materializada após cada alteração de dados na tabela básica.

Quando você usa visões materializadas em pelo menos uma tabela externa, a criação de visões materializadas é incremental em:

- Tabelas padrão de data lake, particionadas e não particionadas, com arquivos de dados em qualquer formato compatível (Parquet, Avro, CSV etc.).
- Tabelas do Apache Iceberg, particionadas e não particionadas, com cópia na gravação e mesclagem na leitura.
- Tabelas do Amazon Redshift Spectrum unidas a qualquer tabela do Amazon Redshift no mesmo banco de dados.

A atualização da visão materializada é incremental em:

- Tabelas de data lake padrão após a substituição de S3 DELETE ou PUT (exclusão de arquivos de dados), se a visão materializada não realizar a agregação.
- Tabelas do Apache Iceberg após INSERT, DELETE, UPDATE ou compactação da tabela.

Para obter mais informações sobre o Amazon Redshift Spectrum, consulte Amazon Redshift Spectrum.

Limitações

As limitações gerais das visões materializadas ainda se aplicam às visões materializadas em tabelas de data lake. Para obter mais informações, consulte Atualizar uma visualização materializada. Além disso, considere as seguintes limitações ao usar visões materializadas em tabelas externas de data lake.

- A criação de visões materializadas não é incremental em:
  - Tabelas do Hudi ou Delta Lake.
  - Acesso a dados aninhados ao espectro.
  - Referências a colunas VARBYTE.
- A atualização da visão materializada volta ao recálculo completo em:
  - Tabelas do Apache Iceberg quando um snapshot necessário expira, se a visão materializada realizar a agregação.
  - Tabelas de data lake padrão após a exclusão ou atualização dos arquivos de dados no Amazon S3, se a visão materializada realizar a agregação.
  - As tabelas de data lake padrão foram atualizadas mais de uma vez em um bloco de transação.
  - Tabelas de data lake padrão governadas por um manifesto. Para obter mais informações sobre manifestos, consulte Uso de um manifesto para especificar arquivos de dados.
  - O Amazon Redshift voltará ao recálculo completo se achar que melhora o desempenho, especialmente para visões materializadas que contêm junções e se mais de uma tabela básica foi atualizada desde a última atualização.
- Nas tabelas do Apache Iceberg, a atualização da visão materializada pode lidar com apenas até 4 milhões de posições excluídas em um único arquivo de dados. Depois que esse limite for atingido, a tabela base do Apache Iceberg deverá ser compactada para que seja possível continuar atualizando a visualização materializada.

- Nas tabelas do Apache Iceberg, não há suporte para o escalonamento simultâneo na criação e na atualização de visões materializadas.
- Não há suporte para os recursos do Autonomics. Isso inclui visões materializadas automatizadas e reescrita automática de consultas.
- Quando uma visão materializada incremental é atualizada, as permissões do IAM se aplicam somente às partes acessadas das tabelas básicas do Amazon Redshift.
- As alterações nas permissões gerenciadas pelo Lake Formation não são verificadas ao consultar uma visão materializada. Isso significa que, se uma visão materializada for definida em uma tabela de data lake e os privilégios de seleção forem removidos da tabela com o Lake Formation, você ainda poderá consultar a visão materializada.

Visualizações materializadas armazenadas como tabelas do Apache Iceberg

É possível criar visualizações materializadas que armazenam seus dados como tabelas do Apache Iceberg em Buckets do Amazon S3 ou Tabelas do Amazon S3. Diferentemente das visualizações materializadas padrão do Amazon Redshift, que armazenam dados internamente no Redshift Managed Storage (RMS), as visualizações materializadas do Iceberg são armazenadas como tabelas padrão do Iceberg e podem ser acessadas por qualquer mecanismo de analytics compatível com o formato Iceberg, incluindo Apache Spark, Amazon Athena e Trino.

As visualizações materializadas do Iceberg combinam os recursos de otimização de consultas e atualização incremental do Amazon Redshift com o formato de tabela aberta do Iceberg. Você pode criar, atualizar e eliminar visualizações materializadas do Iceberg usando os mesmos comandos SQL das visualizações materializadas padrão, com a adição da cláusula `USING ICEBERG` durante a criação.

As visualizações materializadas do Iceberg são aceitas no Redshift sem servidor e em clusters provisionados com tipos de instância RG. Os tipos de instância RA3 e DC2 não são aceitos.

Para acessar informações sobre os comandos SQL usados para criar e gerenciar visualizações materializadas do Iceberg, consulte os seguintes tópicos de comandos:

- CREATE MATERIALIZED VIEW
- REFRESH MATERIALIZED VIEW
- DROP MATERIALIZED VIEW

Como as visualizações materializadas do Iceberg funcionam

Quando você cria uma visualização materializada do Iceberg, o Amazon Redshift faz o seguinte:

1. Executa a consulta de definição e grava os resultados como arquivos de dados do Parquet no Amazon S3.
2. Registra a tabela no AWS Glue Data Catalog com metadados do Iceberg, tornando-a detectável por outros mecanismos.
3. Armazena a definição da visualização materializada e o estado de atualização em AWS Glue para gerenciamento independente do cluster.

Quando você atualiza uma visualização materializada do Iceberg, o Amazon Redshift compara os snapshots do Iceberg da tabela de origem atual com os snapshots registrados na última atualização. Se forem detectadas alterações, o Amazon Redshift vai calcular e aplicar somente os dados alterados (atualização incremental) ou realizar um novo cálculo completo.

Requisitos da tabela de origem

As tabelas de origem para visualizações materializadas do Iceberg devem atender aos seguintes requisitos:

- Devem estar no formato Apache Iceberg. Tabelas que não sejam do Iceberg (tabelas nativas do Amazon Redshift, tabelas externas do Hive ou do Parquet) não são aceitas como tabelas de origem.
- Devem residir na mesma Região da AWS e conta que a visualização materializada.
- Podem ser referidas usando a notação de três partes (`awsdatacatalog.`*`database`*`.`*`table`*), a notação de Tabelas do Amazon S3 (`"`*`bucket`*`@s3tablescatalog".`*`database`*`.`*`table`*) ou a notação de duas partes com um esquema externo (*`external_schema`*`.`*`table`*).

Interoperabilidade entre mecanismos

As visualizações materializadas do Iceberg criadas pelo Amazon Redshift aparecem como tabelas padrão do Iceberg no AWS Glue Data Catalog. Outros mecanismos de analytics podem lê-los usando os padrões padrão de acesso à tabela do Iceberg sem nenhuma configuração adicional.

As seguintes restrições são aplicáveis:

- Somente o Amazon Redshift pode atualizar ou eliminar visualizações materializadas que foram criadas pelo Amazon Redshift.
- As visualizações materializadas do Iceberg criadas por outros mecanismos (como o Apache Spark) podem ser consultadas pelo Amazon Redshift como tabelas somente leitura, mas não podem ser atualizadas nem eliminadas pelo Amazon Redshift.

Atualização incremental

O Amazon Redshift executa uma atualização incremental quando a definição de visualização materializada a aceita. A atualização incremental lê somente os dados alterados nas tabelas de origem desde a última atualização e aplica essas alterações à visualização materializada.

A atualização incremental é aceita em visualizações materializadas que usam:

- `SELECT ... FROM ... WHERE ... GROUP BY` com funções agregadas COUNT e SUM.
- Junções internas entre tabelas de origem do Iceberg.

O Amazon Redshift volta automaticamente à atualização completa quando:

- A visualização materializada usa constructos SQL não aceitos na atualização incremental. Para acessar mais informações, consulte a seção Limitações abaixo e Atualizar uma visualização materializada.
- Os snapshots da tabela de origem da última atualização expiraram.
- Os dados da visualização materializada foram modificados fora de uma operação de atualização do Amazon Redshift.

Atualização simultânea

Vários clusters ou grupos de trabalho do Amazon Redshift podem tentar atualizar a mesma visualização materializada do Iceberg simultaneamente. O Amazon Redshift usa o controle de simultaneidade otimista (OCC) por meio do mecanismo de atualização condicional do AWS Glue Data Catalog para garantir que apenas uma atualização seja bem-sucedida. Se uma atualização simultânea for detectada, o Amazon Redshift vai conferir se a visualização materializada ainda está obsoleta. Se outro cluster concluir a atualização primeiro, a operação local será interrompida.

Compactação e manutenção de tabelas

Para visualizações materializadas do Iceberg armazenadas nos buckets de Tabelas do Amazon S3, as Tabelas do Amazon S3 gerenciam automaticamente a compactação e a otimização do arquivo de dados. Nenhuma ação de usuário é necessária.

Para visualizações materializadas armazenadas em buckets do Amazon S3 de uso geral (especificados por meio do parâmetro `LOCATION`), o Amazon Redshift não realiza compactação. Com o tempo, a atualização incremental produz muitos arquivos de dados pequenos que podem degradar a performance de leitura. Para manter a performance de leitura, use as seguintes práticas recomendadas:

- Execute a compactação regularmente usando uma ferramenta externa, como o procedimento `rewrite_data_files` do Apache Spark.
- Configure a expiração dos snapshots do Iceberg para limitar o crescimento do armazenamento a partir de snapshots históricos.

Para tabelas de origem, configure a retenção de snapshots para exceder o intervalo de atualização esperado. Quando os snapshots da tabela de origem expiram antes da próxima atualização, o Amazon Redshift não pode mais calcular um delta incremental e volta para a atualização completa.

Considerações sobre segurança

Permissões

As seguintes permissões são necessárias:

- CREATE — o chamador precisa da permissão CREATE TABLE no banco de dados AWS Glue de destino. O perfil do IAM associado ao esquema externo (o perfil definidor de MV) deve ter a permissão SELECT em todas as tabelas de origem, concedida por meio de políticas de recursos AWS Lake Formation ou AWS Glue.
- REFRESH — o chamador precisa da permissão ALTER na visualização materializada. O perfil definidor de MV (registrado no momento da criação) deve ter a permissão SELECT em todas as tabelas de origem.
- DROP — o chamador precisa da permissão ALTER na visualização materializada.
- QUERY — o chamador precisa da permissão SELECT na visualização materializada, concedida por meio das políticas de recursos AWS Lake Formation ou AWS Glue.

Para acessar informações sobre a criação de perfis do IAM para o Amazon Redshift e a concessão de permissões AWS Lake Formation, consulte Conceitos básicos do Amazon Redshift Spectrum. Para verificar se o perfil do IAM tem as permissões necessárias nas tabelas de origem, use o console do AWS Lake Formation ou a API `GetEffectivePermissionsForPath`. Para verificar se o banco de dados AWS Glue de destino existe e se o perfil tem acesso, use a API `GetDatabase`.

O exemplo a seguir verifica se o perfil do IAM tem a permissão SELECT em uma tabela de origem por meio de AWS Lake Formation:

```
aws lakeformation list-permissions \
    --principal DataLakePrincipalIdentifier=arn:aws:iam::123456789012:role/
myspectrum_role \
    --resource '{"Table":{"DatabaseName":"mydb","Name":"orders"}}'
```

O seguinte exemplo verifica se o banco de dados AWS Glue de destino existe e está acessível:

```
aws glue get-database --name mydb
```

Visibilidade do histórico de snapshots

As visualizações materializadas do Iceberg armazenam dados como tabelas padrão do Iceberg com histórico completo de snapshots. Um leitor com acesso aos arquivos do Iceberg subjacentes pode comparar snapshots sucessivos para observar quais linhas foram adicionadas ou removidas entre as atualizações.

Para visualizações materializadas agregadas (aquelas com GROUP BY e funções agregadas), o delta visível é limitado a alterações nos valores agregados, não às linhas individuais da tabela de origem. Para visualizações materializadas não agregadas, o delta corresponde às linhas específicas que foram alteradas na saída da visualização materializada.

Esse comportamento é inerente ao formato da tabela do Iceberg. Qualquer tabela do Iceberg que passa por gravações incrementais expõe a mesma propriedade.

Para mitigar esse comportamento:

- Use permissões do AWS Lake Formation ou políticas de bucket do Amazon S3 para controlar quem pode acessar os arquivos de dados do Iceberg subjacentes da visualização materializada.
- Para visualizações materializadas que contenham dados sensíveis em que o histórico de alterações não deve ser observável, defina a visualização materializada com constructos SQL que

forçam a atualização completa (como DISTINCT). A atualização completa substitui todos os dados em cada snapshot, tornando inviável determinar quais linhas específicas foram alteradas.

- Configure a expiração do snapshot do Iceberg para limitar a janela de alterações observáveis. Com os Buckets de Tabelas do Amazon S3, a manutenção gerenciada controla automaticamente o ciclo de vida dos snapshots.
- Use Buckets de Tabelas do Amazon S3, que fornecem controle de acesso integrado em nível de tabela.

Monitoramento

Histórico de atualizações

As operações de atualização realizadas pelo cluster local são registradas em log em SVL_MV_REFRESH_STATUS. Consulte essa visualização para conferir o tempo e o status da atualização e se a atualização foi incremental ou completa.

```
SELECT mv_name, starttime, endtime, status
FROM svl_mv_refresh_status
WHERE mv_name = 'daily_revenue'
ORDER BY starttime DESC
LIMIT 10;
```

Observe que essa visualização do sistema registra somente as atualizações realizadas pelo cluster local. Se vários clusters atualizarem a mesma visualização materializada do Iceberg, as tabelas do sistema de cada cluster mostrarão somente seu próprio histórico de atualizações.

Descobrir visualizações materializadas do Iceberg

Use o comando SHOW TABLES para encontrar visualizações materializadas do Iceberg. As visualizações materializadas do Iceberg aparecem com `table_type = 'EXTERNAL TABLE'` e `table_subtype = 'ICEBERG MATERIALIZED VIEW'`. SHOW TABLES funciona em todos os caminhos do catálogo (esquemas externos, `awsdatacatalog` e Tabelas do Amazon S3).

```
SHOW TABLES FROM SCHEMA awsdatacatalog.mydb LIKE '%';
```

Você também pode usar SVV_EXTERNAL_TABLES para encontrar visualizações materializadas do Iceberg em seus esquemas externos, onde elas aparecem com `tabletype = 'EXTERNAL TABLE'`:

```
SELECT schemaname, tablename, location
FROM svv_external_tables
WHERE tabletype = 'EXTERNAL TABLE';
```

> **Note**
>
> SVV_EXTERNAL_TABLES enumera visualizações materializadas do Iceberg somente para esquemas externos. As visualizações materializadas do Iceberg acessadas por meio de `awsdatacatalog` de catálogos de Tabelas do Amazon S3 não são enumeradas por SVV_EXTERNAL_TABLES; use SHOW TABLES para descobri-las.

STV_MV_INFO não inclui visualizações materializadas do Iceberg.

Limitações

Você não pode definir uma visualização materializada do Iceberg que faça referência ou inclua o seguinte:

- Tabelas de origem que não sejam do Iceberg (tabelas nativas do Amazon Redshift, tabelas externas do Hive, do Parquet, do Delta Lake ou do Hudi)
- Tabelas de origem que usam o formato do Iceberg versão 3
- Funções definidas pelo usuário (UDFs) ou agregados definidos pelo usuário
- Funções mutáveis (GETDATE, RANDOM, CURRENT_TIMESTAMP e funções similares)
- Cláusulas ORDER BY, LIMIT ou OFFSET
- Identificadores em maiúsculas em nomes de tabelas, nomes de colunas ou aliases
- Tabelas de origem em uma Região da AWS ou conta diferente
- Tabelas filtradas do AWS Lake Formation (controle de acesso refinado)

Os seguintes constructos SQL são permitidos na definição da visualização materializada, mas forçam a atualização completa em vez da atualização incremental:

- DISTINCT
- Junções externas (LEFT, RIGHT, FULL)
- Funções de janela

- Subconsultas
- Operações de conjunto
- GROUPING SETS, ROLLUP, CUBE
- Funções agregadas além de COUNT e SUM
- COUNT(DISTINCT), SUM(DISTINCT)

Limitações adicionais:

- A reescrita automática de consultas para usar visualizações materializadas não é aceita em visualizações materializadas do Iceberg.
- As visualizações materializadas automatizadas não são aceitas nas visualizações materializadas do Iceberg.
- A atualização automática não é aceita em visualizações materializadas do Iceberg. Você deve atualizá-las manualmente usando REFRESH MATERIALIZED VIEW.
- As visualizações materializadas do Iceberg são aceitas somente no Redshift sem servidor e em clusters provisionados com tipos de instância RG. Os tipos de instância RA3 e DC2 não são aceitos.
- O controle de acesso refinado (FGAC) não e aceito em visualizações materializadas do Iceberg.
- A atualização CASCADE não é aceita em visualizações materializadas do Iceberg.
- As visualizações materializadas do Iceberg não podem ser criadas nem atualizadas quando `enable_case_sensitive_identifier` está definido como `true`.

Atualizar uma visualização materializada

Este tópico descreve como atualizar os dados em uma visão materializada das tabelas subjacentes.

Quando você cria uma visão materializada, o respectivo conteúdo mostra o estado das relações entre bancos de dados subjacentes (tabelas ou outra visões materializadas) naquele momento. Os dados da visão materializada permanecem inalterados, mesmo quando as aplicações fazem alterações nos dados nas tabelas subjacentes. Para atualizar os dados na visão materializada, você pode usar a instrução `REFRESH MATERIALIZED VIEW` a qualquer momento para atualizar manualmente as visões materializadas. Ao usar essa instrução, o Amazon Redshift identifica as alterações que ocorreram nas relações básicas e aplica essas alterações à visão materializada.

O Amazon Redshift tem duas estratégias para atualizar uma visão materializada:

- Em muitos casos, o Amazon Redshift pode realizar uma atualização incremental. Em uma atualização incremental, o Amazon Redshift identifica rapidamente as alterações nos dados nas relações básicas desde a última atualização e atualiza os dados na visão materializada. A atualização incremental é compatível com as seguintes construções SQL usadas na consulta ao definir a visão materializada:
  - Construções que contêm as cláusulas SELECT, FROM, [INNER] JOIN, WHERE, GROUP BY ou HAVING.
  - Construções que contêm agregações, como SUM, MIN, MAX, AVG e COUNT.
  - A maioria das funções SQL integradas, especificamente aquelas que são imutáveis, visto que têm os mesmos argumentos de entrada e sempre produzem a mesma saída.

  A atualização incremental também é compatível para uma visão materializada baseada em uma tabela da unidade de compartilhamento de dados ou visão materializada.
- Se uma atualização incremental não for possível, o Amazon Redshift executa uma atualização completa. Uma atualização total executa novamente a instrução SQL subjacente substituindo todos os dados na visão materializada.
- O Amazon Redshift seleciona automaticamente o método de atualização para uma visão materializada, dependendo da consulta SELECT usada para defini-la.

Visões materializadas aninhadas

É possível definir uma visão materializada acima de outras visões materializadas. Para atualizar essa visão materializada, você deve usar explicitamente a palavra-chave `CASCADE` na atualização da visão materializada no nível mais alto. Por exemplo, considere a seguinte estrutura de visão materializada aninhada:

```
CREATE TABLE t(a INT);
CREATE MATERIALIZED VIEW u AS SELECT * FROM t;
CREATE MATERIALIZED VIEW v AS SELECT * FROM u;
CREATE MATERIALIZED VIEW w AS SELECT * FROM v;

-- w -> v -> u -> t

INSERT INTO t VALUES (1);
```

Para atualizar a visão w totalmente, há duas opções:

- (Recomendado) Atualize a visão w usando o comando `REFRESH MATERIALIZED VIEW w CASCADE`. Esse comando executa a atualização de todas as visões materializadas em uma única transação.
- Atualize u, v e w como comandos separados, em ordem de dependência (primeiro u, depois v, depois w).

Se a palavra-chave `CASCADE` não for usada explicitamente, a visão materializada será atualizada no modo `RESTRICT`, atualizando somente a visão materializada atual. O exemplo a seguir mostra uma mensagem informativa quando você executa `REFRESH MATERIALIZED VIEW` em uma visão materializada que depende de uma visão materializada desatualizada.

```
REFRESH MATERIALIZED VIEW w;
INFO:  Materialized view w is already up to date.  However, it depends on another
 materialized view that is not up to date.

REFRESH MATERIALIZED VIEW w CASCADE;
INFO:  Materialized view w was incrementally updated successfully.
```

```
REFRESH MATERIALIZED VIEW v;
INFO: Materialized view v is already up to date. However, it depends on another
 materialized view that is not up to date.

REFRESH MATERIALIZED VIEW v CASCADE;
INFO: Materialized view v was incrementally updated successfully.
```

Nos exemplos acima com a opção de atualização em cascata, a visão materializada u é atualizada primeiro, a visão materializada v é atualizada em seguida e a visão materializada w não é atualizada.

O exemplo a seguir mostra como criar um plano de atualização completo para uma visão materializada programaticamente. Para atualizar a visão materializada v, primeiro atualize a visão materializada u. Para atualizar a visão materializada w, primeiro atualize a visão materializada u e, em seguida, a visão materializada v.

```
WITH RECURSIVE recursive_deps (mv_tgt, lvl, mv_dep) AS
( SELECT trim(name) as mv_tgt, 0 as lvl, trim(ref_name) as mv_dep
  FROM stv_mv_deps
  UNION ALL
  SELECT R.mv_tgt, R.lvl+1 as lvl, trim(S.ref_name) as mv_dep
  FROM stv_mv_deps S, recursive_deps R
```

```
  WHERE R.mv_dep = S.name
)

SELECT mv_tgt, mv_dep from recursive_deps
ORDER BY mv_tgt, lvl DESC;

 mv_tgt | mv_dep
--------+--------
 v      | u
 w      | u
 w      | v
(3 rows)
```

Limitações

O Amazon Redshift não permite a atualização em cascata de visões materializadas com base em outras origens além de:

- Tabelas locais
- Tabelas compartilhadas
- VMs locais
- MVs de streaming

A atualização em cascata não se estende às visualizações materializadas compartilhadas.

O Amazon Redshift não oferece suporte à atualização incremental para visões materializadas que são definidas com uma consulta usando os seguintes elementos SQL:

- OUTER JOIN (RIGHT, LEFT ou FULL).
- As operações definidas UNION, INTERSECT, EXCEPT e MINUS.
- As funções de agregação MEDIAN, PERCENTILE_CONT, LISTAGG, STDDEV_SAMP, STDDEV_POP, APPROXIMATE COUNT e APPROXIMATE PERCENTILE e funções de agregação bit a bit.

  > **Note**
  >
  > As funções agregadas COUNT, SUM e AVG são aceitas.

- Funções agregadas DISTINCT, como DISTINCT COUNT, DISTINCT SUM e assim por diante.

- Funções de janela.
- Uma consulta que usa tabelas temporárias para otimização de consultas, como para otimizar subexpressões comuns.
- Subconsultas.
- Tabelas externas referenciando os formatos a seguir na consulta que define a visão materializada.
  - Delta Lake
  - Hudi

  Há suporte para a atualização incremental para visões materializadas definidas usando formatos diferentes dos listados acima. Para obter mais informações, consulte Visões materializadas em tabelas externas de data lake no Amazon Redshift Spectrum.

Atualizar automaticamente uma visão materializada

O Amazon Redshift pode atualizar automaticamente as visões materializadas com dados atualizados de suas tabelas base quando as visões materializadas são criadas ou alteradas para ter a opção de atualização automática. O Amazon Redshift atualiza automaticamente as visões materializadas o mais rápido possível após as alterações nas tabelas base.

Para concluir a atualização das visões materializadas mais importantes com impacto mínimo nos workloads ativos em seu cluster, o Amazon Redshift considera vários fatores. Esses fatores incluem a carga atual do sistema, os recursos necessários para atualização, os recursos de cluster disponíveis e a frequência com que as visões materializadas são usadas.

O Amazon Redshift prioriza suas workloads sobre a atualização automática e pode interromper a atualização automática para preservar a performance da workload do usuário. Essa abordagem pode atrasar a atualização de algumas visões materializadas. Em alguns casos, você precisará de um comportamento de atualização mais determinístico para as visões materializadas. Nesse caso, considere usar a atualização manual conforme descrito em REFRESH MATERIALIZED VIEW ou atualização programada usando as operações da API do programador do Amazon Redshift ou o console.

Você pode definir a atualização automática para visões materializadas usando CREATE MATERIALIZED VIEW. Você também pode usar a cláusula AUTO REFRESH para atualizar visões materializadas automaticamente. Para obter mais informações sobre como criar visões materializadas, consulte CREATE MATERIALIZED VIEW. Você pode ativar a atualização automática para uma visualizações materializada atual usando ALTER MATERIALIZED VIEW.

Considere o seguinte ao atualizar visões materializadas:

- Você ainda pode atualizar uma visualizações materializada explicitamente usando o comando REFRESH MATERIALIZED VIEW mesmo que não tenha ativado a renovação automática para a visualizações materializada.
- É possível aplicar atualização automática em visões materializadas definidas em tabelas de unidade de compartilhamento de dados ou tabelas Iceberg.
- Para status de atualização, você pode verificar SVL_MV_REFRESH_STATUS, que registra consultas iniciadas pelo usuário ou atualizadas automaticamente.
- Para executar a atualização REFRESH com recálculo total de visualizações materializadas, certifique-se de que você possui a permissão CREATE nos esquemas. Para obter mais informações, consulte GRANT.

A partir de 27 de fevereiro de 2026, as consultas de atualização automática para visões materializadas do Amazon Redshift são executadas como consultas de usuários, em vez de processos autônomos em segundo plano. Como resultado, as consultas de atualização automática agora são executadas com a mesma prioridade que as consultas de outros usuários.

Essa alteração melhora a atualização das visões materializadas com a atualização automática habilitada, ajudando-as a se manter mais atualizadas com as alterações mais recentes em suas tabelas base em comparação ao comportamento anterior.

Observação: o recurso de alteração de comportamento de atualização automática de MV só está habilitado para clusters provisionados do Amazon Redshift na faixa CURRENT da versão de patch P198 e mais recente. Atualmente, ele está desabilitado na tecnologia sem servidor.

Visões materializadas automatizadas

Este tópico descreve como o Amazon Redshift usa visões materializadas automatizadas para melhorar a performance. O Amazon Redshift cria visões materializadas automaticamente com base na atividade e na performance do banco de dados. O Amazon Redshift usa visões materializadas automatizadas por padrão.

As visões materializadas são uma ferramenta poderosa para melhorar a performance da consulta no Amazon Redshift. Elas fazem isso armazenando um conjunto de resultados pré-calculado. Consultas semelhantes não precisam executar novamente a mesma lógica toda vez, pois elas podem recuperar registros do conjunto de resultados existente. Desenvolvedores e analistas criam

visões materializadas depois de analisar suas workloads para determinar quais consultas trariam benefício e se o custo de manutenção de cada visão materializada vale a pena. À medida que as workloads crescem ou mudam, essas visualizações materializadas devem ser revisadas para garantir que continuem proporcionando benefícios de desempenho concretos.

O recurso Visão Materializada Automatizada (AutoMV) no Redshift aprimora a performance das consultas criando e gerenciando automaticamente as visões materializadas com base no monitoramento de workloads e em algoritmos de machine learning. Veja abaixo os principais recursos da AutoMV:

- Monitoramento contínuo: o Redshift monitora continuamente a workload usando técnicas de machine learning para identificar oportunidades de melhoria de performance por meio da criação de visões materializadas.
- Criação e exclusão automáticas: quando o sistema detecta que uma visão materializada pode ser benéfica, ele a cria e a mantém automaticamente. Ao passo que, se uma AutoMV criada anteriormente parar de fornecer benefícios de performance, o sistema a descartará automaticamente.
- Não há necessidade de atividade do usuário: o recurso AutoMV opera somente durante períodos de baixa atividade do usuário ou da workload em execução no cluster. Isso garante que as operações da AutoMV não interfiram nem afetem as workloads do cliente.
- Picos de uso da CPU: durante períodos sem atividade da workload, a criação ou a atualização de visões materializadas pela AutoMV pode causar picos no uso da CPU. Esse é um comportamento normal, pois o sistema utiliza os recursos disponíveis para criar e atualizar as visões materializadas.
- Prioridade da workload do usuário: se você iniciar uma workload enquanto uma operação da AutoMV estiver em andamento, a tarefa da AutoMV será interrompida para liberar recursos para a workload do usuário. Isso garante que suas workloads tenham prioridade sobre as operações da AutoMV.

Embora o recurso AutoMV possa causar picos de uso da CPU durante períodos sem atividade do usuário, ele opera de forma transparente e sem afetar as workloads. O sistema gerencia visões materializadas para melhorar a performance das consultas e, simultaneamente, prioriza as workloads do usuário em relação às operações da AutoMV.

O comportamento e os recursos da AutoMV são os mesmos que as visões materializadas criadas pelo usuário. São atualizadas automaticamente e incrementalmente, usando os mesmos critérios e restrições. Assim como as visões materializadas criadas pelo usuário, o Regravação automática

de consulta para usar visões materializadas identifica consultas que podem se beneficiar das AutoMVs criadas pelo sistema. Ele regrava automaticamente essas consultas para usar o AutoMVS, melhorando a performance da consulta. Os desenvolvedores não precisam revisar as consultas para aproveitar a AutoMV.

> **Note**
>
> As visões materializadas automatizadas são atualizadas de forma intermitente. As consultas regravadas para usar o AutoMV sempre retornam os resultados mais recentes. Quando o Redshift detecta que os dados não estão atualizados, as consultas não são regravadas para serem lidas de visões materializadas automatizadas. Em vez disso, as consultas selecionam os dados mais recentes das tabelas de base.

Qualquer workload com consultas usadas repetidamente pode se beneficiar da AutoMV. Entre os casos de uso comuns estão:

- Painéis: os painéis são bastante utilizados para fornecer visualizações rápidas dos principais indicadores de negócios (KPIs), eventos, tendências e outras métricas. Geralmente apresentam um layout comum com gráficos e tabelas, mas mostram diferentes visualizações para filtragem ou para operações de seleção de dimensões, como busca detalhada para baixo. Os painéis muitas vezes têm um conjunto comum de consultas usadas repetidamente com parâmetros diferentes. As consultas de painel podem se beneficiar muito de visões materializadas automatizadas.
- Relatórios: as consultas de relatórios podem ser programadas em várias frequências, com base nos requisitos de negócios e no tipo de relatório. Além disso, podem ser automatizadas ou sob demanda. Uma característica comum das consultas de relatórios é que elas podem ser de longa duração e uso intensivo de recursos. Com a AutoMV, essas consultas não precisam ser recalculadas sempre que forem executadas, o que reduz o ambiente de tempo de execução para cada consulta e utilização de recursos no Redshift.

Para desativar as visões materializadas automatizadas, atualize o grupo de parâmetros `auto_mv` para `false`. Para obter mais informações, consulte Grupos de parâmetros do Amazon Redshift no Guia de gerenciamento de clusters do Amazon Redshift.

Escopo e considerações de SQL para visões materializadas automatizadas

- Uma visão materializada automática pode ser iniciada e criada por uma consulta ou subconsulta, desde que contenha uma cláusula `GROUP BY` ou uma das seguintes funções agregadas: SUM, COUNT, MIN, MAX ou AVG. No entanto, ela não pode conter nenhum destes itens:
  - Esquerda, direita ou junções externas completas
  - Funções agregadas, como SUM, COUNT, MIN, MAX e AVG. (Essas funções específicas funcionam com a regravação de consulta automática.)
  - Qualquer função agregada que inclua DISTINCT.
  - Todas as funções da janela
  - Cláusulas SELECT DISTINCT ou HAVING
  - Outras visões materializadas

  Não é garantido que uma consulta que atenda aos critérios iniciará a criação de uma visão materializada automática. O sistema determina de quais candidatos se deve criar uma visão, com base no benefício esperado para a workload e no custo dos recursos a serem mantidos, o que inclui o custo de atualização do sistema. Cada visão materializada resultante é utilizável pela regravação de consulta automática.
- Mesmo que a AutoMV possa ser iniciada por uma subconsulta ou trechos individuais de operadores de conjunto, a visão materializada resultante não conterá subconsultas nem operadores de conjunto.
- Para determinar se a AutoMV foi usada para consultas, veja o plano EXPLAIN e procure `%_auto_mv_%` na saída. Para obter mais informações, consulte EXPLAIN.
- As visões materializadas automatizadas não são compatíveis em tabelas externas, como unidades de compartilhamento de dados e tabelas federadas.

Limitações das visões materializadas automatizadas

Veja a seguir as limitações para trabalhar com visões materializadas:

- Número máximo de AutoMVs: o limite de visões materializadas automatizadas é 200 por banco de dados no cluster.
- Espaço e capacidade de armazenamento: uma característica importante da AutoMV é que ela é executada usando ciclos de fundo sobressalentes para ajudar a garantir que as workloads do usuário não sejam afetadas. Se o cluster estiver ocupado ou ficar sem espaço de armazenamento,

o AutoMV interromperá sua atividade. Especificamente, com 80% da capacidade total do cluster, nenhuma nova visão materializada automatizada é criada. Com 90% da capacidade total, elas podem ser descartadas para viabilizar que as workloads do usuário continuem sem degradação da performance. Para obter mais informações sobre como determinar a capacidade de clusters, consulte STV_NODE_STORAGE_CAPACITY.

Faturamento para visões materializadas automatizadas

O recurso de otimização automática do Amazon Redshift cria e atualiza visões materializadas automatizadas. Não há cobrança por recursos computacionais para esse processo. O armazenamento de visões materializadas automatizadas é cobrado de acordo com a taxa normal de armazenamento. Para obter mais informações, consulte Preços do Amazon Redshift.

Recursos adicionais

A postagem do blog a seguir fornece mais explicações sobre visões materializadas automatizadas. Ele detalha como eles são criados, mantidos e descartados. Também explica os algoritmos subjacentes que impulsionam essas decisões: Optimize your Amazon Redshift query performance with automated materialized views (Otimizar a performance de suas consultas do Amazon Redshift com visões materializadas automatizadas).

Este vídeo começa com uma explicação das visões materializadas e mostra como elas melhoram a performance e conservam recursos. Depois, ele fornece uma explicação detalhada das visões materializadas automatizadas com uma animação de fluxo de processo e uma demonstração ao vivo.

Como usar uma função definida pelo usuário (UDF) em uma visão materializada

Você pode usar uma UDF escalar em uma visão materializada do Amazon Redshift. Defina essas funções em Python ou SQL e faça referência a elas na definição da visão materializada.

Como fazer referência a uma UDF em uma visão materializada

O procedimento a seguir mostra como usar UDFs que realizam comparações aritméticas simples em uma definição de visão materializada.

1. Crie uma tabela para usar na definição de visão materializada.

```
CREATE TABLE base_table (a int, b int);
```

2. Crie uma função escalar definida pelo usuário em Python que retorne um valor booleano indicando se um inteiro é maior do que outro inteiro de comparação.

```
CREATE OR REPLACE FUNCTION udf_python_bool(x1 int, x2 int) RETURNS bool IMMUTABLE
AS $$
  return x1 > x2
$$ LANGUAGE plpythonu;
```

Opcionalmente, crie uma UDF funcionalmente semelhante com SQL, que você pode usar para comparar os resultados com a primeira.

```
CREATE OR REPLACE FUNCTION udf_sql_bool(int, int) RETURNS bool IMMUTABLE
AS $$
  select $1 > $2;
$$ LANGUAGE SQL;
```

3. Crie uma visão materializada que faça uma seleção na tabela que você criou e faça referência à UDF.

```
CREATE MATERIALIZED VIEW mv_python_udf AS SELECT udf_python_bool(a, b) AS a FROM
 base_table;
```

Opcionalmente, você pode criar uma visão materializada que faça referência à UDF em SQL.

```
CREATE MATERIALIZED VIEW mv_sql_udf AS SELECT udf_sql_bool(a, b) AS a FROM
 base_table;
```

4. Adicione dados à tabela e atualize a visão materializada.

```
INSERT INTO base_table VALUES (1,2), (1,3), (4,2);
```

```
REFRESH MATERIALIZED VIEW mv_python_udf;
```

Opcionalmente, você pode atualizar a visão materializada que faz referência à UDF em SQL.

```
REFRESH MATERIALIZED VIEW mv_sql_udf;
```

5. Consulte dados da visão materializada.

```
SELECT * FROM mv_python_udf ORDER BY a;
```

Os resultados da consulta são os seguintes:

```
a
-----
false
false
true
```

Isso retorna `true` para o último conjunto de valores porque o valor da coluna a (4) é maior que o valor da coluna b (2).

6. Opcionalmente, você pode consultar a visão materializada que faz referência à UDF em SQL. Os resultados da função SQL correspondem aos resultados da versão em Python.

```
SELECT * FROM mv_sql_udf ORDER BY a;
```

Os resultados da consulta são os seguintes:

```
a
-----
false
false
true
```

Isso retorna `true` para o último conjunto de valores a ser comparado.

7. Use uma instrução DROP com CASCADE para descartar a função definida pelo usuário e a visão materializada que faz referência a ela.

```
DROP FUNCTION udf_python_bool(int, int) CASCADE;
```

```
DROP FUNCTION udf_sql_bool(int, int) CASCADE;
```

Ingestão de streaming para uma visão materializada

Este tópico descreve como usar visões materializadas para acessar dados de streaming rapidamente.

A ingestão de streaming fornece ingestão de dados de baixa latência e alta velocidade do Amazon Kinesis Data Streams e do Amazon Managed Streaming para Apache Kafka em banco de dados provisionado pelo Amazon Redshift ou do Amazon Redshift sem servidor. Os dados chegam a uma visão materializada do Redshift que é configurada para essa finalidade. Isso ocasiona acesso rápido a dados externos. A ingestão de streaming reduz o tempo de acesso aos dados e o custo de armazenamento. É possível configurá-la para o cluster do Amazon Redshift ou para o grupo de trabalho do Amazon Redshift sem servidor, usando um pequeno grupo de comandos SQL. Depois de configurada, cada atualização de visão materializada pode ingerir centenas de megabytes de dados por segundo.

Como os dados fluem de um serviço de streaming para o Redshift

Isso ajuda a entender como funciona a ingestão de streaming e os objetos de banco de dados utilizados no processo. Os dados fluem diretamente de um provedor de fluxo de dados para um cluster provisionado pelo Amazon Redshift ou um grupo de trabalho do Amazon Redshift sem servidor. Não há uma área de pouso temporária, como o bucket do Amazon S3. O grupo de trabalho ou o cluster provisionado é o consumidor de fluxos. No banco de dados do Redshift, os dados lidos do fluxo chegam a uma visão materializada. Os dados são processados assim que chegam. Por exemplo, os valores JSON podem ser consumidos e associados às colunas de dados de uma visão materializada usando SQL. Quando a visão materializada é atualizada, o Redshift consome dados dos fragmentos de dados alocados do Kinesis ou das partições do Kafka até que a visão seja atualizada com o fluxo.

Casos de uso para a ingestão de streaming do Amazon Redshift envolvem dados que são gerados continuamente e que precisam ser processados em um curto período ou latência desde sua geração. Isso é chamado de analytics quase em tempo real. As fontes podem incluir dispositivos de TI, dispositivos de telemetria de sistemas e dados de fluxo de cliques de um site ou uma aplicação movimentada.

Práticas recomendadas de análise de dados para melhorar a performance

Quando você configura a ingestão de streaming, há opções de como analisar os dados recebidos. As práticas podem incluir a execução da lógica comercial ou a formatação à medida que os dados

chegam. Recomendamos as práticas recomendadas a seguir para evitar erros ou perda de dados. Elas são derivadas de testes internos e ajudam os clientes a solucionar problemas de configuração e de análise.

- Extração de valores de dados transmitidos: se você usar a função JSON_EXTRACT_PATH_TEXT na definição de visão materializada para fragmentar o JSON de streaming, isso poderá afetar significativamente a performance e a latência. Para explicar, para cada coluna extraída usando JSON_EXTRACT_PATH_TEXT, o JSON de entrada é analisado novamente. Depois disso, ocorrem a conversão do tipo de dados, a filtragem e os cálculos de lógica de negócios. Isso significa, por exemplo, que se você extrair dez colunas dos dados JSON, cada registro JSON será analisado dez vezes, o que inclui lógica adicional. Isso ocasiona maior latência de ingestão. Uma abordagem alternativa que recomendamos é usar a função JSON_PARSE para converter registros JSON no tipo de dados SUPER do Redshift. Depois que os dados transmitidos chegarem à visão materializada, use o PartiQL para extrair strings individuais da representação SUPER dos dados JSON. Para ter mais informações, confira Consultar dados semiestruturados.

  Além disso, observe que JSON_EXTRACT_PATH_TEXT tem um tamanho máximo de dados de 16 MB. Portanto, se algum registro JSON for maior que 16 MB, processá-lo com JSON_EXTRACT_PATH_TEXT vai gerar um erro.
- Associação de um fluxo do Amazon Kinesis Data Streams ou de um tópico do Amazon MSK a visões materializadas: não recomendamos criar várias visões materializadas para ingestão de dados de um único fluxo ou tópico. Isso ocorre porque cada visão materializada cria um consumidor para cada fragmento no fluxo do Kinesis Data Streams ou no tópico do Kafka. Isso pode ocasionar controle de utilização ou throughput excessivo do fluxo ou do tópico. Também pode gerar custos mais altos, já que você ingere os mesmos dados várias vezes. Ao configurar a ingestão de streaming, recomendamos criar uma visão materializada para cada fluxo ou tópico.

  Se o caso de uso exigir que você ingira os dados de um fluxo do KDS ou do tópico do MSK em várias visões materializadas, antes de fazer isso, consulte o AWS Big Data Blog, especificamente Best practices to implement near-real-time analytics using Amazon Redshift Streaming Ingestion with Amazon MSK.

Comportamento de ingestão de streaming e tipos de dados

A tabela a seguir descreve detalhes do comportamento técnico e limites de tamanho para vários tipos de dados. Recomendamos que você se familiarize com eles antes de configurar uma visão materializada para ingestão de streaming.

Recurso ou comportamento	Descrição
Limite de tamanho de tópicos do Kafka	Não é possível usar um tópico do Kafka com um nome maior que 128 caracteres (sem incluir aspas). Para obter mais informações, consulte Nomes e identificadores.
Atualizações e JOINs incrementais em uma visão materializada	A exibição materializada deve poder ser mantida de forma incremental. Não é possível realizar recálculo completo para o Kinesis ou Amazon MSK porque eles não preservam o histórico de fluxos ou tópicos após 24 horas ou 7 dias, por padrão. Você pode definir períodos de retenção de dados mais longos no Kinesis ou no Amazon MSK. No entanto, isso pode resultar em mais manutenção e custo. Além disso, atualmente não há suporte para JOINs em visões materializadas criadas em um fluxo do Kinesis ou em um tópico do Amazon MSK. Depois de criar uma visão materializada em seu fluxo ou tópico, você pode criar outra visão materializada para unir a visão materializada de streaming a outras visões materializadas, tabelas ou visões. Para obter mais informações, consulte ATUALIZAR EXIBIÇÃO MATERIALIZADA.
Análise de registros	A ingestão de streaming do Amazon Redshift não oferece suporte à análise de registros que foram agregados pela Kinesis Producer Library (Conceitos chave da KPL - Agregação). Os registros agregados são ingeridos, mas são armazenados como dados de buffer de protocolo binário. (Consulte Buffers de protocolo para obter mais informações.) Dependendo de como você envia dados para o Kinesis, talvez seja necessário desativar esse recurso.
Valores duplicados em cabeçalhos do Kafka	O cliente consumidor de streaming do Amazon Redshift para tópicos do Kafka provenientes do Amazon MSK, Confluent ou Apache Kafka não aceita valores duplicados no cabeçalho de tópicos do Kafka.
Descompressão	`VARBYTE` não comporta descompactação. Por causa disso, registros contendo dados compactados não podem ser

Recurso ou comportamento	Descrição
	consultados no Redshift. Descompacte seus dados antes de adicioná-los ao fluxo do Kinesis ou ao tópico do Amazon MSK.
Tamanho máximo do registro	O tamanho máximo de qualquer registro que o Amazon Redshift consegue ingerir de um serviço de streaming é 16.777.216 bytes (16 MiB), o tamanho máximo compatível com o tipo de dados VARBYTE no Amazon Redshift. As visualizações materializadas de streaming criadas em um tópico do Amazon MSK ou em um fluxo de dados do Kinesis definem o tamanho da coluna de dados VARBYTE como 16 MiB e podem consumir registros de até esse tamanho. No entanto, ao consumir de um fluxo de dados do Kinesis, o tamanho máximo de registro atualmente aceito pelo Kinesis é de 10.485.760 bytes (10 MiB). O suporte ao limite de tamanho de registro de 10 MiB do Kinesis exige o patch 203 ou posterior. As visualizações materializadas criadas em um fluxo de dados do Kinesis em um patch anterior têm a coluna de dados VARBYTE definida como 1.048.576 bytes (1 MiB) e continuam ingerindo apenas registros de até 1 MiB. Para ingerir registros maiores, recrie a visualização materializada no patch 203 ou posterior. Consulte mais informações sobre os limites de tamanho do Kinesis em Cotas e limites e em Como lidar com registros grandes no Guia do desenvolvedor do Amazon Kinesis Data Streams.

Recurso ou comportamento	Descrição
Registros de erros	Em cada caso em que um registro não pode ser ingerido no Redshift porque o tamanho dos dados excede o tamanho máximo, esse registro é ignorado. A atualização da visão materializada ainda é bem-sucedida, nesse caso, e um segmento de cada registro de erro é gravado na tabela do sistema SYS_STREAM_SCAN_ERRORS. Erros resultantes da lógica comercial, como um erro em um cálculo ou um erro resultante de uma conversão de tipo, não são ignorados. Teste a lógica com cuidado antes de adicioná-la à definição de visão materializada.
Amazon MSK Multi-VPC private connectivity	No momento, a conectividade privada de várias VPCs do Amazon MSK não é compatível com a ingestão de streaming do Redshift. Uma alternativa é usar o emparelhamento de VPC para conectar VPCs ou o AWS Transit Gateway para conectar VPCs e redes on-premises em um hub central. Qualquer um deles permite que o Redshift se comunique com um cluster do Amazon MSK ou com o Amazon MSK Sem Servidor em outra VPC.
Uso e ativação da atualização automática	As consultas de atualização automática para uma visão materializada são tratadas como qualquer outra workload do usuário. A atualização automática carrega os dados do fluxo assim que eles chegam. A atualização automática pode ser ativada explicitamente para uma visão materializada criada para ingestão de streaming. Para fazer isso, especifique `AUTO REFRESH` na definição da visão materializada. A atualização manual é o padrão. Para especificar a atualização automática de uma visão materiali zada existente para ingestão de streaming, você pode executar `ALTER MATERIALIZED VIEW` para ativá-la. Para obter mais informações, consulte CREATE MATERIALIZED VIEW ou ALTER MATERIALIZED VIEW.

Recurso ou comportamento	Descrição
Ingestão de streaming e Amazon Redshift sem servidor	As instruções de configuração que se aplicam à ingestão de streaming do Amazon Redshift em um cluster provisionado também se aplicam à ingestão de streaming no Amazon Redshift sem servidor. É importante especificar o nível necessário de RPUs para comportar a ingestão de streaming com atualização automática e outras workloads. Para obter mais informações, consulte Faturamento do Amazon Redshift Serverless.
Nós do Amazon Redshift em uma zona de disponibilidade diferente do cluster do Amazon MSK	Ao configurar a ingestão de streaming, o Amazon Redshift tenta se conectar a um cluster do Amazon MSK na mesma zona de disponibilidade, caso o reconhecimento de rack esteja habilitado para o Amazon MSK. Se todos os seus nós estiverem em zonas de disponibilidade diferentes do cluster do Amazon Redshift, você poderá ter custos de transferência de dados entre zonas de disponibilidade. Para evitar isso, mantenha pelo menos um nó de cluster de agente do Amazon MSK na mesma AZ do cluster ou grupo de trabalho provisionado do Redshift.
Atualizar local de início	Depois de criar uma visão materializada, sua atualização inicial começa em `TRIM_HORIZON` de um fluxo do Kinesis ou do desvio 0 de um tópico do Amazon MSK.
Formatos de dados	Os formatos de dados aceitos são limitados àqueles que podem ser convertidos de `VARBYTE`. Para obter mais informações, consulte Tipo VARBYTE e Operadores VARBYTE.

Recurso ou comportamento	Descrição
Anexar registros a uma tabela	É possível executar `ALTER TABLE APPEND` para anexar linhas a uma tabela de destino de uma visão materializada da origem existente. Isso funciona somente se a visão materializada estiver configurada para ingestão de streaming. Para obter mais informações, consulte ALTER TABLE APPEND. As operações `ALTER TABLE APPEND` mantêm bloqueios exclusivos quando executadas em visões materializadas de streaming do Amazon Redshift conectadas a qualquer um dos seguintes: • Um Amazon Kinesis data stream • Um tópico Amazon Managed Streaming for Apache Kafka • Um stream externo compatível, como um tópico do Confluent Cloud Kafka

Recurso ou comportamento	Descrição
Executar TRUNCATE ou DELETE	É possível remover registros de uma visão materializada que é usada para a ingestão de streaming usando o seguinte: • `TRUNCATE`: exclui todas as linhas de uma visão materializada configurada para ingestão de streaming. Ele não verifica a tabela. Para obter mais informações, consulte TRUNCATE. • `DELETE`: exclui todas as linhas de uma visão materializada configurada para ingestão de streaming. Para obter mais informações, consulte DELETE. As operações `TRUNCATE` e `DELETE` mantêm bloqueios exclusivos quando executadas em visões materializadas de streaming do Amazon Redshift conectadas a qualquer um dos seguintes: • Um Amazon Kinesis data stream • Um tópico Amazon Managed Streaming for Apache Kafka • Um stream externo compatível, como um tópico do Confluent Cloud Kafka

Recurso ou comportamento	Descrição
Identificadores não grafados em minúsculas	Ao criar visões materializadas de streaming em tópicos do Amazon Managed Streaming for Apache Kafka ou Kinesis Data Streams que contêm identificadores não grafados em minúscula s, as atualizações automáticas podem falhar. Para resolver isso, execute um dos seguintes procedimentos: • Use atualizações manuais com o parâmetro `enable_ca se_sensitive_identifier` definido como `true`. • Habilite as atualizações automáticas configurando `enable_case_sensitive_identifier` como `true` no nível do banco de dados ou do cluster. **Note** A configuração de `enable_case_sensit ive_identifier` no nível do usuário não é suficient e para atualizações automáticas, mas funcionará para atualizações manuais. Para ter mais informações sobre identificadores que diferenciam maiúsculas e minúsculas, consulte enable_case_sensitive_ident ifier.
Idempotência	O Amazon Redshift garante que cada registro seja processad o exatamente uma vez ao ingerir dados de fontes de streaming . Essa garantia se aplica a dois tipos de fontes: Amazon Kinesis (usando identificadores de stream, fragmento e número de sequência) e Apache Kafka (usando identificadores de tópico, partição e deslocamento), incluindo Amazon Managed Streaming for Apache Kafka (Amazon MSK) e Confluent Cloud.

Conceitos básicos da ingestão de streaming do Amazon Kinesis Data Streams

Este tópico descreve como consumir dados de streaming do Kinesis Data Streams usando uma visão materializada.

Configurar a ingestão de transmissão do Amazon Redshift envolve a criação de um esquema externo que mapeia para a fonte de dados de transmissão e a criação de uma exibição materializada que faça referência ao esquema externo. A ingestão de transmissão do Amazon Redshift oferece suporte ao Kinesis Data Streams como uma fonte. Como tal, você precisa ter uma fonte do Kinesis Data Streams disponível antes de configurar a ingestão de streaming. Se você não tiver uma fonte, siga as instruções na documentação do Kinesis em Conceitos básicos do Amazon Kinesis Data Streams ou crie uma no console usando as instruções em Criar um fluxo por meio do Console de Gerenciamento da AWS.

A ingestão de transmissão do Amazon Redshift usa uma exibição materializada, que é atualizada diretamente da transmissão quando `REFRESH` é executado. A exibição materializada mapeia para a fonte de dados da transmissão. Você pode executar filtragem e agregações nos dados da transmissão como parte da definição de exibição materializada. Sua exibição materializada de ingestão de transmissão (a exibição materializada de base) pode fazer referência a apenas uma transmissão, mas você pode criar exibições materializadas adicionais que se unam à exibição materializada de base e com outras exibição ou tabelas materializadas.

> **Note**
>
> Ingestão de streaming e Amazon Redshift Serverless: as etapas de configuração neste tópico se aplicam a clusters provisionados do Amazon Redshift e ao Amazon Redshift Serverless. Para obter mais informações, consulte Comportamento de ingestão de streaming e tipos de dados.

Supondo que você tenha uma transmissão do Kinesis Data Streams disponível, a primeira etapa é definir um esquema no Amazon Redshift com `CREATE EXTERNAL SCHEMA` e fazer referência a um recurso do Kinesis Data Streams. Depois disso, para acessar dados na transmissão, defina a `STREAM` em uma exibição materializada. Você pode armazenar registros da transmissão no formato semiestruturado `SUPER`, ou defina um esquema que resulte em dados convertidos em tipos de dados do Redshift. Quando você consulta a exibição materializada, os registros retornados são uma exibição pontual da transmissão.

1. Crie um perfil do IAM com uma política de confiança que permita que o cluster do Amazon Redshift ou o grupo de trabalho do Amazon Redshift sem servidor assuma o perfil. Para obter informações sobre como configurar a política de confiança da função do IAM, consulte Autorização do Amazon Redshift para acessar outros serviços da AWS em seu nome. Após sua criação, a função deve ter a seguinte política do IAM, que forneça permissão para comunicação com o fluxo de dados do Amazon Kinesis.

   Política do IAM para um fluxo não criptografado do Kinesis Data Streams

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ReadStream",
            "Effect": "Allow",
            "Action": [
                "kinesis:DescribeStreamSummary",
                "kinesis:GetShardIterator",
                "kinesis:GetRecords",
                "kinesis:ListShards",
                "kinesis:DescribeStream"
            ],
            "Resource": "arn:aws:kinesis:*:111122223333:stream/*"
        },
        {
            "Sid": "ListStream",
            "Effect": "Allow",
            "Action": "kinesis:ListStreams",
            "Resource": "*"
        }
    ]
}
```

   Política do IAM para um fluxo criptografado do Kinesis Data Streams

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ReadStream",
            "Effect": "Allow",
            "Action": [
                "kinesis:DescribeStreamSummary",
                "kinesis:GetShardIterator",
                "kinesis:GetRecords",
                "kinesis:ListShards",
                "kinesis:DescribeStream"
            ],
            "Resource": "arn:aws:kinesis:*:111122223333:stream/*"
        },
        {
            "Sid": "DecryptStream",
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        },
        {
            "Sid": "ListStream",
            "Effect": "Allow",
            "Action": "kinesis:ListStreams",
            "Resource": "*"
        }
    ]
}
```

2. Confira a VPC e verifique se o cluster do Amazon Redshift ou do Amazon Redshift sem servidor tem uma rota para chegar aos endpoints do Kinesis Data Streams pela internet usando um gateway NAT ou gateway da Internet. Se você quiser que o tráfego entre o Redshift e o Kinesis Data Streams permaneça na rede da AWS, considere usar um endpoint da VPC de interface do Kinesis. Para obter mais informações, consulte Usar o Amazon Kinesis Data Streams com endpoints da VPC de interface.

3. No Amazon Redshift, crie um esquema externo para mapear os dados do Kinesis para um esquema.

```
CREATE EXTERNAL SCHEMA kds
FROM KINESIS
IAM_ROLE { default | 'iam-role-arn' };
```

A ingestão de streaming para o Kinesis Data Streams não exige um tipo de autenticação. Ele usa o perfil do IAM definido na instrução `CREATE EXTERNAL SCHEMA` para fazer solicitações do Kinesis Data Streams.

Opcional: use a palavra-chave REGION para especificar a região onde o Amazon Kinesis Data Streams ou o fluxo do Amazon MSK está localizado.

```
CREATE EXTERNAL SCHEMA kds
FROM KINESIS
REGION 'us-west-2'
IAM_ROLE { default | 'iam-role-arn' };
```

Neste exemplo, a região especifica o local do fluxo de origem. IAM_ROLE é um exemplo.

4. Crie uma exibição materializada para consumir os dados da transmissão. Com uma declaração como a seguinte, se não for possível analisar um registro, será gerado um erro. Use um comando como esse se você não quiser que os registros de erro sejam ignorados.

```
CREATE MATERIALIZED VIEW my_view AUTO REFRESH YES AS
SELECT *
FROM kds.my_stream_name;
```

Nomes de transmissão do Kinesis fazem distinção entre maiúsculas e minúsculas e podem conter tanto maiúsculas quanto minúsculas. Para ingerir fluxos com nomes em letras maiúsculas, é possível definir a configuração `enable_case_sensitive_identifier` como `true` no nível do banco de dados. Para obter mais informações, consulte Nomes e identificadores e enable_case_sensitive_identifier.

Para ativar a atualização automática, use `AUTO REFRESH YES`. O comportamento padrão é atualização manual. Observe que, ao usar CAN_JSON_PARSE, é possível que os registros que não podem ser analisados sejam ignorados.

As colunas de metadados incluem o seguinte:

Coluna de metadados	Tipo de dados	Descrição
approximate_arrival_timestamp	time stamp sem fuso horário	A hora aproximada em que o registro foi inserido no fluxo do Kinesis
partition_key	varchar(256)	A chave usada pelo Kinesis para atribuir o registro a um fragmento
shard_id	char(20)	O identificador exclusivo do fragmento dentro do fluxo do qual o registro foi recuperado
sequence_number	varchar(128)	O identificador exclusivo do registro do fragmento do Kinesis
refresh_time	time stamp sem fuso horário	A hora de início da atualização
kinesis_data	varbyte	O registro do fluxo do Kinesis

É importante observar que, se você tiver uma lógica de negócios na definição de visão materializada, erros nesse lógica poderão causar bloqueios na ingestão de streaming em alguns casos. Isso pode fazer com que você precise descartar e recriar a visão materializada. Para evitar isso, recomendamos manter a lógica o mais simples possível e realizar a maioria das verificações de lógica de negócios nos dados após a ingestão.

5. Atualize a visão, o que invoca o Redshift para ler pelo fluxo e carregar dados na visão materializada.

```
REFRESH MATERIALIZED VIEW my_view;
```

6. Consulte dados na exibição materializada.

```
select * from my_view;
```

Conceitos básicos da ingestão de streaming de origens do Apache Kafka

Este tópico descreve como consumir dados de streaming do Amazon MSK, Apache Kafka ou Confluent Cloud usando uma visão materializada.

O objetivo da ingestão de streaming do Amazon Redshift é simplificar o processo de ingestão direta de dados de fluxo de um serviço de streaming no Amazon Redshift ou no Amazon Redshift sem servidor. Isso funciona com o Amazon MSK Provisioned e o Amazon MSK sem servidor, com o Apache Kafka de código aberto e com o Confluent Cloud. A ingestão de streaming do Amazon Redshift elimina a necessidade de preparar um tópico do Apache Kafka no Amazon S3 antes de preparar os dados de streaming no Redshift.

Em um nível técnico, a ingestão de streaming fornece ingestão de dados de fluxo ou tópico de baixa latência e alta velocidade em uma visão materializada do Amazon Redshift. Após a configuração, usando a atualização de visão materializada, você pode absorver grandes volumes de dados.

Você deve ter uma origem do Apache Kafka disponível para configurar a ingestão de streaming do Amazon Redshift. Caso você não tenha uma origem, crie uma usando as seguintes instruções:

- Amazon MSK: Getting Started Using Amazon MSK
- Apache Kafka: Apache Kafka Quickstart
- Confluent Cloud: Quick Start for Confluent Cloud

Configurar a ingestão de streaming por meio do Kafka

Use os procedimentos a seguir para configurar a ingestão de streaming para o Amazon Redshift usando origens do Amazon MSK ou do Apache Kafka que não são gerenciadas pela AWS (Apache Kafka e Confluent Cloud).

Tópicos

- Configurar a autenticação
- Configurar a VPC
- Criar uma visão materializada

Configurar a autenticação

Esta seção descreve a configuração da autenticação para permitir que aplicação do Amazon Redshift acesse uma origem do Amazon MSK.

Depois de criar o perfil da aplicação, anexe uma das políticas a seguir para permitir o acesso ao cluster do Amazon MSK, do Apache Kafka ou do Confluent Cloud. No caso da autenticação mTLS, você pode armazenar os certificados que o Amazon Redshift usa no ACM ou no Secrets Manager; portanto, você deve escolher a política que corresponde ao local onde o certificado está armazenado.

Observe que não é possível usar certificados autoassinados com autenticação ou dados em trânsito quando você usa a ingestão direta de streaming no Amazon Redshift com qualquer uma das fontes de streaming compatíveis do Apache Kafka. Isso inclui o Amazon MSK, o Apache Kafka e o Confluent Cloud. Considere a possibilidade de usar certificados gerados com o Gerenciador de certificados da AWS ou qualquer outra autoridade de certificação pública confiável.

É possível usar a autenticação do IAM para o Amazon Redshift com o MSK somente com o Kafka versão 2.7.1 ou posterior.

AUTHENTICATION IAM (somente Amazon MSK):

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "MSKIAMpolicy",
            "Effect": "Allow",
            "Action": [
                "kafka-cluster:ReadData",
                "kafka-cluster:DescribeTopic",
                "kafka-cluster:Connect"
            ],
            "Resource": [
                "arn:aws:kafka:*:111122223333:cluster/MyTestCluster/*",
                "arn:aws:kafka:*:111122223333:topic/MyTestCluster/*"
            ]
        },
        {
```

```
            "Effect": "Allow",
            "Action": [
                "kafka-cluster:AlterGroup",
                "kafka-cluster:DescribeGroup"
            ],
            "Resource": [
                "arn:aws:kafka:*:111122223333:group/MyTestCluster/*"
            ]
        }
    ]
}
```

AUTHENTICATION MTLS: usando um certificado armazenado em Gerenciador de certificados da AWS

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "MSKmTLSACMpolicy",
            "Effect": "Allow",
            "Action": [
                "acm:ExportCertificate"
            ],
            "Resource": [
                "arn:aws:acm:us-east-1:444455556666:certificate/certificate_ID"
            ]
        }
    ]
}
```

AUTHENTICATION MTLS: usando um certificado armazenado em AWS Secrets Manager

JSON

```
{
    "Version":"2012-10-17",
```

```
    "Statement": [
        {
            "Sid": "MSKmTLSSecretsManagerpolicy",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [
                "arn:aws:secretsmanager:us-east-1:444455556666:secret:secret_ID"
            ]
        }
    ]
}
```

Amazon MSK

Se você usar AUTHENTICATION NONE para se conectar a uma origem do Amazon MSK, nenhum perfil do IAM será necessário. No entanto, se você usar AUTHENTICATION IAM ou mTLS para se autenticar em seu cluster do Amazon MSK, seu cluster do Amazon Redshift ou namespace do Amazon Redshift sem servidor deverá ter um perfil do IAM anexado com as permissões apropriadas. Crie um perfil do IAM com uma política de confiança que permita que o cluster do Amazon Redshift ou o namespace do Amazon Redshift sem servidor assuma o perfil. Depois de criar o perfil, adicione uma das permissões a seguir para oferecer suporte ao IAM ou ao mTLS. No caso da autenticação mTLS, os certificados que o Amazon Redshift usa podem ser armazenados no Gerenciador de certificados da AWS ou no AWS Secrets Manager; portanto, você deve escolher a política que corresponda ao local em que o certificado está armazenado. Anexe o perfil ao cluster provisionado pelo Amazon Redshift ou ao namespace do Redshift sem servidor. Para obter informações sobre como configurar a política de confiança da função do IAM, consulte Autorização do Amazon Redshift para acessar outros serviços da AWS em seu nome.

A seguinte tabela mostra as opções de configuração complementares a serem definidas para a ingestão de streaming do Amazon MSK:

Configuração do Amazon Redshift	Configuração do Amazon MSK	Porta a ser aberta entre o Redshift e o Amazon MSK
AUTHENTICATION NONE	Transporte TLS desabilitado	9092
AUTHENTICATION NONE	Transporte TLS habilitado	9094

Configuração do Amazon Redshift	Configuração do Amazon MSK	Porta a ser aberta entre o Redshift e o Amazon MSK
AUTHENTICATION IAM	IAM	9098/9198
AUTHENTICATION MTLS	Transporte TLS habilitado	9094

A autenticação do Amazon Redshift é definida na instrução CREATE EXTERNAL SCHEMA.

> **Note**
>
> Quando o cluster do Amazon MSK tiver a autenticação Mutual Transport Layer Security (mTLS) habilitada, a configuração do Amazon Redshift para usar AUTHENTICATION NONE o direcionará a usar a porta 9094 para acesso não autenticado. No entanto, isso apresentará falha, já que a porta está sendo usada pela autenticação mTLS. Por isso, recomendamos que você mude para AUTHENTICATION IAM ao usar mTLS.

Apache Kafka or Confluent Cloud

Para Apache Kafka e Confluent Cloud, o Amazon Redshift é com os seguintes protocolos de conexão:

- Você pode usar mTLS ou texto simples com transporte TLS para autenticação ao se conectar ao Apache Kafka.
- Você só pode usar mTLS para autenticação ao se conectar ao Confluent Cloud.

O Amazon Redshift é compatível com os seguintes protocolos de criptografia para conexão ao Apache Kafka ou ao Confluent Cloud:

Métodos de autenticação compatíveis do Apache Kafka e Confluent Cloud

Amazon Redshift	Protocolo de segurança do Kafka	Suporte do Apache Kafka	Suporte do Confluent Cloud
AUTHENTICATION NONE	PLAINTEXT	Não	Não

Amazon Redshift	Protocolo de segurança do Kafka	Suporte do Apache Kafka	Suporte do Confluent Cloud
AUTHENTICATION NONE	SSL	Sim	Não
AUTHENTICATION IAM	SASL_SSL	Não	Não
AUTHENTICATION MTLS	SSL	Sim (com certificado)	Sim (com certificado)

O Amazon Redshift não é compatível com SASL/SCRAM ou SASL/PLAINTEXT.

Configurar a VPC

Depois de criar os recursos de autenticação, verifique a VPC e se o cluster do Amazon Redshift ou grupo de trabalho do Amazon Redshift sem servidor tem uma rota para chegar à origem do Apache Kafka.

> **Note**
>
> No caso do Amazon MSK, as regras do grupo de segurança de entrada para o cluster do Amazon MSK devem dar permissão ao grupo de segurança do cluster do Amazon Redshift ou ao grupo de trabalho do Redshift sem servidor. As portas que você especifica dependem dos métodos de autenticação usados em seu cluster, caso use o Amazon MSK. Para obter mais informações, consulte Informações sobre portas e Acesso de dentro da AWS, mas de fora da VPC.

Em seguida, habilite o roteamento de VPC avançado no cluster do Amazon Redshift ou no grupo de trabalho do Amazon Redshift sem servidor. Para obter mais informações, consulte Habilitar o roteamento aprimorado de VPC.

Criar uma visão materializada

Nesta seção, você configura a visão materializada que o Amazon Redshift usa para acessar os dados de streaming do Apache Kafka.

Supondo que você tenha um cluster do Apache Kafka disponível, o primeiro passo é definir um esquema no Redshift com `CREATE EXTERNAL SCHEMA` e fazer referência ao cluster como a fonte de dados. Depois disso, para acessar dados no tópico, defina `STREAM` em uma visão materializada. Você pode armazenar registros de seu tópico usando o tipo de dados VARBYTE padrão do Amazon Redshift ou definir um esquema que converta os dados no formato `SUPER` semiestruturado. Quando você consulta a visão materializada, os registros retornados são uma exibição pontual do tópico.

1. No Amazon Redshift, crie um esquema externo para associá-lo ao cluster do Apache Kafka. A sintaxe é a seguinte:

```
CREATE EXTERNAL SCHEMA MySchema
FROM KAFKA
[ IAM_ROLE [ default | 'iam-role-arn' ] ]
AUTHENTICATION [ none | iam | mtls ]
{AUTHENTICATION_ARN 'acm-certificate-arn' |  SECRET_ARN 'asm-secret-arn'};
```

   Na cláusula `FROM`, o `KAFKA` indica que o esquema mapeia dados de uma origem do Apache Kafka.

   `AUTHENTICATION` denota o tipo de autenticação para ingestão de streaming. Existem três modos disponíveis:

   - none: especifica que não há nenhuma etapa obrigatória. Isso corresponde ao acesso não autenticado no MSK. Isso corresponde à autenticação SSL no Apache Kafka. Esse método de autenticação não é compatível com o Confluent Cloud.
   - iam: especifica a autenticação do IAM. Você só pode usar a autenticação do IAM com o Amazon MSK. Ao escolher isso, o perfil do IAM tem permissões para autenticação do IAM. Para ter mais informações sobre como configurar as políticas do IAM necessárias, consulte Configurar a ingestão de streaming por meio do Kafka.
   - mtls: especifica que a segurança da camada de transporte mútuo oferece comunicação segura ao facilitar a autenticação entre um cliente e um servidor. Neste caso, o cliente é o Redshift e o servidor é o Apache Kafka. Para ter mais informações sobre como configurar a ingestão de streaming com mTLS, consulte Autenticação com mTLS para ingestão de streaming do Redshift por meio de origens do Apache Kafka.

   A autenticação do Amazon MSK com nome de usuário e senha não permite ingestão de streaming.

O parâmetro AUTHENTICATION_ARN especifica o ARN do certificado Transport Layer Security mútuo (mTLS) do ACM que você usa para estabelecer uma conexão criptografada.

O parâmetro SECRET_ARN especifica o ARN do segredo do AWS Secrets Manager que contém o certificado a ser usado pelo Amazon Redshift para mTLS.

Os seguintes exemplos mostram como definir o URI do agente para o cluster do Amazon MSK ao criar o esquema externo:

Usar a autenticação do IAM:

```
CREATE EXTERNAL SCHEMA my_schema
FROM KAFKA
IAM_ROLE 'arn:aws:iam::012345678901:role/my_role'
AUTHENTICATION IAM
URI 'b-1.myTestCluster.123z8u.c2.kafka.us-
west-1.amazonaws.com:9098,b-2.myTestCluster.123z8u.c2.kafka.us-
west-1.amazonaws.com:9098'
```

Sem usar autenticação:

```
CREATE EXTERNAL SCHEMA my_schema
FROM KAFKA
AUTHENTICATION none
URI 'b-1.myTestCluster.123z8u.c2.kafka.us-
west-1.amazonaws.com:9092,b-2.myTestCluster.123z8u.c2.kafka.us-
west-1.amazonaws.com:9092'
```

Usar o mTLS:

```
CREATE EXTERNAL SCHEMA my_schema
FROM KAFKA
IAM_ROLE 'arn:aws:iam::012345678901:role/my_role'
AUTHENTICATION MTLS
URI 'b-1.myTestCluster.123z8u.c2.kafka.us-west-1.amazonaws.com:9094,b-
 2.myTestCluster.123z8u.c2.kafka.us-west-1.amazonaws.com:9094'
{AUTHENTICATION_ARN 'acm-certificate-arn' |  SECRET_ARN 'asm-secret-arn'}
```

Consulte mais informações sobre como criar um esquema externo em CREATE EXTERNAL SCHEMA.

2. Crie uma visão materializada para consumir os dados do tópico. Use um comando SQL como o exemplo a seguir.

```
CREATE MATERIALIZED VIEW MyView AUTO REFRESH YES AS
SELECT *
FROM MySchema."mytopic";
```

Nomes de tópico do Kafka diferenciam letras maiúsculas e minúsculas, podendo conter ambas. Para ingerir tópicos com nomes em letras maiúsculas, é possível definir a configuração `enable_case_sensitive_identifier` como `true` no nível da sessão ou do banco de dados. Para obter mais informações, consulte Nomes e identificadores e enable_case_sensitive_identifier.

Para ativar a atualização automática, use `AUTO REFRESH YES`. O comportamento padrão é atualização manual.

3. As colunas de metadados incluem o seguinte:

Coluna de metadados	Tipo de dados	Descrição
kafka_partition	bigint	ID de partição do registro do tópico do Kafka
kafka_offset	bigint	Deslocamento do registro no tópico do Kafka para determinada partição
kafka_timestamp_type	char(1)	Tipo de time stamp usado no registro do Kafka: • C: hora de criação do registro (CREATE_TIME) no lado do cliente • L: hora de inclusão do registro (LOG_APPEND_TIME) no lado do servidor do Kafka

Coluna de metadados	Tipo de dados	Descrição
		• U: a hora de criação do registro não está disponível (NO_TIMESTAMP_TYPE)
kafka_timestamp	time stamp sem fuso horário	O valor do time stamp para o registro
kafka_key	varbyte	A chave do registro do Kafka
kafka_value	varbyte	O registro recebido do Kafka
kafka_headers	super	O cabeçalho do registro recebido do Kafka
refresh_time	time stamp sem fuso horário	A hora de início da atualização

É importante observar que, se você tiver uma lógica de negócios na definição de visão materializada, isso pode gerar falhas na ingestão de streaming em alguns casos. Isso pode fazer com que você precise descartar e recriar a visão materializada. Para evitar isso, recomendamos manter a simplicidade da lógica de negócios e executar lógica adicional nos dados após a ingestão.

4. Atualize a visão, o que invoca o Amazon Redshift para ler o tópico e carregar dados na visão materializada.

```
REFRESH MATERIALIZED VIEW MyView;
```

5. Consulte dados na exibição materializada.

```
select * from MyView;
```

A visão materializada é atualizada diretamente do tópico quando REFRESH é executada. Você cria uma visão materializada que mapeia à fonte de dados do tópico do Kafka. Você pode executar filtragem e agregações nos dados como parte da definição de visão materializada. Sua visão materializada de ingestão de streaming (a visão materializada de base) pode fazer

referência a apenas um tópico do Kafka, mas você pode criar visões materializadas adicionais que se unam à visão materializada de base e com outras visões materializadas ou tabelas.

Para obter mais informações sobre as limitações da ingestão de streaming, consulte Comportamento de ingestão de streaming e tipos de dados.

Autenticação com mTLS para ingestão de streaming do Redshift por meio de origens do Apache Kafka

O Transport Layer Security mútuo (mTLS) fornece os meios para um servidor autenticar um cliente ao qual está enviando informações e para o cliente autenticar o servidor. A vantagem de usar o mTLS é fornecer autenticação confiável para uma variedade de casos de uso em várias aplicações verticais do setor. Isso inclui casos de uso nos setores financeiro, de varejo, governamental e de saúde. No caso de ingestão de streaming no Redshift, a autenticação ocorre entre um servidor, que pode ser Amazon MSK, Apache Kafka ou Confluent Cloud, e um cluster provisionado do Amazon Redshift ou um grupo de trabalho do Amazon Redshift sem servidor.

Este tópico apresenta procedimentos e exemplos de comandos SQL que mostram maneiras de criar um esquema externo usando mTLS para autenticação entre o cliente do Redshift e qualquer servidor do Apache Kafka. As etapas neste tópico complementam o conjunto completo de etapas para configurar a ingestão de streaming de origens do Apache Kafka. Para obter mais informações, consulte Conceitos básicos da ingestão de streaming de origens do Apache Kafka.

Pré-requisitos para usar mTLS para ingestão de streaming

Esta seção apresenta as etapas essenciais para usar mTLS em ingestão de streaming com o Gerenciador de certificados da AWS ou o AWS Secrets Manager.

Como etapa preliminar, você deve ter ou criar uma autoridade de certificação privada (PCA), que possa ser usada para emitir certificados que, entre outras funções, permitem a comunicação segura por meio de canais de comunicação seguros. O . Autoridade de certificação privada da AWS (CA privada) é um serviço disponível que executa essa função. Para ter mais informações, consulte Creating a private CA, no Guia do usuário do Autoridade de certificação privada da AWS. Depois de criar a CA privada, exporte o certificado da CA raiz e salve-o em um arquivo com extensão .pem.

Para criar um cluster que usa o certificado da CA, faça o seguinte:

Amazon MSK

1. Crie um cluster do Amazon MSK que ofereça suporte à autenticação de cliente mTLS. Para obter mais informações sobre como configurar um cluster do Amazon MSK, consulte Mutual TLS client authentication for Amazon MSK no Guia do desenvolvedor do Amazon Managed Streaming for Apache Kafka.
2. Edite as configurações de segurança do cluster do Amazon MSK, ativando a autenticação de cliente TLS por meio do Gerenciador de certificados da AWS (ACM) e selecionando o CA privada da AWS (PCA) que você criou anteriormente. Para ter mais informações, consulte Updating security settings of a cluster no Guia do desenvolvedor do Amazon Managed Streaming for Apache Kafka.

Confluent Cloud

1. Crie um cluster dedicado do Confluent Cloud, de preferência na mesma Região da AWS do cluster do Amazon Redshift. Para obter informações sobre como criar um cluster do Confluent Cloud, consulte Create a Kafka cluster in Confluent Cloud.
2. Carregue o arquivo pem do certificado da CA raiz CA privada da AWS exportado que você criou anteriormente. Para obter mais informações, consulte Manage certificate authority for mTLS authentication for Confluent Cloud. O Confluent Cloud usa esse certificado para verificar o certificado do cliente do Amazon Redshift.

Usar mTLS para ingestão de streaming com o Gerenciador de certificados da AWS

O seguinte procedimento mostra como configurar o mTLS para ingestão de streaming do Redshift usando o Gerenciador de certificados da AWS (ACM) para armazenamento e gerenciamento de certificados:

1. Solicite um certificado privado por meio do ACM. Ao fazer isso, selecione o PCA que você criou na seção “Pré-requisitos” como autoridade de certificação. O ACM armazena o certificado assinado e a chave privada anexada para comunicação segura. Para ter mais informações sobre gerenciamento de certificados com o ACM, consulte Issuing and managing certificates no Guia do usuário do Gerenciador de certificados da AWS.
2. Quanto ao perfil do IAM que você usa para gerenciar seu cluster do Redshift ou grupo de trabalho do Amazon Redshift sem servidor, anexe a permissão para exportar o certificado, que é acm:ExportCertificate. Para obter mais informações sobre como configurar os recursos do IAM

necessários para ingestão de streaming, consulte Configurar a ingestão de streaming por meio do Kafka. Especifique o mesmo perfil do IAM na próxima etapa para criar o esquema externo.

> **Note**
>
> Solicitações para o Gerenciador de certificados da AWS exigir um gateway da internet (IGW) ou um gateway NAT (NGW) em sua VPC. Se sua VPC não tiver um IGW nem um NGW, faça o seguinte:
>
> - Use o Secrets Manager em vez do ACM para armazenar seus certificados.
> - Anexe um endpoint da VPC do Secrets Manager à sua VPC.
>
> Para obter informações sobre como usar o Secrets Manager com mTLS para ingestão de streaming, consulte Usar mTLS para ingestão de streaming com o AWS Secrets Manager a seguir.

3. Obtenha o URI do agente de bootstrap para o cluster do Amazon MSK, do Apache Kafka ou do Confluent Cloud. Para obter informações sobre como obter o URI do agente de bootstrap para o Amazon MSK, consulte Getting the bootstrap brokers for an Amazon MSK cluster no Guia do desenvolvedor do Amazon Managed Streaming for Apache Kafka.
4. Execute um comando SQL como o exemplo a seguir para criar um esquema externo que associe o cluster a um esquema externo do Redshift usando `mtls`.

   Amazon MSK

   ```
   CREATE EXTERNAL SCHEMA my_schema
   FROM KAFKA
   IAM_ROLE 'arn:aws:iam::012345678901:role/my_role'
   AUTHENTICATION mtls
   URI 'b-1.myTestCluster.123z8u.c2.kafka.us-
   west-1.amazonaws.com:9094,b-2.myTestCluster.123z8u.c2.kafka.us-
   west-1.amazonaws.com:9094'
   AUTHENTICATION_ARN 'arn:aws:acm:Region:444455556666:certificate/certificate_ID';
   ```

   Apache Kafka or Confluent Cloud

   ```
   CREATE EXTERNAL SCHEMA my_schema
   FROM KAFKA
   IAM_ROLE 'arn:aws:iam::012345678901:role/my_role'
   ```

```
AUTHENTICATION mtls
URI 'lkc-2v531.domz6wj0p.us-west-1.aws.confluent.cloud:9092'
AUTHENTICATION_ARN 'arn:aws:acm:region:444455556666:certificate/certificate_ID';
```

Parâmetros importantes:

- IAM_ROLE: o perfil do IAM associado ao cluster para ingestão de streaming.
- URI: o URI do agente de bootstrap para o cluster. Observe que, no caso do Amazon MSK, a porta 9094 destina-se à comunicação com agentes para criptografia TLS.
- AUTHENTICATION_ARN: o ARN do certificado do ACM. O ARN está disponível no console do ACM quando você escolhe o certificado emitido.

Depois de executar essas etapas de configuração, você pode criar uma visão materializada do Redshift que faça referência ao esquema definido no exemplo e, em seguida, usar REFRESH MATERIALIZED VIEW para transmitir dados. Para obter mais informações, consulte Conceitos básicos da ingestão de streaming de origens do Apache Kafka.

Usar mTLS para ingestão de streaming com o AWS Secrets Manager

Você pode configurar o mTLS para ingestão de streaming do Redshift usando o AWS Secrets Manager para gerenciamento de certificados se não quiser fazer referência ao certificado no Gerenciador de certificados da AWS. As etapas a seguir descrevem como configurar o mTLS usando o Secrets Manager.

1. Crie uma solicitação de assinatura de certificado e uma chave privada com a ferramenta de sua preferência. Em seguida, você pode usar a solicitação de assinatura para gerar um certificado assinado, utilizando o mesmo AWS Private CA (PCA) usado para gerar o certificado para o cluster. Para ter mais informações sobre emissão de certificados, consulte IssueCertificate na Referência de API do Autoridade de certificação privada da AWS.
2. Extraia o certificado usando o Autoridade de certificação privada da AWS. Para obter mais informações, consulte Retrieve a private certificate no Guia do usuário do Autoridade de certificação privada da AWS.
3. Armazene o certificado e a chave privada gerados na etapa anterior no AWS Secrets Manager. Escolha `Other type of secret` e use o formato de texto simples. Os pares de chave-valor devem estar no formato `{"certificate":"<cert value>","privateKey":"<pkey value>"}`, como no exemplo a seguir. Para ter mais informações sobre como criar segredos

no AWS Secrets Manager, consulte Create and manage secrets with AWS Secrets Manager no Guia do usuário do AWS Secrets Manager.

```
{"certificate":"-----BEGIN CERTIFICATE-----
klhdslkfjahksgdfkgioeuyihbflahabhbdslv6akybeoiwv1hoaiusdhbahsbdi
H4hAX8/eE96qCcjkpfT84EdvHzp6fC+/WwM0oXlwUEWlvfMCXNaG5D8SqRq3qA==
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
klhdslkfjahksgdfkgioeuyihbflahabhbdslv6akybeoiwv1hoaiusdhbahsbdi
wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
-----END CERTIFICATE-----",
"privateKey":"-----BEGIN PRIVATE KEY-----
klhdslkfjahksgdfkgioeuyihbflahabhbdslv6akybeoiwv1hoaiusdhbahsbdi
7OD4m1dBEs3Fj++hDMH9rYRp99RqtCOf0EWOUe139KOilOsW+cyhAoc9Ci2+Jo/k
17u2N1iGILMQEZuCRtnJOkFYkw==
-----END PRIVATE KEY-----"}
```

4. Anexe a política de permissão de recuperação do segredo ao perfil do IAM que você usa para gerenciar o cluster do Amazon Redshift ou grupo de trabalho do Amazon Redshift sem servidor. Essa permissão é `secretsmanager:GetSecretValue`. Para obter mais informações, consulte Configurar a autenticação. Para obter mais informações sobre como gerenciar políticas do IAM, consulte Editar políticas do IAM. Especifique o mesmo perfil do IAM na próxima etapa para criar o esquema externo.
5. No Redshift, execute o comando SQL para criar o esquema externo. Use o tipo de AUTENTICAÇÃO `mtls`. Você também especifica o URI do cluster e o ARN secreto do AWS Secrets Manager.

```
CREATE EXTERNAL SCHEMA my_schema
FROM KAFKA
IAM_ROLE 'arn:aws:iam::012345678901:role/my_role'
AUTHENTICATION mtls
URI 'b-1.myTestCluster.123z8u.c2.kafka.us-
west-1.amazonaws.com:9094,b-2.myTestCluster.123z8u.c2.kafka.us-
west-1.amazonaws.com:9094'
SECRET_ARN 'arn:aws:secretsmanager:us-east-1:012345678910:secret:myMTLSSecret';
```

Parâmetros importantes:

- IAM_ROLE: o perfil do IAM associado ao cluster para ingestão de streaming.

- URI: o URI do agente de bootstrap para o cluster. Observe que, no caso do Amazon MSK, a porta 9094 destina-se à comunicação com agentes para criptografia TLS.
- SECRET_ARN: o ARN do segredo do Secrets Manager, que contém o certificado a ser usado para mTLS.

Habilitar a autenticação mTLS para um esquema externo existente

Se você tiver um esquema externo usado para ingestão de streaming e quiser implementar o TLS mútuo para autenticação, poderá executar um comando como o apresentado a seguir, que especifica a autenticação mTLS e o ARN do certificado do ACM no ACM.

```
ALTER EXTERNAL SCHEMA schema_name
AUTHENTICATION mtls
AUTHENTICATION_ARN 'arn:aws:acm:Region:444455556666:certificate/certificate_ID';
```

Ou você pode especificar a autenticação mTLS, com referência ao ARN do segredo no AWS Secrets Manager.

```
ALTER EXTERNAL SCHEMA schema_name
AUTHENTICATION mtls
SECRET_ARN 'arn:aws:secretsmanager:us-east-1:012345678910:secret:myMTLSSecret';
```

Para ter informações sobre o comando ALTER EXTERNAL SCHEMA, consulte ALTER EXTERNAL SCHEMA.

Ingestão de dados de streaming usando o Kinesis

Esse procedimento demonstra como ingerir dados de uma transmissão do Kinesis chamada ev_station_data, que contém dados de consumo de diferentes estações de carregamento EV, no formato JSON. O esquema está bem definido. O exemplo mostra como armazenar os dados como JSON bruto, e também como converter os dados JSON em tipos de dados do Amazon Redshift à medida que são ingeridos.

Configuração do produtor

1. Usando o Amazon Kinesis Data Streams, siga as etapas para criar uma transmissão chamada `ev_station_data`. Escolha On-demand (Sob demanda) para o Capacity mode (Modo de capacidade). Para obter mais informações, consulte Gerenciamento de transmissões via Console de Gerenciamento da AWS.

2. O Gerador de dados do Amazon Kinesis pode ajudá-lo a gerar dados de teste para uso com sua transmissão. Siga as etapas detalhadas na ferramenta para começar, e use o seguinte modelo de dados para gerar seus dados:

```
{

   "_id" : "{{random.uuid}}",
   "clusterID": "{{random.number(
        {   "min":1,
            "max":50
        }
    )}}",
    "connectionTime": "{{date.now("YYYY-MM-DD HH:mm:ss")}}",
    "kWhDelivered": "{{commerce.price}}",
    "stationID": "{{random.number(
        {   "min":1,
            "max":467
        }
    )}}",
      "spaceID": "{{random.word}}-{{random.number(
        {   "min":1,
            "max":20
        }
    )}}",

   "timezone": "America/Los_Angeles",
   "userID": "{{random.number(
        {   "min":1000,
            "max":500000
        }
    )}}"
}
```

Cada objeto JSON nos dados da transmissão tem as seguintes propriedades:

```
{
    "_id": "12084f2f-fc41-41fb-a218-8cc1ac6146eb",
    "clusterID": "49",
    "connectionTime": "2022-01-31 13:17:15",
    "kWhDelivered": "74.00",
    "stationID": "421",
    "spaceID": "technologies-2",
```

```
    "timezone": "America/Los_Angeles",
    "userID": "482329"
}
```

Configuração do Amazon Redshift

Essas etapas mostram como configurar a exibição materializada para ingerir dados.

1. Crie um esquema externo para mapear os dados do Kinesis para um objeto do Redshift.

   ```
   CREATE EXTERNAL SCHEMA evdata FROM KINESIS
   IAM_ROLE 'arn:aws:iam::0123456789:role/redshift-streaming-role';
   ```

   Para obter mais informações sobre como configurar a função do IAM, consulte Conceitos básicos da ingestão de streaming do Amazon Kinesis Data Streams.

2. Crie uma exibição materializada para consumir os dados da transmissão. O exemplo a seguir mostra como definir uma visão materializada para ingerir dados formatados em JSON de um fluxo do Kinesis.

   Primeiro, armazene registros da transmissão em formato SUPER semiestruturado. Neste exemplo, a origem JSON é armazenada no Redshift sem conversão para tipos do Redshift.

   ```
   CREATE MATERIALIZED VIEW ev_station_data AS
       SELECT approximate_arrival_timestamp,
       partition_key,
       shard_id,
       sequence_number,
       case when can_json_parse(kinesis_data) then json_parse(kinesis_data) else null
    end as payload,
       case when not can_json_parse(kinesis_data) then kinesis_data else null end as
    failed_payload
       FROM evdata."ev_station_data" ;
   ```

Consulte a transmissão

1. Habilite os atributos SUPER com distinção de maiúsculas e minúsculas usando o comando a seguir. Por padrão, o Amazon Redshift não diferencia maiúsculas de minúsculas. Portanto, para

acessar os atributos SUPER com distinção de maiúsculas e minúsculas, você precisa habilitar essa funcionalidade.

```
SET enable_case_sensitive_super_attribute to TRUE;
```

2. Atualize a visão materializada com o comando a seguir para extrair dados do fluxo.

```
REFRESH MATERIALIZED VIEW ev_station_data;
```

3. Consulte a exibição materializada atualizada para obter estatísticas de uso.

```
SELECT e.payload.connectionTime::date as connectiontime
,SUM(e.payload.kWhDelivered::decimal(10,2)) AS Energy_Consumed
,count(distinct e.payload.userID) AS #Users
from ev_station_data as e
group by connectiontime
order by 1 desc;
```

4. Exibir resultados.

```
connectiontime        energy_consumed    #users
2022-02-08                 4139          10
2022-02-09                 5571          10
2022-02-10                 8697          20
2022-02-11                 4408          10
2022-02-12                 4257          10
2022-02-23                 6861          10
```

AWS Glue Data CatalogVisualizações do

Este tópico descreve como criar visualizações no AWS Glue Data Catalog. Você pode usar visualizações no Catálogo de Dados para acessar dados em diferentes fontes de dados usando o mesmo esquema.

Ao criar visualizações no Catálogo de Dados, você pode criar um único esquema de visualização comum e um objeto de metadados a ser usado em vários mecanismos, como Amazon Athena e Amazon EMR Spark. Dessa forma, é possível usar as mesmas visualizações nos seus data lakes e data warehouses para atender aos seus casos de uso. As exibições no Data Catalog são especiais por serem categorizadas como exibições definidoras, nas quais as permissões de acesso são definidas pelo usuário que criou a exibição, e não pelo usuário que consulta a exibição. Estes são alguns casos de uso e benefícios da criação de exibições no Data Catalog:

- Crie uma exibição que restrinja o acesso aos dados com base nas permissões das quais o usuário precisa. Por exemplo, você pode usar as exibições do Data Catalog para evitar que funcionários que não trabalham no departamento de RH vejam informações de identificação pessoal (PII).
- Verifique se os usuários não conseguem acessar registros incompletos. Aplicando determinados filtros à exibição do Data Catalog, você garante que os registros de dados em uma exibição do Data Catalog estejam sempre completos.
- As exibições do Data Catalog têm um benefício de segurança incluído de garantir que a definição da consulta usada para criar a exibição seja concluída para criar a exibição. Esse benefício de segurança significa que as exibições no Data Catalog não são suscetíveis a comandos SQL de jogadores mal-intencionados.
- As exibições no Data Catalog oferecem as mesmas vantagens das exibições normais, como permitir a usuários acessar uma exibição sem disponibilizar a tabela subjacente para usuários.

Para criar uma exibição no Data Catalog, você deve ter uma tabela externa do Spectrum, um objeto contido em uma unidade de compartilhamento de dados gerenciada pelo Lake Formation, ou uma tabela do Apache Iceberg.

As definições de exibição do Data Catalog são armazenadas no AWS Glue Data Catalog. Use o AWS Lake Formation para conceder acesso por meio das concessões de recursos, concessões de colunas ou dos controles de acesso baseados em etiquetas. Para obter mais informações sobre como conceder e revogar acesso no Lake Formation, consulte Granting and revoking permissions on Data Catalog resources.

Quando você usa o Amazon Redshift para executar uma consulta que faz referência a essas visualizações do AWS Glue Data Catalog, o Amazon Redshift automaticamente mascara campos em determinadas colunas de tabela e visualização do sistema ao registrar metadados sobre essa consulta. Para obter mais informações, consulte Registros em log seguros no Guia de gerenciamento do Amazon Redshift.

Pré-requisitos

Para criar uma exibição no Data Catalog, verifique se você tem os seguintes pré-requisitos atendidos:

- Verifique se o perfil do IAM tem a política de confiança a seguir.

  JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
               "Service": [
                    "glue.amazonaws.com",
                    "lakeformation.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

- Você também precisa da politica da função da aprovação.

  JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt1",
            "Action": [
                "iam:PassRole"
```

```
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "glue.amazonaws.com",
                        "lakeformation.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

- Por fim, você também precisa das seguintes permissões:
  - `Glue:GetDatabase`
  - `Glue:GetDatabases`
  - `Glue:CreateTable`
  - `Glue:GetTable`
  - `Glue:UpdateTable`
  - `Glue:DeleteTable`
  - `Glue:GetTables`
  - `Glue:SearchTables`
  - `Glue:BatchGetPartition`
  - `Glue:GetPartitions`
  - `Glue:GetPartition`
  - `Glue:GetTableVersion`
  - `Glue:GetTableVersions`

Exemplo completo

Comece criando um esquema externo com base no banco de dados do Data Catalog.

```
CREATE EXTERNAL SCHEMA IF NOT EXISTS external_schema FROM DATA CATALOG DATABASE
  'external_data_catalog_db'
```

```
IAM_ROLE 'arn:aws:iam::123456789012:role/sample-role';
```

Você já pode criar uma exibição do Data Catalog.

```
CREATE EXTERNAL PROTECTED VIEW external_schema.remote_view
AS SELECT * FROM external_schema.remote_table;
```

Você pode acabar começando a consultar a exibição.

```
SELECT * FROM external_schema.remote_view;
```

Para obter mais informações sobre os comandos SQL relacionados a exibições no Data Catalog, consulte CREATE EXTERNAL VIEW, ALTER EXTERNAL VIEW e DROP EXTERNAL VIEW.

Considerações e limitações

Estas são considerações e limitações que podem se aplicar às exibições criadas no Data Catalog.

- AWS Glue Data CatalogAs visões do são aceitas apenas em clusters RG e RA3 provisionados ou grupos de trabalho do Redshift sem servidor.
- Você não pode criar uma exibição do Data Catalog baseada em outra exibição.
- Você só pode ter dez tabelas base em uma exibição do Data Catalog.
- O definidor da exibição deve ter permissões `SELECT GRANTABLE` completas nas tabelas base.
- As exibições só podem conter objetos e integrações do Lake Formation. Os objetos a seguir não são permitidos dentro de uma exibição.
  - Tabelas de sistema
  - Funções definidas pelo usuário (UDFs)
  - Tabelas do Redshift, exibições, visões materializadas e exibições de vinculação tardia que não estejam em um compartilhamento de dados gerenciado pelo Lake Formation.
- As exibições não podem conter tabelas do Redshift Spectrum aninhadas.
- As representações AWS Glue dos objetos base de uma exibição devem estar na mesma Conta da AWS e na mesma região da exibição.

Consultar dados espaciais no Amazon Redshift

Dados espaciais descrevem a posição e a forma de uma geometria em um espaço definido (um sistema de referência espacial). O Amazon Redshift oferece suporte a dados espaciais com os tipos de dados GEOMETRY e GEOGRAPHY, que contêm dados espaciais e, opcionalmente, o identificador do sistema de referência espacial (SRID) dos dados.

Os dados espaciais contêm dados geométricos que você pode usar para representar recursos geográficos. Entre os exemplos desse tipo de dados estão previsões do tempo, rotas em mapa, tweets com posições geográficas, locais de lojas e rotas aéreas. Os dados espaciais têm uma função importante na análise comercial, na geração de relatórios e na previsão.

É possível consultar dados espaciais com funções SQL do Amazon Redshift. Os dados espaciais contêm valores geométricos de um objeto.

As operações do tipo de dados GEOMETRY funcionam no plano cartesiano. Embora o identificador do sistema de referência espacial (SRID) seja armazenado dentro do objeto, o SRID é meramente um identificador do sistema de coordenadas e não tem nenhuma função nos algoritmos usados para processar os objetos GEOMETRY. Por outro lado, as operações do tipo de dados GEOGRAPHY tratam as coordenadas dentro dos objetos como coordenadas esféricas em um esferoide. Esse esferoide é definido pelo SRID, que faz referência a um sistema de referência espacial geográfica. Por padrão, os tipos de dados GEOGRAPHY são criados com referência espacial (SRID) 4326, referenciando o sistema geodésico mundial (WGS) 84. Para obter mais informações sobre SRIDs consulte Sistema de referência espacial na Wikipédia.

É possível usar a função ST_Transform para transformar as coordenadas de vários sistemas de referência espacial. Após concluir a transformação das coordenadas, você também pode usar um cast simples entre os dois, desde que a entrada GEOMETRY esteja codificada com o SRID geográfico. Esse cast simplesmente copia coordenadas sem realizar nenhuma outra transformação. Por exemplo:

```
SELECT ST_AsEWKT(ST_GeomFromEWKT('SRID=4326;POINT(10 20)')::geography);
```

```
st_asewkt
------------------------
 SRID=4326;POINT(10 20)
```

Para entender melhor a diferença entre os tipos de dados GEOMETRY e GEOGRAPHY, calcule a distância entre o aeroporto de Berlim (BER) e o aeroporto de São Francisco (SFO) usando o sistema geodésico mundial (WGS) 84. Com o tipo de dados GEOGRAPHY, o resultado é exibido em metros. Ao usar um tipo de dados GEOMETRY com SRID 4326, o resultado é exibido em graus, que não pode ser convertido em metros porque a distância de um grau depende de onde as geometrias do globo estão localizadas.

Utilizam-se cálculos com o tipo de dados GEOGRAPHY sobretudo para cálculos realistas de terra redonda, como a área exata de um país sem distorção. Mas a computação deles é muito mais cara. Portanto, o ST_Transform pode transformar suas coordenadas em um sistema de coordenadas projetado local adequado e fazer o cálculo com o tipo de dados GEOMETRY mais rápido.

Usando dados espaciais, você pode executar consultas para fazer o seguinte:

- Encontrar a distância entre dois pontos.
- Verificar se uma área (polígono) contém outra.
- Verificar se uma linestring intercepta outra linestring ou polígono.

É possível usar o tipo de dados GEOMETRY para manter os valores de dados espaciais. No Amazon Redshift, um valor GEOMETRY pode definir tipos de dados primitivos de geometria bidimensionais (2D), tridimensionais (3DM), bidimensionais com uma medida (3DM) e quadridimensionais (4D):

- Uma geometria bidimensional (2D) é especificada por duas coordenadas cartesianas (x, y) em um plano.
- Uma geometria tridimensional (3DZ) é especificada por três coordenadas cartesianas (x, y, z) no espaço.
- Uma geometria bidimensional com medida (3DM) é especificada por três coordenadas (x, y, m), onde as duas primeiras são coordenadas cartesianas em um plano e a terceira é uma medida.
- Uma geometria quadridimensional (4D) é especificada por quatro coordenadas (x, y, z, m), onde as três primeiras são coordenadas cartesianas em um espaço e a quarta é uma medida.

Para obter mais informações sobre tipos de dados primitivos de geometria, consulte Well-known text representation of geometry na Wikipedia.

É possível usar o tipo de dados GEOGRAPHY para manter os valores de dados espaciais. No Amazon Redshift, um valor GEOGRAPHY pode definir tipos de dados primitivos de geometria bidimensionais (2D), tridimensionais (3DM), bidimensionais com uma medida (3DM) e quadridimensionais (4D):

- Uma geometria bidimensional (2D) é especificada por coordenadas de longitude e latitude em um esferoide.
- Uma geometria tridimensional (3DZ) é especificada por coordenadas de longitude, latitude e latitude em um esferoide.
- Uma geometria bidimensional com medida (3DM) é especificada por três coordenadas (longitude, latitude e medida), em que as duas primeiras são coordenadas angulares em um plano e a terceira é uma medida.
- Uma geometria quadridimensional (4D) é especificada por quatro coordenadas (longitude, latitude, altitude, medida), em que as três primeiras são longitude, latitude e altitude, e a quarta é uma medida.

Para obter mais informações sobre os sistemas de coordenadas geográficas, consulte Sistema das coordenadas geográficas e Sistema esférico de coordenadas na Wikipédia.

Os tipos de dados `GEOMETRY` e `GEOGRAPHY` têm os seguintes subtipos:

- `POINT`
- `LINESTRING`
- `POLYGON`
- `MULTIPOINT`
- `MULTILINESTRING`
- `MULTIPOLYGON`
- `GEOMETRYCOLLECTION`

Existem funções SQL do Amazon Redshift que suportam as seguintes representações de dados geométricos:

- GeoJSON
- Well-Known Text (WKT - texto bem-conhecido).
- Extended well-known text (EWKT - texto bem-conhecido estendido)
- Representação de Well-known binary (WKB - binário bem-conhecido)
- Extended well-known binary (EWKB - binário bem-conhecido estendido)

Você pode converter os tipos de dados `GEOMETRY` e `GEOGRAPHY`.

O SQL a seguir converte uma linestring de GEOMETRY para GEOGRAPHY.

```
SELECT ST_AsEWKT(ST_GeomFromText('LINESTRING(110 40, 2 3, -10 80, -7 9)')::geography);
```

```
 st_asewkt
-------------------------------------------
 SRID=4326;LINESTRING(110 40,2 3,-10 80,-7 9)
```

O SQL a seguir converte uma linestring de GEOGRAPHY para GEOMETRY.

```
SELECT ST_AsEWKT(ST_GeogFromText('LINESTRING(110 40, 2 3, -10 80, -7 9)')::geometry);
```

```
 st_asewkt
-------------------------------------------
 SRID=4326;LINESTRING(110 40,2 3,-10 80,-7 9)
```

O Amazon Redshift oferece muitas funções SQL para consultar dados espaciais. Exceto a função ST_IsValid, funções espaciais que aceitam um objeto GEOMETRY como um argumento esperam que este objeto GEOMETRY seja uma geometria válida. Se o objeto GEOMETRY ou GEOGRAPHY não for válido, o comportamento da função espacial é indefinido. Para obter mais informações sobre validação, consulte Validade geométrica.

Para obter detalhes sobre as funções SQL para consultar dados espaciais, consulte Funções espaciais.

Para obter detalhes sobre como carregar dados espaciais, consulte Carregar uma coluna do tipo de dados GEOMETRY ou GEOGRAPHY.

Tópicos

- Tutorial: Uso de funções SQL espaciais com Amazon Redshift
- Carregar um shapefile no Amazon Redshift
- Terminologia para dados espaciais do Amazon Redshift
- Considerações ao usar dados espaciais no Amazon Redshift

Tutorial: Uso de funções SQL espaciais com Amazon Redshift

Este tutorial demonstra como usar algumas das funções SQL espaciais com o Amazon Redshift.

Para fazer isso, você consulta duas tabelas usando funções SQL espaciais. O tutorial usa dados de conjuntos de dados públicos que correlacionam dados de localização de acomodações alugadas com códigos postais em Berlim, Alemanha.

Tópicos

- Pré-requisitos
- Etapa 1: Criar tabelas e carregar dados de teste
- Etapa 2: Consultar dados espaciais
- Etapa 3: Limpar os recursos

Pré-requisitos

Para este tutorial, você precisa dos seguintes recursos:

- Um cluster e banco de dados existentes do Amazon Redshift que você pode acessar e atualizar. No cluster existente, você cria tabelas, carrega dados de amostra e executa consultas SQL para demonstrar funções espaciais. O cluster deve ter pelo menos dois nós. Para saber como criar um cluster, siga as etapas no Guia de conceitos básicos do Amazon Redshift.
- Para usar o editor de consultas do Amazon Redshift, certifique-se de que seu cluster esteja em uma região da AWS que oferece suporte ao editor de consulta. Para obter mais informações, confira Consultar um banco de dados usando o editor de consultas no Guia de gerenciamento do Amazon Redshift.
- AWSCredenciais da para seu cluster do Amazon Redshift que permitem carregar dados de teste do Amazon S3. Para obter informações sobre como acessar outros serviços da AWS, como o Amazon S3, consulte Autorizar o Amazon Redshift a acessar serviços da AWS.
- A função do AWS Identity and Access Management (IAM) chamada `mySpatialDemoRole`, que tem a política gerenciada `AmazonS3ReadOnlyAccess`anexada para ler dados do Amazon S3. Para criar um perfil com permissões para carregar dados de um bucket do Amazon S3, consulte "Autorizar operações COPY, UNLOAD, CREATE EXTERNAL FUNCTION e CREATE EXTERNAL SCHEMA usando funções do IAM" no Guia de gerenciamento de clusters do Amazon Redshift.
- Após criar a função do IAM `mySpatialDemoRole`, essa função precisa de uma associação com seu cluster do Amazon Redshift. Para obter mais informações sobre como criar essa associação,

consulte “Autorizar operações COPY, UNLOAD, CREATE EXTERNAL FUNCTION e CREATE EXTERNAL SCHEMA usando funções do IAM” no Guia de gerenciamento de clusters do Amazon Redshift.

Etapa 1: Criar tabelas e carregar dados de teste

Os dados de origem usados por este tutorial estão em arquivos chamados `accommodations.csv` e `zipcodes.csv`.

O arquivo `accommodations.csv` são dados de código aberto de insideairbnb.com. O arquivo `zipcodes.csv` fornece códigos postais que são dados de código aberto do instituto nacional de estatística de Berlim-Brandemburgo, na Alemanha (Amt für Statistik Berlim-Brandenburg). Ambas as fontes de dados são fornecidas sob uma licença Creative Commons. Os dados estão limitados à região de Berlim, Alemanha. Esses arquivos estão localizados em um bucket público do Amazon S3 para serem usados com este tutorial.

Opcionalmente, você pode baixar os dados de fonte dos seguintes links do Amazon S3:

- Dados de origem para a tabela `accommodations`.
- Dados de origem para a tabela `zipcode`.

Use o procedimento a seguir para criar tabelas e carregar dados de teste.

Para criar tabelas e carregar dados de teste

1. Abra o editor de consulta do Amazon Redshift. Para obter mais informações sobre como trabalhar com o editor de consultas, consulte “Consultar um banco de dados usando o Query Editor” no Guia de gerenciamento de clusters do Amazon Redshift.
2. Descarte todas as tabelas usadas por este tutorial se elas já existirem em seu banco de dados. Para obter mais informações, consulte Etapa 3: Limpar os recursos.
3. Crie a tabela `accommodations` para armazenar a localização geográfica de cada acomodação (longitude e latitude), o nome da listagem e outros dados comerciais.

   Este tutorial explora o aluguel de quartos em Berlim, Alemanha. A coluna `shape` armazena pontos geográficos da localização das acomodações. As outras colunas contêm informações sobre o aluguel.

Para criar a tabela `accommodations`, execute a instrução SQL a seguir no editor de consulta do Amazon Redshift.

```
CREATE TABLE public.accommodations (
  id INTEGER PRIMARY KEY,
  shape GEOMETRY,
  name VARCHAR(100),
  host_name VARCHAR(100),
  neighbourhood_group VARCHAR(100),
  neighbourhood VARCHAR(100),
  room_type VARCHAR(100),
  price SMALLINT,
  minimum_nights SMALLINT,
  number_of_reviews SMALLINT,
  last_review DATE,
  reviews_per_month NUMERIC(8,2),
  calculated_host_listings_count SMALLINT,
  availability_365 SMALLINT
);
```

4. Crie a tabela `zipcode` no editor de consulta para armazenar os códigos postais de Berlim.

   O código postal é definido como um polígono na coluna `wkb_geometry`. O resto das colunas descrevem metadados espaciais adicionais sobre o código postal.

   Para criar a tabela `zipcode`, execute a instrução SQL a seguir no editor de consulta do Amazon Redshift.

```
CREATE TABLE public.zipcode (
  ogc_field INTEGER PRIMARY KEY NOT NULL,
  wkb_geometry GEOMETRY,
  gml_id VARCHAR(256),
  spatial_name VARCHAR(256),
  spatial_alias VARCHAR(256),
  spatial_type VARCHAR(256)
 );
```

5. Carregue as tabelas usando dados de amostra.

Os dados de amostra para este tutorial são fornecidos em um bucket do Amazon S3 que permite acesso de leitura a todos os usuários da AWS autenticados. Certifique-se de fornecer credenciais da AWS válidas que permitem o acesso ao Amazon S3.

Para carregar dados de teste para suas tabelas, execute os comandos COPY a seguir. Substitua *account-number* pelo número da sua conta da AWS. O segmento da string de credenciais entre aspas simples não pode conter espaços ou quebras de linha.

```
COPY public.accommodations
FROM 's3://redshift-downloads/spatial-data/accommodations.csv'
DELIMITER ';'
IGNOREHEADER 1 REGION 'us-east-1'
CREDENTIALS 'aws_iam_role=arn:aws:iam::account-number:role/mySpatialDemoRole';
```

```
COPY public.zipcode
FROM 's3://redshift-downloads/spatial-data/zipcode.csv'
DELIMITER ';'
IGNOREHEADER 1 REGION 'us-east-1'
CREDENTIALS 'aws_iam_role=arn:aws:iam::account-number:role/mySpatialDemoRole';
```

6. Verifique se cada tabela foi carregada corretamente executando os comandos a seguir.

```
select count(*) from accommodations;
```

```
select count(*) from zipcode;
```

Os resultados a seguir mostram o número de linhas em cada tabela de dados de teste.

Nome da tabela	Linhas
acomodações	22.248
código postal	190

Etapa 2: Consultar dados espaciais

Depois que suas tabelas são criadas e carregadas, você pode consultá-las usando instruções SQL SELECT. As consultas a seguir demonstram algumas das informações que você pode recuperar. Você pode escrever muitas outras consultas que usam funções espaciais para satisfazer suas necessidades.

Para consultar dados espaciais

1. Consulte para obter a contagem do número total de listagens armazenadas na tabela `accommodations`, conforme mostrado a seguir. O sistema de referência espacial é o Sistema Geodésico Mundial (WGS - World Geodetic System) 84, que possui o identificador de referência espacial único 4326.

```
SELECT count(*) FROM public.accommodations WHERE ST_SRID(shape) = 4326;
```

```
 count
-------
 22248
```

2. Obtém os objetos de geometria em formato de texto bem-conhecido (WKT - Well-Known Text) com alguns atributos adicionais. Além disso, você pode validar se esses dados de código postal também são armazenados no Sistema Geodésico Mundial (WGS) 84, que usa o ID de referência espacial (SRID) 4326. Os dados geográficos devem ser armazenados no mesmo sistema de referência espacial para serem interoperáveis.

```
SELECT ogc_field, spatial_name, spatial_type, ST_SRID(wkb_geometry),
 ST_AsText(wkb_geometry)
FROM public.zipcode
ORDER BY spatial_name;
```

```
ogc_field  spatial_name  spatial_type  st_srid  st_astext
-------------------------------------------------------------
0          10115        Polygon        4326     POLYGON((...))
4          10117        Polygon        4326     POLYGON((...))
8          10119        Polygon        4326     POLYGON((...))
...
```

```
(190 rows returned)
```

3. Selecione o polígono de Berlin Mitte (10117), um bairro de Berlim, no formato GeoJSON, sua dimensão e o número de pontos neste polígono.

```
SELECT ogc_field, spatial_name, ST_AsGeoJSON(wkb_geometry),
 ST_Dimension(wkb_geometry), ST_NPoints(wkb_geometry)
FROM public.zipcode
WHERE spatial_name='10117';
```

```
ogc_field  spatial_name  spatial_type
 st_dimension  st_npoint
-------------------------------------------------------------------------------------------
4          10117         {"type":"Polygon", "coordinates":[[[...]]]}    2
    331
```

4. Execute o seguinte comando SQL para ver quantas acomodações estão dentro de 500 metros do Portão de Brandemburgo.

```
SELECT count(*)
FROM public.accommodations
WHERE ST_DistanceSphere(shape, ST_GeomFromText('POINT(13.377704 52.516431)', 4326))
 < 500;
```

```
count
------
  29
```

5. Obtenha a localização aproximada do Portão de Brandemburgo a partir de dados armazenados nas acomodações que estão listadas como próximas, executando a seguinte consulta.

   Esta consulta requer uma subseleção. Isso leva a uma contagem diferente porque o local solicitado não é o mesmo que a consulta anterior porque está mais perto das acomodações.

```
WITH poi(loc) as (
  SELECT st_astext(shape) FROM accommodations WHERE name LIKE '%brandenburg gate%'
)
SELECT count(*)
FROM accommodations a, poi p
```

```
WHERE ST_DistanceSphere(a.shape, ST_GeomFromText(p.loc, 4326)) < 500;
```

```
count
------
  60
```

6. Execute a seguinte consulta para mostrar os detalhes de todas as acomodações ao redor do Portão de Brandemburgo, ordenadas por preço em ordem decrescente.

```
SELECT name, price, ST_AsText(shape)
FROM public.accommodations
WHERE ST_DistanceSphere(shape, ST_GeomFromText('POINT(13.377704 52.516431)', 4326))
 < 500
ORDER BY price DESC;
```

```
name                                                    price  st_astext
-------------------------------------------------------------------------------------------
DUPLEX APARTMENT/PENTHOUSE in 5* LOCATION! 7583        300
 POINT(13.3826510209548 52.5159819722552)
DUPLEX-PENTHOUSE IN FIRST LOCATION! 7582               300
 POINT(13.3799997083855 52.5135918444834)
...
(29 rows returned)
```

7. Execute a consulta a seguir para recuperar a acomodação mais cara com o seu código postal.

```
SELECT
  a.price, a.name, ST_AsText(a.shape),
  z.spatial_name, ST_AsText(z.wkb_geometry)
FROM accommodations a, zipcode z
WHERE price = 9000 AND ST_Within(a.shape, z.wkb_geometry);
```

```
price   name                                st_astext
     spatial_name      st_astext
-------------------------------------------------------------------------------------------
```

```
9000    Ueber den Dächern Berlins Zentrum    POINT(13.334436985013
 52.4979779501538)    10777             POLYGON((13.3318284987227
 52.4956021172799,...
```

8. Calcule o preço máximo, mínimo ou mediano das acomodações usando uma subconsulta.

   A consulta a seguir lista o preço mediano das acomodações por código postal.

```
SELECT
  a.price, a.name, ST_AsText(a.shape),
  z.spatial_name, ST_AsText(z.wkb_geometry)
FROM accommodations a, zipcode z
WHERE
  ST_Within(a.shape, z.wkb_geometry) AND
  price = (SELECT median(price) FROM accommodations)
ORDER BY a.price;
```

```
price name                            st_astext
 spatial_name   st_astext
-------------------------------------------------------------------------------------------

45    "Cozy room Berlin-Mitte"        POINT(13.3864349535358 52.5292016386514)
 10115          POLYGON((13.3658598465795 52.535659581048,...
...
(723 rows returned)
```

9. Execute a consulta a seguir para recuperar o número de acomodações listadas em Berlim. Para encontrar os hot spots, estes são agrupados por código postal e classificados pela quantidade de fornecimento.

```
SELECT z.spatial_name as zip, count(*) as numAccommodations
FROM public.accommodations a, public.zipcode z
WHERE ST_Within(a.shape, z.wkb_geometry)
GROUP BY zip
ORDER BY numAccommodations DESC;
```

```
zip   numaccommodations
---------------------------
10245 872
10247 832
```

```
10437 733
10115 664
...
(187 rows returned)
```

Etapa 3: Limpar os recursos

O cluster continua a acumular cobranças enquanto estiver em execução. Ao concluir este tutorial, você pode excluir seu cluster de amostra.

Se você deseja manter o cluster, mas recuperar o armazenamento usado pelas tabelas de dados de teste, execute os comandos a seguir para excluir as tabelas.

```
drop table public.accommodations cascade;
```

```
drop table public.zipcode cascade;
```

Carregar um shapefile no Amazon Redshift

Você pode usar o comando COPY para ingerir shapefiles Esri armazenados no Amazon S3 em tabelas do Amazon Redshift. Um shapefile armazena a localização geométrica e as informações de atributos de feições geográficas em um formato vetorial. O formato shapefile pode descrever espacialmente objetos espacialmente, como pontos, linhas e polígonos. Para obter mais informações sobre um shapefile, consulte Shapefile na Wikipédia.

O comando COPY suporta o parâmetro de formato de dados `SHAPEFILE`. Por padrão, a primeira coluna do shapefile é `GEOMETRY` ou uma coluna `IDENTITY`. Todas as colunas subsequentes seguem a ordem especificada no shapefile. No entanto, a tabela de destino não precisa estar nesse layout exato porque você pode usar o mapeamento de coluna COPY para definir a ordem. Para obter informações sobre o suporte a shapefile do comando COPY, consulte SHAPEFILE.

Em alguns casos, o tamanho da geometria resultante pode ser maior que o máximo para armazenar uma geometria no Amazon Redshift. Em caso afirmativo, você pode usar a opção COPY `SIMPLIFY` ou `SIMPLIFY AUTO` para simplificar as geometrias durante a ingestão da seguinte forma:

- Especifique `SIMPLIFY` *`tolerance`* para simplificar todas as geometrias durante a ingestão usando o algoritmo Ramer-Douglas-Peucker e a tolerância dada.

- Especifique `SIMPLIFY AUTO` sem tolerância para simplificar apenas geometrias maiores que o tamanho máximo usando o algoritmo Ramer-Douglas-Peucker. Essa abordagem calcula a tolerância mínima que é grande o suficiente para armazenar o objeto dentro do limite máximo de tamanho.
- Especifique `SIMPLIFY AUTO` *`max_tolerance`* para simplificar apenas geometrias maiores que o tamanho máximo usando o algoritmo Ramer-Douglas-Peucker e a tolerância calculada automaticamente. Essa abordagem garante que a tolerância não exceda a tolerância máxima.

Para obter informações sobre o tamanho máximo de um valor de dados `GEOMETRY`, consulte Considerações ao usar dados espaciais no Amazon Redshift.

Em alguns casos, a tolerância é baixa o suficiente para que o registro não possa diminuir abaixo do tamanho máximo de um valor de dados `GEOMETRY`. Nesses casos, você pode usar a opção `MAXERROR` do comando COPY para ignorar todos ou até um certo número de erros de ingestão.

O comando COPY também suporta o carregamento de shapefiles GZIP. Para fazer isso, especifique o parâmetro COPY GZIP. Com essa opção, todos os componentes shapefile devem ser compactados independentemente e compartilhar o mesmo sufixo de compactação.

Se existir um arquivo de descrição de projeção (.prj) com o shapefile, o Redshift o usará para determinar o ID do sistema de referência espacial (SRID). Se o SRID for válido, a forma geométrica resultante terá essa SRID atribuída. Se o valor de SRID associado à forma geométrica de entrada não existir, o valor da SRID da forma geométrica resultante será zero. Você pode desabilitar a detecção automática de ID do sistema de referência espacial no nível da sessão usando `SET read_srid_on_shapefile_ingestion` para `OFF`.

Consulte nas visualizações do sistema `SYS_SPATIAL_SIMPLIFY` ou `SVL_SPATIAL_SIMPLIFY` quais registros foram simplificados, bem como a tolerância calculada. Quando você especifica `SIMPLIFY` *`tolerance`*, esta vista contém um registo para cada operação COPY. Caso contrário, ele contém um registro para cada geometria simplificada. Para ter mais informações, consulte SYS_SPATIAL_SIMPLIFY ou SVL_SPATIAL_SIMPLIFY.

Para obter exemplos de carregamento de um shapefile, consulte Carregar um shapefile no Amazon Redshift.

Terminologia para dados espaciais do Amazon Redshift

Os termos a seguir são usados para descrever algumas funções espaciais do Amazon Redshift.

Caixa delimitadora

Uma caixa delimitadora de uma geometria ou geografia é definida como o produto cruzado (entre dimensões) das extensões das coordenadas de todos os pontos da geometria ou geografia. Para geometrias bidimensionais, a caixa delimitadora é um retângulo que inclui completamente todos os pontos da geometria. Por exemplo, uma caixa delimitadora do polígono `POLYGON((0 0,1 0,0 2,0 0))` é o retângulo que é definido pelos pontos (0, 0) e (1, 2) como seus cantos inferior esquerdo e superior direito. O Amazon Redshift pré-calcula e armazena uma caixa delimitadora dentro de uma geometria para acelerar predicados geométricos e junções espaciais. Por exemplo, se as caixas delimitadoras de duas geometrias não se cruzam, então essas duas geometrias não podem se cruzar e não podem estar no conjunto de resultados de uma junção espacial usando o predicado ST_Intersects.

Você pode usar funções espaciais para adicionar (AddBBox), descartar (DropBBox) e determinar o suporte (SupportsBBox) para uma caixa delimitadora. O Amazon Redshift oferece suporte ao pré-cálculo de caixas delimitadoras para todos os subtipos de geometria.

O exemplo a seguir mostra como atualizar geometrias existentes em uma tabela para armazená-las com uma caixa delimitadora. Se o cluster estiver na versão 1.0.26809 ou posterior do cluster, todas as novas geometrias serão criadas com uma caixa delimitadora pré-computada por padrão.

```
UPDATE my_table SET geom = AddBBox(geom) WHERE SupportsBBox(geom) = false;
```

Depois de atualizar geometrias existentes, recomendamos que você execute o comando VACUUM na tabela atualizada. Para obter mais informações, consulte VACUUM.

Para definir se as geometrias são codificadas com uma caixa delimitadora durante uma sessão, consulte default_geometry_encoding.

Validade geométrica

Os algoritmos geométricos usados pelo Amazon Redshift assumem que a geometria de entrada é uma geometria válida. Se uma entrada para um algoritmo não é válida, então o resultado é indefinido. A seção a seguir descreve as definições de validade geométrica usadas pelo Amazon Redshift para cada subtipo de geometria.

Point

Um ponto será considerado válido se uma das seguintes condições for true:

- O ponto é o ponto vazio.
- Todas as coordenadas de ponto são números finitos de ponto flutuante.

Um ponto pode ser o ponto vazio.

Linestring

Uma linestring será considerada válida se uma das seguintes condições for true:

- A linestring está vazia; isto é, não contém nenhum ponto.
- Todos os pontos em uma linestring não vazio têm coordenadas que são números de ponto flutuante finito.
- A linestring, se não vazia, deve ser unidimensional; isto é, não pode degenerar até um ponto.

Uma linestring não pode conter pontos vazios.

Uma linestring pode ter pontos consecutivos duplicados.

Uma linestring pode ter auto-interseções.

Polígono

Um polígono será considerado válido se uma das seguintes condições for true:

- O polígono está vazio; isto é, não contém anéis.
- Se não estiver vazio, um polígono será válido se todas as condições a seguir forem true:
  - Todos os anéis do polígono são válidos. Um anel será considerado válido se todas as condições a seguir forem true:
    - Todos os pontos do anel têm coordenadas que são números de ponto flutuante finitos.
    - O anel está fechado; isto é, seu primeiro ponto e seu último ponto coincidem.
    - O anel não tem nenhuma auto-intersecção.
    - O anel é bidimensional.
  - Os anéis do polígono devem ter orientações consistentes. Ou seja, se você atravessar qualquer anel, o interior do polígono está à sua direita ou à sua esquerda.
  - Todos os anéis interiores devem estar dentro do anel externo do polígono.
  - Todos os anéis internos devem ter a mesma orientação, e ela deve ser oposta à orientação do anel externo.

- Os anéis interiores não podem ser aninhados; isto é, um anel interior não pode estar dentro de outro anel interior.
- Os anéis internos e externos podem se cruzar somente em um número finito de pontos.
- O interior do polígono deve ser simplesmente conectado.

Um polígono não pode conter pontos vazios.

Multiponto

Um multiponto será considerado válido se uma das seguintes condições for true:

- O multiponto está vazio; isto é, não contém nenhum ponto.
- Um multiponto não está vazio, e todos os pontos são válidos de acordo com a definição de validade de ponto.

Um multiponto pode conter um ou mais pontos vazios.

Um multiponto pode ter pontos duplicados.

Multilinestring

Uma multilinestring será considerada válida se uma das seguintes condições for verdadeira:

- A multilinestring está vazia; isto é, não contém linhas de linha.
- Todas as linestrings em uma multilinestring não vazia são válidas de acordo com a definição de validade de linestring.

Uma multilinestring não vazia que consiste em apenas linhas vazias é considerada válida.

Uma linestring vazia em uma multilinestring não afeta sua validade.

Uma multilinestring pode ter linestrings com pontos consecutivos duplicados.

Um multilinestring pode ter auto-interseções.

Uma multilinestring não pode conter pontos vazios.

Multipolígono

Um multipolígono será considerado válido se uma das seguintes condições for true:

- O multipolígono não contém nenhum polígono (está vazio).
- O multipolígono não está vazio e todas as opções a seguir são true:

- Todos os polígonos no multipolígono são válidos.
- Não há dois polígonos no multipolígono que podem se cruzar em um número infinito de pontos. Em particular, isso implica que o interior de quaisquer dois polígonos não podem se cruzar e que eles podem tocar somente em um número finito de pontos.

Um polígono vazio em um multipolígono não invalida um multipolígono.

Um multipolígono não pode conter pontos vazios.

Coleção de geometria

Uma coleção de geometria será considerada válida se uma das seguintes condições for true:

- A coleção de geometria está vazia; isto é, ela não contém nenhuma geometria.
- Todas as geometrias em uma coleção de geometria não vazia são válidas.

Esta definição ainda se aplica, embora de forma recursiva, para coleções de geometria aninhadas.

Uma coleção de geometria pode conter pontos vazios e multipontos com pontos vazios.

Simplicidade geométrica

Os algoritmos geométricos usados pelo Amazon Redshift assumem que a geometria de entrada é uma geometria válida. Se uma entrada para um algoritmo não é válida, então a verificação de simplicidade é indefinida. A seção a seguir descreve as definições de simplicidade geométrica usadas pelo Amazon Redshift para cada subtipo de geometria.

Point

Um ponto válido será considerado simples se uma das seguintes condições for true:

- Um ponto válido é sempre considerado simples.
- Um ponto vazio é considerado simples.

Linestring

Uma linestring válida será considerada simples se uma das seguintes condições for true:

- A linestring está vazia.
- A linestring não está vazia e todas as condições a seguir são true:
  - Não tem pontos consecutivos duplicados.

- Ela não tem auto-interseções, exceto possivelmente por seu primeiro ponto e último ponto, que pode coincidir. Em outras palavras, a linestring não pode ter auto-interseções exceto em pontos de limite.

Polígono

Um polígono válido é considerado simples se não contiver quaisquer pontos consecutivos duplicados.

Multiponto

Um multiponto válido será considerado simples se uma das seguintes condições for true:

- O multiponto está vazio; isto é, não contém nenhum ponto.
- Dois pontos não vazios do multiponto não coincidem.

Multilinestring

Uma multilinestring válida será considerada simples se uma das seguintes condições for true:

- A multilinestring está vazia.
- A multilinestring está vazia e todas as condições a seguir são true:
  - Todas as suas linhas são simples.
  - Quaisquer duas linestrings da multilinestring não se cruzam, exceto em pontos que são pontos de limite das duas linestrings.

Uma multilinestring não vazia que consiste apenas em linestrings vazias é considerada vazia.

Uma linestring vazia em uma multilinestring não afeta sua simplicidade.

Uma linestring fechada em uma multilinestring não pode cruzar com qualquer outra linestring na multilinestring.

Uma multilinestring não pode ter linestrings com pontos consecutivos duplicados.

Multipolígono

Um multipolígono válido é considerado simples se não contiver quaisquer pontos consecutivos duplicados.

Coleção de geometria

Uma coleção de geometria válida será considerada simples se uma das seguintes condições for true:

- A coleção de geometria está vazia; isto é, ela não contém nenhuma geometria.

- Todas as geometrias em uma coleção de geometria não vazia são simples.

  Esta definição ainda se aplica, embora de forma recursiva, para coleções de geometria aninhadas.

H3

O H3 é um sistema de grade de indexação geoespacial hierárquico, que oferece uma maneira de indexar coordenadas espaciais com resolução de até um metro quadrado. Os dados indexados podem ser unidos em conjuntos de dados diferentes e agregados em níveis de precisão distintos. O H3 possibilita uma grande variedade de algoritmos e otimizações com base na grade, inclusive vizinhos mais próximos, caminho mais curto, suavização de gradiente e muito mais. Os índices H3 se referem a células que podem ser hexágonos ou pentágonos. O espaço é subdividido hierarquicamente dada uma resolução. O H3 dá suporte a 16 resoluções de 0 a 15, inclusive. Com `0` sendo o mais aproximado e `15` sendo o mais preciso.

O Amazon Redshift oferece as seguintes funções espaciais H3:

- H3_Boundary
- H3_Center
- H3_FromLongLat
- H3_FromPoint
- H3_IsValid
- H3_Polyfill
- H3_Resolution
- H3_ToChildren
- H3_ToParent

Considerações ao usar dados espaciais no Amazon Redshift

Veja a seguir algumas considerações ao usar dados espaciais com o Amazon Redshift:

- O tamanho máximo de um objeto `GEOMETRY` ou `GEOGRAPHY` é 1.048.447 bytes.
- O Amazon Redshift Spectrum não oferece suporte nativo para dados espaciais. Portanto, não é possível criar ou alterar uma tabela externa com uma coluna `GEOMETRY` ou `GEOGRAPHY`.

- Os tipos de dados para funções definidas pelo usuário (UDFs) não são compatíveis com os tipos de dados GEOMETRY ou GEOGRAPHY.
- Não é possível usar uma coluna GEOMETRY ou GEOGRAPHY como uma chave de classificação ou uma chave de distribuição em uma tabela do Amazon Redshift.
- Não é possível usar colunas GEOMETRY ou GEOGRAPHY em cláusulas SQL ORDER BY, GROUP BY ou DISTINCT.
- Não é possível usar colunas GEOMETRY ou GEOGRAPHY em muitas funções SQL.
- Não é possível executar uma operação UNLOAD em colunas GEOMETRY ou GEOGRAPHY em todos os formatos. Você pode usar UNLOAD em colunas GEOMETRY ou GEOGRAPHY para arquivos de texto ou valores separados por vírgulas (CSV). Essa operação grava dados GEOMETRY ou GEOGRAPHY em formato EWKB hexadecimal. Se o tamanho dos dados EWKB for maior que 4 MB, ocorrerá um aviso porque os dados não poderão ser carregados em uma tabela mais tarde.
- A codificação da compactação de dados de GEOMETRY e GEOGRAPHY compatível é RAW.
- Ao usar drivers JDBC ou ODBC, use mapeamento de tipos personalizados. Nesse caso, a aplicação do cliente deve ter informações sobre quais parâmetros de um objeto ResultSet são objetos GEOMETRY ou GEOGRAPHY. A operação ResultSetMetadata retorna o tipo VARCHAR.
- Para copiar a data geográfica de um SHAPEFILE, primeiro ingira em uma coluna GEOMETRY e transmita os objetos para objetos GEOGRAPHY.

As seguintes funções não espaciais podem aceitar uma entrada do tipo GEOMETRY ou GEOGRAPHY ou colunas do tipo GEOMETRY ou GEOGRAPHY:

- A função agregada COUNT
- As expressões condicionais COALESCE e NVL
- Expressões de CASOS
- A codificação padrão para GEOMETRY e GEOGRAPHY é RAW. Para obter mais informações, consulte Codificações de compactação.

Consultar dados com consultas federadas no Amazon Redshift

Usando consultas federadas no Amazon Redshift, você pode consultar e analisar dados em bancos de dados operacionais, data warehouses e data lakes. Com o recurso de consulta federada, você pode integrar consultas do Amazon Redshift em dados ativos em bancos de dados externos com consultas em seus ambientes Amazon Redshift e Amazon S3. As consultas federadas podem funcionar com bancos de dados externos no Amazon RDS for PostgreSQL, Amazon Aurora Edição compatível com PostgreSQL, Amazon RDS for MySQL e Amazon Aurora Edição compatível com MySQL.

Você pode usar consultas federadas para incorporar dados ativos como parte de seus aplicativos de business intelligence (BI) e relatórios. Por exemplo, para facilitar a ingestão de dados para o Amazon Redshift, você pode usar consultas federadas para fazer o seguinte:

- Consultar bancos de dados operacionais diretamente.
- Aplicar transformações rapidamente.
- Carregar dados nas tabelas de destino sem a necessidade de pipelines complexos de extração, transformação e carga (ETL).

Para reduzir a movimentação de dados pela rede e melhorar a performance, o Amazon Redshift distribui parte da computação para consultas federadas diretamente nos bancos de dados operacionais remotos. O Amazon Redshift também usa sua capacidade de processamento paralelo para dar suporte à execução dessas consultas, conforme necessário.

Ao executar consultas federadas, primeiro o Amazon Redshift estabelece uma conexão de cliente com a instância de banco de dados do cluster de banco de dados do RDS ou Aurora pelo nó líder para recuperar metadados da tabela. De um nó de computação, o Amazon Redshift emite subconsultas com um predicado empurrado e recupera as linhas de resultado. O Amazon Redshift então distribui as linhas de resultados para os nós de computação para processamento adicional.

Detalhes sobre consultas enviadas ao banco de dados do Amazon Aurora PostgreSQL ou Amazon RDS for PostgreSQL são registrados na visualização do sistema SVL_FEDERATED_QUERY.

Tópicos

- Conceitos básicos do uso de consultas federadas no PostgreSQL

- Conceitos básicos do uso de consultas federadas no PostgreSQL com o AWS CloudFormation
- Conceitos básicos do uso de consultas federadas no MySQL
- Criar um segredo e uma função do IAM para usar consultas federadas
- Exemplo de uso de uma consulta federada
- Diferenças de tipo de dados entre Amazon Redshift e bancos de dados do PostgreSQL e MySQL compatíveis
- Considerações ao acessar dados federados com o Amazon Redshift

Conceitos básicos do uso de consultas federadas no PostgreSQL

Para criar uma consulta federada, siga esta abordagem geral:

1. Configure a conectividade do cluster do Amazon Redshift para sua instância de banco de dados do Amazon RDS ou Aurora PostgreSQL.

   Para isso, verifique se a instância de banco de dados do RDS PostgreSQL ou Aurora PostgreSQL é capaz de aceitar conexões do cluster do Amazon Redshift. Recomendamos que o cluster do Amazon Redshift e a instância do Amazon RDS ou Aurora PostgreSQL estejam na mesma nuvem privada virtual (VPC) e no grupo de sub-rede. Dessa forma, você pode adicionar o grupo de segurança para o cluster Amazon Redshift às regras de entrada do grupo de segurança para sua instância de Banco de Dados RDS ou Aurora PostgreSQL.

   Você também pode configurar o emparelhamento da VPC ou outra rede que permite que o Amazon Redshift faça conexões com sua instância RDS ou Aurora PostgreSQL. Para obter mais informações sobre redes de VPC, consulte as seções a seguir.

   - O que é emparelhamento da VPC? no Guia de emparelhamento da Amazon VPC
   - Trabalhar com uma instância de banco de dados em uma VPC no Guia do usuário do Amazon RDS

   > **Note**
   >
   > Há casos em que você deve habilitar o roteamento aprimorado de VPC; por exemplo, quando seu cluster do Amazon Redshift estiver em uma VPC diferente da instância do RDS ou do Aurora PostgreSQL, ou quando estiverem na mesma VPC e as rotas exigirem

esse tipo de roteamento. Caso contrário, você pode receber erros de tempo limite ao executar uma consulta federada.

2. Configure segredos no AWS Secrets Manager em seus bancos de dados do RDS PostgreSQL e Aurora PostgreSQL. Em seguida, faça referência aos segredos nas políticas e funções de acesso do AWS Identity and Access Management (IAM). Para obter mais informações, consulte Criar um segredo e uma função do IAM para usar consultas federadas.

   > **Note**
   >
   > Se seu cluster usar roteamento de VPC avançado, talvez seja necessário configurar uma interface do VPC endpoint para o AWS Secrets Manager. Isso é necessário quando a VPC e a sub-rede do cluster do Amazon Redshift não têm acesso ao endpoint do AWS Secrets Manager público. Ao usar um endpoint da interface da VPC, a comunicação entre o cluster do Amazon Redshift em sua VPC e AWS Secrets Manager é roteada de forma privada de sua VPC para a interface da endpoint. Para obter mais informações, consulte Criar um endpoint de interface no Guia do usuário da Amazon VPC.

3. Aplique a função do IAM que você criou anteriormente ao cluster do Amazon Redshift. Para obter mais informações, consulte Criar um segredo e uma função do IAM para usar consultas federadas.
4. Conecte-se aos bancos de dados do RDS PostgreSQL e Aurora PostgreSQL com um esquema externo. Para obter mais informações, consulte CREATE EXTERNAL SCHEMA. Para obter exemplos sobre como usar uma consulta federada, consulte Exemplo de uso de uma consulta federada.
5. Execute suas consultas SQL referenciando o esquema externo que referencia seus bancos de dados do RDS PostgreSQL e Aurora PostgreSQL.

Conceitos básicos do uso de consultas federadas no PostgreSQL com o AWS CloudFormation

É possível usar consultas federadas para consultar bancos de dados operacionais. Neste guia de conceitos básicos, você pode automatizar a configuração usando uma pilha de exemplo do AWS CloudFormation para habilitar uma consulta federada de um cluster do Amazon Redshift para um banco de dados sem servidor do Aurora PostgreSQL. Você pode iniciar a execução rapidamente sem precisar executar instruções SQL para provisionar seus recursos.

A pilha cria um esquema externo, referenciando sua instância do Aurora PostgreSQL, que inclui tabelas com amostra de dados. É possível consultar tabelas no esquema externo pelo cluster do Redshift.

Caso prefira começar a usar consultas federadas executando instruções SQL para configurar um esquema externo, sem usar o CloudFormation, consulte Conceitos básicos do uso de consultas federadas no PostgreSQL.

Antes de executar a pilha do CloudFormation para consultas federadas, certifique-se de ter um banco de dados sem servidor Amazon Aurora edição compatível com PostgreSQL com a API Data ativada. Ative a API Data nas propriedades do banco de dados. Se não conseguir encontrar a configuração, verifique se está executando uma instância sem servidor do Aurora PostgreSQL. Você também deve ter um cluster do Amazon Redshift que use nós RG ou RA3. Recomendamos que tanto o cluster do Redshift como a instância do Aurora PostgreSQL sem servidor estejam na mesma Virtual Private Cloud (VPC) e grupo de sub-rede. Dessa forma, você pode adicionar o grupo de segurança para o cluster Amazon Redshift às regras de entrada do grupo de segurança para sua instância de banco de dados do Aurora PostgreSQL.

Consulte mais informações sobre conceitos básicos da configuração de um cluster do Amazon Redshift em Clusters provisionados do Amazon Redshift. Para obter mais informações sobre configurar recursos com o CloudFormation, consulte O que é o AWS CloudFormation?. Consulte mais informações sobre a configuração de um cluster de banco de dados do Aurora em Criar um cluster de banco de dados do Aurora Serverless v1.

Iniciar uma pilha do CloudFormation para consultas federadas do Redshift

Use o procedimento a seguir para iniciar a pilha do CloudFormation para o Amazon Redshift a fim de habilitar consultas federadas. Antes, certifique-se de que o cluster do Amazon Redshift e a instância do Aurora PostgreSQL sem servidor estão configurados.

Para iniciar uma pilha do CloudFormation para consultas federadas

1. Escolha Iniciar pilha do CFN aqui para iniciar o serviço do CloudFormation no Console de gerenciamento da AWS.

   Se for solicitado, faça login.

   Inicia-se o processo de criação de pilha, fazendo referência a um arquivo de template do CloudFormation, que é armazenado no Amazon S3. O template do CloudFormation é um arquivo de texto em formato JSON que declara os recursos da AWS que compõem uma pilha.

2. Selecione Next (Próximo) Para inserir os detalhes da pilha.
3. Em Parameters (Parâmetros), para o cluster, insira o seguinte:

   - O nome do cluster do Amazon Redshift, por exemplo **`ra3-consumer-cluster`**
   - Um nome do banco de dados específico, por exemplo **`dev`**
   - O nome de um usuário do banco de dados, por exemplo **`consumeruser`**

   Insira também os parâmetros para o cluster de banco de dados do Aurora, incluindo usuário, nome do banco de dados, porta e endpoint. Recomendamos usar um cluster de teste e testar o banco de dados sem servidor, pois a pilha criará vários objetos de banco de dados.

   Escolha Próximo.

   São exibidas as opções de pilha.
4. Selecione Next (Próximo) para aceitar as configurações padrão.
5. Em Capabilities (Capacidades), escolha I acknowledge that AWS CloudFormation might create IAM resources (Estou ciente de que o pode criar recursos do IAM).
6. Selecione Criar pilha.

Selecione Criar pilha. O CloudFormation provisiona os recursos do template, o que leva cerca de 10 minutos, e cria um esquema externo.

Se ocorrer um erro durante a criação da pilha, faça o seguinte:

- Acesse a guia Events (Eventos) do CloudFormation para obter informações que podem ajudar você a solucionar o erro.
- Verifique se inseriu o nome correto, o nome do banco de dados e o nome de usuário do banco de dados do cluster do Redshift. Confira também os parâmetros para a instância do Aurora PostgreSQL.
- Verifique se o cluster tem nós RG ou RA3.
- Certifique-se de que o banco de dados e o cluster do Redshift estejam na mesma sub-rede e no mesmo grupo de segurança.

Consultar dados do esquema externo

Para usar o procedimento a seguir, verifique se você tem as permissões necessárias para executar consultas no cluster e no banco de dados descrito.

Para consultar um banco de dados externo com consulta federada

1. Conecte-se ao banco de dados do Redshift que você inseriu ao criar a pilha, usando uma ferramenta cliente, como o editor de consultas do Redshift.
2. Realize a consulta para o esquema externo criado pela pilha.

   ```
   select * from svv_external_schemas;
   ```

   A visualização SVV_EXTERNAL_SCHEMAS retorna informações sobre esquemas externos disponíveis. Nesse caso, o esquema externo criado pela pilha é retornado, `myfederated_schema`. Outros esquemas externos também poderão ser retornados, se houver alguma configuração. A visualização também retorna o banco de dados associado do esquema. O banco de dados é o do cluster de banco de dados do Aurora que você inseriu ao criar a pilha. A pilha adiciona uma tabela ao cluster de banco de dados do Aurora, chamada `category`, e outra tabela chamada `sales`.
3. Execute consultas SQL em tabelas do esquema externo que referencia os bancos de dados do Aurora PostgreSQL. O exemplo a seguir mostra uma consulta.

   ```
   SELECT count(*) FROM myfederated_schema.category;
   ```

   A tabela `category` retorna vários registros. Também é possível retornar registros da tabela `sales`.

   ```
   SELECT count(*) FROM myfederated_schema.sales;
   ```

   Para obter mais exemplos de , consulte Exemplo de uso de uma consulta federada.

Conceitos básicos do uso de consultas federadas no MySQL

Para criar uma consulta federada para bancos de dados do MySQL, você segue esta abordagem geral:

1. Configure a conectividade do cluster do Amazon Redshift para sua instância de banco de dados do Amazon RDS ou Aurora MySQL.

   Para fazer isso, certifique-se de que sua instância de Banco de Dados RDS MySQL ou Aurora MySQL pode aceitar conexões de seu cluster do Amazon Redshift. Recomendamos que o cluster do Amazon Redshift e a instância do Amazon RDS ou Aurora MySQL estejam na mesma nuvem privada virtual (VPC) e no grupo de sub-rede. Dessa forma, você pode adicionar o grupo de segurança para o cluster do Amazon Redshift às regras de entrada do grupo de segurança para sua instância de Banco de Dados RDS ou Aurora PostgreSQL.

   Você também pode configurar o emparelhamento da VPC ou outra rede que permite que o Amazon Redshift faça conexões com sua instância RDS ou Aurora MySQL. Para obter mais informações sobre redes de VPC, consulte as seções a seguir.

   - O que é emparelhamento da VPC? no Guia de emparelhamento da Amazon VPC
   - Trabalhar com uma instância de banco de dados em uma VPC no Guia do usuário do Amazon RDS

   > **Note**
   >
   > Se o cluster do Amazon Redshift estiver em uma VPC diferente da instância do RDS ou do Aurora MySQL, habilite o roteamento aprimorado da VPC. Caso contrário, você pode receber erros de tempo limite ao executar uma consulta federada.

2. Configure os segredos do AWS Secrets Manager para seus bancos de dados do RDS MySQL e do Aurora MySQL. Em seguida, faça referência aos segredos nas políticas e funções de acesso do AWS Identity and Access Management (IAM). Para obter mais informações, consulte Criar um segredo e uma função do IAM para usar consultas federadas.

   > **Note**
   >
   > Se seu cluster usar roteamento de VPC avançado, talvez seja necessário configurar uma interface do VPC endpoint para o AWS Secrets Manager. Isso é necessário quando a VPC e a sub-rede do cluster do Amazon Redshift não têm acesso ao endpoint público do AWS Secrets Manager. Ao usar um endpoint da interface da VPC, a comunicação entre o cluster do Amazon Redshift em sua VPC e AWS Secrets Manager é roteada de forma privada de sua VPC para a interface da endpoint. Para obter mais informações, consulte Criar um endpoint de interface no Guia do usuário da Amazon VPC.

3. Aplique a função do IAM que você criou anteriormente ao cluster do Amazon Redshift. Para obter mais informações, consulte Criar um segredo e uma função do IAM para usar consultas federadas.
4. Conecte-se aos bancos de dados do RDS MySQL e Aurora MySQL com um esquema externo. Para obter mais informações, consulte CREATE EXTERNAL SCHEMA. Para obter exemplos sobre como usar uma consulta federada, consulte Exemplo de uso de uma consulta federada com MySQL.
5. Execute suas consultas SQL referenciando o esquema externo que referencia seus bancos de dados do RDS MySQL e Aurora MySQL.

Criar um segredo e uma função do IAM para usar consultas federadas

As etapas a seguir mostram como criar um segredo e uma função do IAM para usar com consultas federadas.

Pré-requisitos

Certifique-se de ter os seguintes pré-requisitos para criar um segredo e uma função do IAM para usar com consultas federadas:

- Uma instância de banco de dados do RDS PostgreSQL, Aurora PostgreSQL, do RDS MySQL ou do Aurora com autenticação por nome de usuário e senha.
- Um cluster do Amazon Redshift com uma versão de manutenção de cluster que oferece suporte a consultas federadas.

Como criar um segredo (nome de usuário e senha) com o AWS Secrets Manager

1. Faça login no console do Secrets Manager com a conta que possui a instância do cluster de banco de dados do RDS ou Aurora.
2. Selecione Armazenar um novo segredo.
3. Escolha o bloco Credentials for RDS database (Credenciais para banco de dados do RDS). Em User name (Nome de usuário) e Password (Senha), insira valores para a instância. Confirme ou escolha um valor para Encryption key (Chave de criptografia). Em seguida, escolha o banco de dados do RDS que seu segredo acessará.

> **Note**
>
> Recomendamos usar a chave de criptografia padrão (`DefaultEncryptionKey`). Se você usar uma chave de criptografia personalizada, a função do IAM usada para acessar o segredo deverá ser adicionada como um usuário da chave.

4. Insira um nome para o segredo, siga as etapas de criação com as opções padrão e escolha Store (Armazenar).
5. Visualize o segredo e observe o valor de Secret ARN (ARN secreto) que você criou para identificar o segredo.

Como criar uma política de segurança usando o segredo

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Crie uma política com JSON semelhante às informações a seguir.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AccessSecret",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetResourcePolicy",
                "secretsmanager:GetSecretValue",
                "secretsmanager:DescribeSecret",
                "secretsmanager:ListSecretVersionIds"
            ],
            "Resource": "arn:aws:secretsmanager:us-
west-2:123456789012:secret:my-rds-secret-VNenFy"
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword",
```

```
                "secretsmanager:ListSecrets"
            ],
            "Resource": "*"
        }
    ]
}
```

Para recuperar o segredo, são necessárias ações de listagem e leitura. Recomendamos que você restrinja o recurso ao segredo específico criado. Para fazer isso, use o nome de recurso da Amazon (ARN) do segredo para limitar o recurso. Você também pode especificar as permissões e os recursos usando o editor visual no console do IAM.

3. Dê um nome à política e termine de criá-la.
4. Navegue até IAM roles (Funções do IAM).
5. Crie uma função do IAM para Redshift - Customizable (Redshift - Personalizável).
6. Anexe a política do IAM que você acabou de criar a uma função do IAM existente ou crie uma nova função do IAM e anexe a política a ela.
7. Na guia Trust relationships (Relações de confiança) da função do IAM, confirme se a função contém a entidade de confiança `redshift.amazonaws.com`.
8. Anote o Role ARN (ARN da função) que você criou. Este ARN tem acesso ao segredo.

Para anexar a função do IAM ao seu cluster do Amazon Redshift

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. No menu de navegação, escolha Clusters. Os clusters de sua conta na região atual da AWS são listados.
3. Escolha o nome do cluster na lista para visualizar mais detalhes sobre o cluster.
4. Em Actions (Ações), escolha Manage IAM roles (Gerenciar funções do IAM). A página Manage IAM roles (Gerenciar funções do IAM) é exibida.
5. Adicione a função do IAM ao cluster.

Exemplo de uso de uma consulta federada

Os exemplos a seguir mostram como executar uma consulta federada. Execute o SQL usando o cliente SQL conectado ao banco de dados do Amazon Redshift.

Exemplo de uso de uma consulta federada no PostgreSQL

O exemplo a seguir mostra como configurar uma consulta federada que faz referência a um banco de dados do Amazon Redshift, um banco de dados do Aurora PostgreSQL e Amazon S3. Este exemplo ilustra o funcionamento das consultas federadas. Para executá-lo em seu próprio ambiente, altere-o para se adequar ao ambiente. Para obter os pré-requisitos para fazer isso, consulte Conceitos básicos do uso de consultas federadas no PostgreSQL.

Crie um esquema externo que faça referência a um banco de dados do Aurora PostgreSQL.

```
CREATE EXTERNAL SCHEMA apg
FROM POSTGRES
DATABASE 'database-1' SCHEMA 'myschema'
URI 'endpoint to aurora hostname'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-SecretsManager-RO'
SECRET_ARN 'arn:aws:secretsmanager:us-west-2:123456789012:secret:federation/test/
dataplane-apg-creds-YbVKQw';
```

Crie outro esquema externo que faça referência ao Amazon S3, que usa o Amazon Redshift Spectrum. Além disso, conceda permissão para usar o esquema no modo `public`.

```
CREATE EXTERNAL SCHEMA s3
FROM DATA CATALOG
DATABASE 'default' REGION 'us-west-2'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-S3';

GRANT USAGE ON SCHEMA s3 TO public;
```

Mostra a contagem de linhas na tabela do Amazon Redshift.

```
SELECT count(*) FROM public.lineitem;

  count
----------
```

```
25075099
```

Mostra a contagem de linhas na tabela do Aurora PostgreSQL.

```
SELECT count(*) FROM apg.lineitem;

count
-------
11760
```

Mostra a contagem de linhas no Amazon S3.

```
SELECT count(*) FROM s3.lineitem_1t_part;

   count
------------
6144008876
```

Crie uma visualização das tabelas do Amazon Redshift, do Aurora PostgreSQL e do Amazon S3. Essa visualização será usada para executar a consulta federada.

```
CREATE VIEW lineitem_all AS
  SELECT
 l_orderkey,l_partkey,l_suppkey,l_linenumber,l_quantity,l_extendedprice,l_discount,l_tax,l_retu
        l_shipdate::date,l_commitdate::date,l_receiptdate::date,
 l_shipinstruct ,l_shipmode,l_comment
  FROM s3.lineitem_1t_part
  UNION ALL SELECT * FROM public.lineitem
  UNION ALL SELECT * FROM apg.lineitem
     with no schema binding;
```

Mostrar a contagem de linhas na visualização `lineitem_all` com um predicado para limitar os resultados.

```
SELECT count(*) from lineitem_all WHERE l_quantity = 10;

   count
-----------
123373836
```

Descubra quantas vezes um item vendeu em janeiro de cada ano.

```
SELECT extract(year from l_shipdate) as year,
       extract(month from l_shipdate) as month,
       count(*) as orders
FROM lineitem_all
WHERE extract(month from l_shipdate) = 1
AND l_quantity < 2
GROUP BY 1,2
ORDER BY 1,2;

 year | month | orders
------+-------+---------
 1992 |     1 |  196019
 1993 |     1 | 1582034
 1994 |     1 | 1583181
 1995 |     1 | 1583919
 1996 |     1 | 1583622
 1997 |     1 | 1586541
 1998 |     1 | 1583198
 2016 |     1 |   15542
 2017 |     1 |   15414
 2018 |     1 |   15527
 2019 |     1 |     151
```

Exemplo de uso de um nome com maiúsculas e minúsculas

Para consultar um banco de dados remoto PostgreSQL suportado que tenha um nome com maiúsculas e minúsculas de um banco de dados, esquema, tabela ou coluna, defina `enable_case_sensitive_identifier` como `true`. Para obter mais informações sobre este parâmetro de sessão, consulte enable_case_sensitive_identifier.

```
SET enable_case_sensitive_identifier TO TRUE;
```

Normalmente, os nomes do banco de dados e do esquema estão em minúsculas. O exemplo a seguir mostra como você pode se conectar a um banco de dados remoto PostgreSQL suportado que tenha nomes minúsculos para banco de dados e esquema e nomes com maiúsculas e minúsculas para tabela e coluna.

Crie um esquema externo que faça referência a um banco de dados do Aurora PostgreSQL que tenha um nome do banco de dados minúsculas (`dblower`) e nome do esquema em minúsculas (`schemalower`).

```
CREATE EXTERNAL SCHEMA apg_lower
FROM POSTGRES
DATABASE 'dblower' SCHEMA 'schemalower'
URI 'endpoint to aurora hostname'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-SecretsManager-RO'
SECRET_ARN 'arn:aws:secretsmanager:us-west-2:123456789012:secret:federation/test/
dataplane-apg-creds-YbVKQw';
```

Na sessão em que a consulta é executada, defina `enable_case_sensitive_identifier` como `true`.

```
SET enable_case_sensitive_identifier TO TRUE;
```

Execute uma consulta federada para selecionar todos os dados do banco de dados do PostgreSQL. A tabela (`MixedCaseTab`) e coluna (`MixedCaseName`) têm nomes com maiúsculas e minúsculas. O resultado é uma linha (`Harry`).

```
select * from apg_lower."MixedCaseTab";
```

```
 MixedCaseName
-------
 Harry
```

O exemplo a seguir mostra como você pode se conectar a um banco de dados remoto PostgreSQL compatível com um nome com maiúsculas e minúsculas para o banco de dados, esquema, tabela e coluna.

Defina `enable_case_sensitive_identifier` como `true` antes de criar o esquema externo. Se `enable_case_sensitive_identifier` não está definido como `true` antes de criar o esquema externo, ocorre um erro de banco de dados inexistente.

Crie um esquema externo que faça referência a um banco de dados Aurora PostgreSQL que tenha um banco de dados com maiúsculas e minúsculas (`UpperDB`) e um nome de esquema (`UpperSchema`).

```
CREATE EXTERNAL SCHEMA apg_upper
FROM POSTGRES
DATABASE 'UpperDB' SCHEMA 'UpperSchema'
```

```
URI 'endpoint to aurora hostname'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-SecretsManager-RO'
SECRET_ARN 'arn:aws:secretsmanager:us-west-2:123456789012:secret:federation/test/
dataplane-apg-creds-YbVKQw';
```

Execute uma consulta federada para selecionar todos os dados do banco de dados do PostgreSQL. A tabela (`MixedCaseTab`) e coluna (`MixedCaseName`) têm nomes com maiúsculas e minúsculas. O resultado é uma linha (`Harry`).

```
select * from apg_upper."MixedCaseTab";
```

```
 MixedCaseName
-------
 Harry
```

Exemplo de uso de uma consulta federada com MySQL

O exemplo a seguir mostra como configurar uma consulta federada que faz referência a um banco de dados do Aurora MySQL. Este exemplo ilustra como o funcionamento das consultas federadas. Para executá-lo em seu próprio ambiente, altere-o para se adequar ao ambiente. Para obter os pré-requisitos para fazer isso, consulte Conceitos básicos do uso de consultas federadas no MySQL.

Este exemplo depende dos seguintes pré-requisitos:

- Um segredo que foi configurado no Secrets Manager para o banco de dados do MySQL do Aurora. Este segredo é referenciado nas políticas e funções de acesso do IAM. Para obter mais informações, consulte Criar um segredo e uma função do IAM para usar consultas federadas.
- Um grupo de segurança configurado para vincular o Amazon Redshift e o Aurora MySQL.

Crie um esquema externo que faça referência a um banco de dados do Aurora MySQL.

```
CREATE EXTERNAL SCHEMA amysql
FROM MYSQL
DATABASE 'functional'
URI 'endpoint to remote hostname'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-SecretsManager-RO'
SECRET_ARN 'arn:aws:secretsmanager:us-west-2:123456789012:secret:federation/test/
dataplane-apg-creds-YbVKQw';
```

Execute um exemplo de seleção SQL da tabela do Aurora MySQL para exibir uma linha da tabela de funcionários no Aurora MySQL.

```
SELECT level FROM amysql.employees LIMIT 1;

 level
-------
     8
```

Diferenças de tipo de dados entre Amazon Redshift e bancos de dados do PostgreSQL e MySQL compatíveis

A tabela a seguir mostra o mapeamento de um tipo de dados do Amazon Redshift para um tipo de dados do Amazon RDS PostgreSQL ou Aurora PostgreSQL correspondente.

Tipo de dados do Amazon Redshift	Tipo de dados do RDS PostgreSQL ou Aurora PostgreSQL	Descrição
SMALLINT	SMALLINT	Número inteiro de dois bytes assinado
INTEGER	INTEGER	Número inteiro de quatro bytes assinado
BIGINT	BIGINT	Número inteiro de oito bytes assinado
DECIMAL	DECIMAL	Numérico exato com precisão selecionável
REAL	REAL	Número de ponto flutuante de precisão simples
DOUBLE PRECISION	DOUBLE PRECISION	Número de ponto flutuante de precisão dupla

Tipo de dados do Amazon Redshift	Tipo de dados do RDS PostgreSQL ou Aurora PostgreSQL	Descrição
BOOLEAN	BOOLEAN	Booleanos lógicos (verdadeiro/falso)
CHAR	CHAR	String de caracteres com comprimento fixo
VARCHAR	VARCHAR	String de caracteres de comprimento variável com limite definido pelo usuário
DATE	DATE	Data de calendário (ano, mês, dia)
TIMESTAMP	TIMESTAMP	Data e hora (sem fuso horário)
TIMESTAMPTZ	TIMESTAMPTZ	Data e hora (com fuso horário)
GEOMETRY	PostGIS GEOMETRY	Dados espaciais

Os seguintes tipos de dados do RDS PostgreSQL e Aurora PostgreSQL são convertidos em VARCHAR(64K) no Amazon Redshift:

- JSON, JSONB
- Matrizes
- BIT, BIT VARYING
- BYTEA
- Tipos compostos
- Tipos de data e hora INTERVAL, TIME, TIME WITH TIMEZONE
- Tipos enumerados
- Tipos monetários

- Tipos de endereço de rede
- Tipos numéricos SERIAL, BIGSERIAL, SMALLSERIAL e MONEY
- Tipos de identificador de objeto
- tipo pg_lsn
- Pseudotipos
- Tipos de intervalo
- Tipos de pesquisa de texto
- TXID_SNAPSHOT
- UUID
- Tipo XML

A tabela a seguir mostra o mapeamento de um tipo de dados do Amazon Redshift para um tipo de dados do Amazon RDS MySQL ou Aurora.

Tipo de dados do Amazon Redshift	Tipo de dados do RDS MySQL ou Aurora MySQL	Descrição
BOOLEAN	TINYINT(1)	Booleanos lógicos (true/false)
SMALLINT	TINYINT(UNSIGNED)	Número inteiro de dois bytes assinado
SMALLINT	SMALLINT	Número inteiro de dois bytes assinado
INTEGER	SMALLINT UNSIGNED	Número inteiro de quatro bytes assinado
INTEGER	MEDIUMINT (UNSIGNED)	Número inteiro de quatro bytes assinado
INTEGER	INT	Número inteiro de quatro bytes assinado

Tipo de dados do Amazon Redshift	Tipo de dados do RDS MySQL ou Aurora MySQL	Descrição
BIGINT	INT UNSIGNED	Número inteiro de oito bytes assinado
BIGINT	BIGINT	Número inteiro de oito bytes assinado
DECIMAL	BIGINT UNSIGNED	Numérico exato com precisão selecionável
DECIMAL	DECIMAL(M,D)	Numérico exato com precisão selecionável
REAL	FLOAT	Número de ponto flutuante de precisão simples
DOUBLE PRECISION	DOUBLE	Número de ponto flutuante de precisão dupla
CHAR	CHAR	String de caracteres com comprimento fixo
VARCHAR	VARCHAR	String de caracteres de comprimento variável com limite definido pelo usuário
DATE	DATE	Data de calendário (ano, mês, dia)
TIME	TIME	Hora (sem fuso horário)
TIMESTAMP	TIMESTAMP	Data e hora (sem fuso horário)

Tipo de dados do Amazon Redshift	Tipo de dados do RDS MySQL ou Aurora MySQL	Descrição
TIMESTAMP	DATETIME	Hora (sem fuso horário)
VARCHAR(4)	YEAR	Caractere de comprimento variável que representa o ano

Dados TIME fora do intervalo (00:00:00 - 24:00:00) resultam em erro.

Os seguintes tipos de dados do RDS MySQL e Aurora MySQL são convertidos em VARCHAR(64K) no Amazon Redshift:

- BIT
- BINARY
- VARBINARY
- TINYBLOB, BLOB, MEDIUMBLOB, LONGBLOB
- TINYTEXT, TEXT, MEDIUMTEXT, LONGTEXT
- ENUM
- SET
- SPATIAL

Considerações ao acessar dados federados com o Amazon Redshift

Alguns recursos do Amazon Redshift não oferecem suporte ao acesso a dados federados. Você pode encontrar as limitações e as considerações relacionadas a seguir.

Veja as seguintes limitações e considerações ao usar consultas federadas com o Amazon Redshift:

- Consultas federadas oferecem suporte ao acesso de leitura para fontes de dados externas. Não é possível gravar ou criar objetos de banco de dados na fonte de dados externa.

- Em alguns casos, você pode acessar um cluster de banco de dados do Amazon RDS ou Aurora em uma região da AWS que não seja a do Amazon Redshift. Esses casos normalmente incorre em cobranças de latência de rede para transferir dados entre regiões da AWS. Recomendamos o uso de um banco de dados global Aurora com um endpoint local na mesma região da AWS que seu cluster do Amazon Redshift. Os bancos de dados globais Aurora usam infraestrutura dedicada para replicação baseada em armazenamento em quaisquer duas regiões da AWS com latência típica de menos de 1 segundo.
- Considere o custo de acessar o cluster de banco de dados do Amazon RDS ou Aurora. Por exemplo, ao usar esse recurso para acessar o cluster de banco de dados do Aurora, as respectivas cobranças são baseadas em IOPS.
- Consultas federadas não habilitam o acesso ao Amazon Redshift pelo cluster de banco de dados do RDS ou Aurora.
- As consultas federadas estão disponíveis apenas em regiões da AWS onde o Amazon Redshift e o cluster de banco de dados do Amazon RDS ou Aurora estão disponíveis.
- Atualmente, as consultas federadas não oferecem suporte a `ALTER SCHEMA`. Para alterar um esquema, use `DROP` e, depois, `CREATE EXTERNAL SCHEMA`.
- Consultas federadas não funcionam com a escalabilidade da simultaneidade.
- Atualmente, as consultas federadas não oferecem suporte ao acesso por meio de um wrapper de dados externos do PostgreSQL.
- Consultas federadas para RDS MySQL ou Aurora MySQL suportam o isolamento de transações no nível READ COMMITTED.
- Se não for especificado, o Amazon Redshift se conectará ao RDS para MySQL ou ao Aurora MySQL na porta 3306. Confirme o número da porta MySQL antes de criar um esquema externo para o MySQL.
- Se não for especificado, o Amazon Redshift se conectará ao RDS PostgreSQL ou ao Aurora PostgreSQL na porta 5432. Confirme o número da porta do PostgreSQL antes de criar um esquema externo para o PostgreSQL.
- Ao buscar os tipos de dados TIMESTAMP e DATE do MySQL, os valores zero são tratados como NULL.
- Se um endpoint leitor do cluster de banco de dados do Aurora for usado, poderá ocorrer um erro de “snapshot inválido”. Isso pode ser evitado por um dos seguintes métodos:
  - Use um endpoint de instância do cluster de banco de dados específico do Aurora (em vez de usar o endpoint de cluster de banco de dados do Aurora). Esse método usa o isolamento de transação REPEATABLE READ para os resultados do banco de dados PostgreSQL.

- Use um endpoint de leitor do cluster de banco de dados do Aurora e defina `pg_federation_repeatable_read` como falso para a sessão. Esse método usa o isolamento de transação READ COMMITTED para os resultados do banco de dados PostgreSQL. Consulte mais informações sobre os endpoints de leitor do cluster de banco de dados do Aurora em Tipos de endpoints do Aurora no Guia do usuário do Amazon Aurora. Para saber mais sobre o `pg_federation_repeatable_read`, consulte pg_federation_repeatable_read.

A seguir estão as considerações para transações ao trabalhar com consultas federadas para bancos de dados do PostgreSQL:

- Se uma consulta consistir em tabelas federadas, o nó líder iniciará uma transação READ ONLY REPEATABLE READ no banco de dados remoto. Esta transação permanece durante a transação do Amazon Redshift.
- O nó líder cria um snapshot do banco de dados remoto chamando `pg_export_snapshot` e estabelece um bloqueio de leitura nas tabelas afetadas.
- Um nó de computação inicia uma transação e usa o snapshot criado no nó líder para emitir consultas no banco de dados remoto.

Versões compatíveis de bancos de dados federados

Um esquema externo do Amazon Redshift pode fazer referência a um banco de dados em um RDS PostgreSQL externo ou Aurora PostgreSQL. Quando isso acontece, estas limitações se aplicam:

- Ao criar um esquema externo com referência ao cluster de banco de dados do Aurora, o banco de dados do Aurora PostgreSQL deve estar na versão 9.6 ou posterior.
- Ao criar um esquema externo que faz referência ao Amazon RDS, o banco de dados do Amazon RDS PostgreSQL deve estar na versão 9.6 ou posterior.

Um esquema externo do Amazon Redshift pode fazer referência a um banco de dados em um RDS MySQL ou Aurora MySQL externo. Quando isso acontece, estas limitações se aplicam:

- Ao criar um esquema externo com referência ao cluster de banco de dados do Aurora, o banco de dados do Aurora MySQL deve estar na versão 5.6 ou posterior.
- Ao criar um esquema externo que faz referência ao Amazon RDS, o banco de dados do RDS MySQL deve estar na versão 5.6 ou posterior.

Permissões federadas do Amazon Redshift

As permissões federadas do Amazon Redshift simplificam o gerenciamento de permissões em vários data warehouses do Redshift, para que seja possível definir as permissões de dados uma vez e aplicá-las automaticamente em todos os data warehouses na sua Conta da AWS. Isso elimina a necessidade de redefinir, gerenciar permissões e políticas de controle de acesso refinadas em vários warehouses.

Quando você registra o namespace/cluster do warehouse do Redshift no AWS Glue Data Catalog, todos os bancos de dados dos namespaces/clusters registrados do warehouse são montados automaticamente em cada warehouse, fornecendo descoberta de dados perfeita sem configuração manual.

Você define permissões em objetos de banco de dados usando comandos SQL conhecidos do Redshift, especificando identidades globais por meio do AWS Identity and Access Management (IAM) ou do Centro de Identidade do AWS IAM. Essas permissões são armazenadas junto com os dados do warehouse e aplicadas de forma consistente, independentemente de qual warehouse executa a consulta.

Principais conceitos

- Warehouse do Redshift com permissões federadas: o warehouse do produtor registrado no Catálogo de Dados e onde os dados e as permissões do Redshift são armazenados.
- Warehouse de consumo do Redshift: qualquer warehouse que consulte dados do warehouse remoto. O warehouse de consumo pode ser habilitado para permissões federadas do Redshift.
- Identidade global: o IAM e o Centro de Identidade do IAM fornecem identidade global em todos os warehouses habilitados para permissões federadas do Redshift. Os usuários se autenticam uma vez por meio de seu provedor de identidade existente e recebem acesso consistente com base em sua identidade global, independentemente do warehouse ao qual se conectam.
- Montagem automática: todos os warehouses habilitados para permissões federadas do Redshift são automaticamente visíveis em todos os warehouses da sua conta. Esse recurso de montagem automática permite a descoberta de catálogos e bancos de dados para analytics entre warehouses.
- Propagação de identidade: quando você executa uma consulta entre warehouses, o Redshift propaga sua identidade global (perfil do IAM ou usuário do Centro de Identidade do IAM) para o warehouse remoto.

- Autorização entre warehouses: as permissões federadas do Redshift habilitadas para warehouse remoto validam suas permissões para consultas entre warehouses e são aplicadas nos warehouses de consumidor.
- Controle de acesso refinado: políticas para segurança por linha (RLS), políticas em nível de coluna (CLP) e mascaramento dinâmico de dados (DDM) que podem ser aplicadas em todos os warehouses.

Benefícios

Administração simplificada

- Defina as permissões uma vez no warehouse.
- Aplique automaticamente as mesmas permissões em todos os warehouses de consumo.
- Elimine a necessidade de redefinir, gerenciar permissões e políticas de controle de acesso refinadas em vários warehouses.
- Reduza a sobrecarga administrativa e o potencial de erros de configuração.

Aumente a segurança e a conformidade

- Mantenha a aplicação consistente das políticas de segurança em todos os data warehouses.
- Implemente controles de acesso refinados em nível de tabela e coluna.
- Faça auditoria de permissões de qualquer warehouse.
- Ferramentas de conformidade aprimoradas com comandos SHOW adicionais.

Experiência do usuário aprimorada

- Registre-se uma vez e não precise criar compartilhamentos de dados manualmente.
- SSO em todos os warehouses e acesso consistente com base na identidade global.
- Descoberta perfeita de namespaces sem configuração manual do catálogo.
- Não há necessidade de gerenciar contas de usuário locais separadas em cada warehouse.

Escalabilidade horizontal

- Adicione novos warehouses sem aumentar a complexidade da governança.

- Novos warehouses de consumo aplicam automaticamente as políticas de permissão.
- Os analistas veem imediatamente todos os bancos de dados dos warehouses registrados.

Casos de uso

Isolamento da workload com governança unificada

Separe os recursos de computação para diferentes workloads (ETL, analytics, relatórios), mantendo políticas de segurança consistentes em todos os warehouses.

Acesso a dados de várias equipes

Permita que várias equipes acessem dados compartilhados de seus próprios warehouses com controles de acesso apropriados aplicados automaticamente.

Arquitetura de data mesh

Implemente uma abordagem de data mesh em que vários recursos computacionais independentes operem em dados compartilhados com governança unificada.

Otimização de custos

Escale os recursos de computação de forma independente para diferentes casos de uso, mantendo o gerenciamento centralizado de permissões.

Considerações ao usar permissões federadas do Amazon Redshift

Veja a seguir considerações e limitações sobre o compartilhamento de dados do Amazon Redshift com o AWS Glue Data Catalog usando permissões federadas. Para acessar informações sobre considerações e limitações de compartilhamento de dados, consulte Considerações ao usar o compartilhamento de dados no Amazon Redshift.

Esse recurso é compatível somente com as versões de cluster 197 e posterior.

Requisitos de ambiente

Instâncias registradas e do Redshift de consumidor devem atender a estes requisitos:

- Tipo de instância: clusters RG ou RA3 provisionados ou grupos de trabalho com tecnologia sem servidor.

- Região: a mesma Região da AWS.
- Conta: a mesma Conta da AWS.
- Criptografia: habilitada.
- Nível de isolamento: isolamento de snapshot.

Objetos não aceitos

As instâncias de consumidor não podem acessar os seguintes objetos do catálogo de permissões federadas:

- UDFs SQL, UDFs Python e UDFs Lambda
- Modelos de ML
- Esquema externo criado na instância registrada

Restrições amplas de controle de acesso

A concessão é aceita somente em tabelas, bancos de dados, esquemas e funções usadas com a notação de 3 pontos.

Restrições refinadas de controle de acesso

Além das restrições de política padrão de segurança por linha (RLS) e mascaramento dinâmico de dados (DDM) no Amazon Redshift, as instâncias de consumidor não poderão acessar objetos protegidos por RLS ou DDM do catálogo de permissões federadas se as políticas contiverem estas funções do sistema:

- `user_is_member_of`
- `role_is_member_of`
- `user_is_member_of_role`

Observação: na versão atual do Redshift, os metadados das tabelas relacionadas ao FGAC acessadas em warehouses do Redshift de consumo estão temporariamente visíveis no catálogo.

Descoberta de metadados

- Os comandos SHOW são aceitas em colunas, tabelas, procedimentos armazenados, funções e parâmetros.

Lake Formation

- As permissões do Lake Formation não são aceitas em objetos no catálogo de permissões federadas do Amazon Redshift.

Identidade

- Somente usuários registrados no IAM ou no Centro de Identidade do AWS IAM podem consultar objetos no catálogo de permissões federadas do Amazon Redshift.
- Quando o cluster do Amazon Redshift ou o namespace do Amazon Redshift sem servidor é registrado com permissões federadas do Amazon Redshift, você não pode gerenciar a governança de dados para usuários federados do IAM usando os grupos federados do IAM. Isso inclui todos os controles de acesso granulares configurados anteriormente em objetos por meio de grupos federados do IAM.
- Ao registrar um cluster do Amazon Redshift ou um namespace do Amazon Redshift sem servidor existente com um catálogo de permissões federadas do Amazon Redshift, todos os usuários federados do Centro de Identidade do AWS IAM, incluindo aqueles que já tinham acesso, devem receber explicitamente os privilégios CONNECT para acessar o cluster ou o grupo de trabalho. Para acessar mais informações sobre como conceder privilégios CONNECT, consulte the section called “Privilégios Connect”.
- AWSOs usuários federados do IAM que se conectam aos clusters ou grupos de trabalho do Amazon Redshift usando tags de entidade principal e credenciais temporárias do IAM não são reconhecidos como identidades globais e não podem acessar os catálogos de permissões federadas do Amazon Redshift. Somente usuários federados do Centro de Identidade do AWS IAM e perfis ou usuários federados do AWS IAM estão autorizados a consultar os catálogos de permissões federadas do Amazon Redshift.
- Quando seu cluster do Amazon Redshift ou namespace do Amazon Redshift sem servidor é registrado com permissões federadas do Amazon Redshift, as seguintes limitações do comando GRANT se aplicam a perfis ou usuários federados do Centro de Identidade do AWS IAM e a perfis ou usuários federados do AWS IAM:
  - Você não pode conceder um perfil federado a nenhum usuário ou perfil. Uma exceção a essa regra é que você pode conceder um perfil de banco de dados do Redshift a um usuário federado do IAM.
  - Você não pode conceder nenhum perfil a um usuário ou perfil federado. Uma exceção a essa regra é que você pode conceder um perfil definido pelo sistema a um usuário ou perfil federado.

Acesso ao mecanismo

- O acesso de mecanismos diferentes do Redshift não é aceito

Alterar identidade global do conjunto de usuários

- Aceito apenas em “Selecionar”, “Excluir”, “Atualizar”, “Mostrar”, “Inserir”
- O perfil do IAM associado a um usuário usando ALTER USER SET GLOBAL IDENTITY é usado somente quando a consulta é feita em um data warehouse do Redshift com permissões federadas e somente quando a consulta tem como destino uma relação, como nas consultas SELECT, UPDATE e DELETE.
- Esse perfil do IAM também é usado nas consultas SHOW DATABASES, SHOW SCHEMAS e SHOW TABLES em relação a recursos no warehouse do Redshift com permissões federadas.
- Esse perfil do IAM não é usado em consultas de definição de dados, como CREATE, ALTER e DROP.

Permissão DEBUG de registro em log seguro

- A permissão DEBUG pode ser concedida somente em um banco de dados em um warehouse do Amazon Redshift com permissões federadas, não em um banco de dados local ou em um banco de dados criado a partir de um catálogo AWS Lake Formation ou AWS Glue Data Catalog.
- Somente um superusuário ou o proprietário do banco de dados pode conceder ou revogar a permissão DEBUG.
- A permissão DEBUG poderá ser concedida apenas a uma identidade global, a um grupo IdC ou aos administradores de uma conta de consumidor. Ela não pode ser concedida a PUBLIC, a um usuário local ou a um grupo local.
- Para o formulário de beneficiário ADMINS OF ACCOUNT, somente a mesma conta que possui o produtor pode ser especificada.
- A permissão DEBUG não está incluída em GRANT ALL ON DATABASE e deve ser concedida explicitamente. Pode ser concedida em apenas um banco de dados por instrução e não pode ser combinada com outras permissões na mesma instrução.
- A permissão DEBUG é apenas prospectiva. Os registros gerados enquanto a permissão DEBUG estava ativa permanecem visíveis durante toda a vida útil do log.

- Quando uma consulta se une a vários bancos de dados com permissões federadas, a permissão DEBUG deve ser concedida em cada banco de dados que contribua com dados protegidos pelo FGAC para a consulta. Caso contrário, os registros da consulta permanecerão suprimidos.
- Para obter mais informações, consulte Observações de uso para conceder a permissão DEBUG.

Mensagem de erro

- Qualquer operação não aceita no banco de dados no catálogo de permissões federadas do Amazon Redshift mostrará o seguinte erro:

```
Operation is not supported through datashares
```

Pré-requisitos

Requisitos da política do IAM para a configuração de permissões federadas do Amazon Redshift.

Com as permissões federadas do Amazon Redshift, é possível gerenciar centralmente o acesso aos dados nas suas workloads de analytics, com as permissões gerenciadas diretamente pelo data warehouse do Redshift.

Para habilitar as permissões federadas do Amazon Redshift, são necessárias permissões específicas do IAM além das permissões padrão necessárias para criar clusters provisionados do Redshift e namespaces sem servidor.

Para que o warehouse provisionado do Redshift habilite as permissões federadas do Redshift:

- `redshift:ModifyLakehouseConfiguration`
- `redshift:RegisterNamespace`

Para que o warehouse do Redshift sem servidor habilite as permissões federadas do Redshift:

- `redshift-serverless:UpdateLakehouseConfiguration`
- `redshift:RegisterNamespace`

Para a integração do AWS Glue Data Catalog criar um catálogo com permissões federadas do Redshift:

- `glue:CreateCatalog`
- `glue:GetCatalog`

Para o registro de recursos do Lake Formation como um registro único para permitir a federação de permissões do Redshift a partir de um warehouse remoto com permissões federadas:

- `lakeformation:RegisterResource`
- `lakeformation:RegisterResourceWithPrivilegedAccess`

Configuração da aplicação Centro de Identidade do IAM para o warehouse do Redshift com permissões federadas

O Amazon Redshift aceita a propagação de identidades do centro de identidade para transmitir facilmente as identidades dos usuários entre instâncias do Redshift e os serviços AWS Lake Formation e AWS Glue. Esse recurso requer a configuração de aplicações IdC dedicados.

Permissões obrigatórias do IAM

Para criar e gerenciar a aplicação do Centro de Identidade para a propagação de identidade do Centro de Identidade, verifique se suas permissões do IAM incluem as seguintes permissões:

Para o gerenciamento de aplicações IdC do Amazon Redshift:

- `redshift:CreateRedshiftIdcApplication`
- `redshift:ModifyRedshiftIdcApplication`
- `redshift:DescribeRedshiftIdcApplications`

Para o gerenciamento de aplicações IdC do Lake Formation:

- `lakeformation:CreateLakeFormationIdentityCenterConfiguration`
- `lakeformation:DescribeLakeFormationIdentityCenterConfiguration`
- `lakeformation:UpdateLakeFormationIdentityCenterConfiguration`

Crie aplicações e configurações correspondentes do IdC

Para estabelecer a propagação de identidade para suas workloads de analytics, crie uma aplicação IdC do Amazon Redshift do tipo Lakehouse. Ele gerencia as permissões sem exigir atribuições explícitas do usuário. Os warehouses do Redshift vinculados a essa aplicação exigem privilégios CONNECT para que os usuários do IdC autentiquem as conexões.

É possível criar somente uma aplicação IdC do Amazon Redshift do tipo Lakehouse por Conta da AWS. Essa aplicação lida com a propagação de identidade em todos os warehouses do Redshift que estão integrados ao Lake Formation e aos serviços do AWS Glue. A aplicação só pode ser usada com warehouses do Redshift registrados no AWS Glue Data Catalog.

Prepare o perfil do IAM assumido pelo Redshift e usado pela propagação de identidade do IdC

A criação da aplicação IdC do Redshift Lakehouse exige um perfil do IAM da sua conta com determinadas permissões do IAM. Seu perfil do IAM usado em suas aplicações IdC do Redshift deve ter o relacionamento de confiança a seguir para permitir que o Redshift o assuma e defina o contexto para a propagação da identidade do IdC.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Statement1",
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "redshift-serverless.amazonaws.com",
                    "redshift.amazonaws.com"
                ]
            },
            "Action": [
                "sts:AssumeRole",
                "sts:SetContext"
            ]
        }
    ]
}
```

E abaixo as permissões para seu perfil do IAM IdC para comportar a propagação de identidade do IdC.

- AmazonRedshiftFederatedAuthorization: essa política permite que o Amazon Redshift consulte bancos de dados AWS Glue Data Catalog por meio de permissões federadas.
- Política de contexto definida do AWSIDC

```
{
   "Version": "2012-10-17",
   "Statement": [
       {
           "Effect": "Allow",
           "Action": [
               "sts:SetContext"
           ],
           "Resource": "*"
       }
   ]
}
```

- Configurar o Redshift como uma aplicação gerenciada pela AWS com o Centro de Identidade do AWS IAM.
- Política do IAM do SSO do centro de identidade do AWSIDC:
  - `sso:DescribeApplication`: necessária para criar uma entrada do provedor de identidades (IdP) no catálogo.
  - `sso:DescribeInstance`: usada para criar manualmente funções ou usuários federados do IdP.

```
{
  "Sid": "VisualEditor1",
  "Effect": "Allow",
  "Action": [
    "sso:DescribeApplication",
    "sso:DescribeInstance"
  ],
  "Resource": [
    "arn:aws:sso:::instance/<IAM Identity Center Instance ID>",
    "arn:aws:sso::<AWS-account-id>:application/<IAM Identity Center Instance ID>/*"
  ]
}
```

Crie uma aplicação IdC do Redshift do tipo Lakehouse

CLI

Crie sua aplicação IdC do Lakehouse especificando o tipo de aplicação Lakehouse na solicitação `create-redshift-idc-application`, o que elimina a necessidade de atribuições explícitas de usuários no Centro de Identidade e, ao mesmo tempo, habilita o requisito de privilégios para autenticação `CONNECT` de usuário do IdC:

```
aws redshift create-redshift-idc-application \
--idc-instance-arn <your_idc_instance_arn> \
--idc-display-name '<name_of_idc_application_display_on_idc_console>' \
--iam-role-arn <idc_carrier_role_arn> \
--application-type Lakehouse \
--redshift-idc-application-name '<name_of_idc_display_on_redshift_console>' \
--service-integrations '[
        {
            "LakeFormation":[
                {
                    "LakeFormationQuery":{"Authorization": "Enabled"}
                }
            ]
        },
        {
            "Redshift":[
                {
                    "Connect" : {
                        "Authorization": "Enabled"
                    }
                }
            ]
        }
    ]'
```

Essa configuração permite a propagação de identidade confiável entre o Redshift e o Lake Formation, permitindo que os usuários acessem dados em vários serviços usando suas credenciais do Centro de Identidade sem atribuições adicionais de permissão.

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. Navegue até a página de conexões do Centro de Identidade do IAM e selecione Criar aplicação.

3. Defina as configurações gerais da aplicação idc.
4. Selecione Configurar permissões federadas do Amazon Redshift usando Centro de Identidade do AWS IAM (recomendado) para definir o tipo de aplicação.
5. As integrações de propagação de identidade do Lake Formation e do Redshift Connect estão habilitadas por padrão.
6. Conclua as configurações restantes do cluster e escolha Criar aplicação.

Modificar uma aplicação IdC existente do Redshift

Se você tiver uma aplicação IdC do Redshift existente que não tenha as integrações de serviços necessárias habilitadas, poderá atualizá-lo para comportar a propagação de identidade entre serviços e clusters/namespaces.

CLI

Use o comando `modify-redshift-idc-application` para habilitar as autorizações `LakeFormation:query` e `Redshift:Connect`. Essas integrações são essenciais para a propagação de identidade do IdC entre serviços e clusters:

```
aws redshift modify-redshift-idc-application \
--redshift-idc-application-arn '<arn_of_the_target_redshift_idc_application>' \
--service-integrations '[
        {
            "LakeFormation":[
                {
                    "LakeFormationQuery":{"Authorization": "Enabled"}
                }
            ]
        },
        {
            "Redshift":[
                {
                    "Connect" : {
                        "Authorization": "Enabled"
                    }
                }
            ]
        }
    ]'
```

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. Navegue até a página de conexões do Centro de Identidade do IAM e escolha uma aplicação IDC existente que você deseja editar.
3. Escolha integrações de propagação de identidade para habilitar e definir outras configurações e escolha Salvar alterações.

Criar a configuração do centro de identidade do Lake Formation

Seu serviço Lake Formation requer uma aplicação IdC dedicada, caso ainda não tenha sido criada. Você também deve habilitar a autorização `Redshift:Connect` para que a configuração funcione corretamente.

CLI

Use o comando `create-lake-formation-identity-center-configuration` para habilitar a autorização `Redshift:Connect`. Essas integrações são essenciais para que o Lake Formation propague a identidade do IdC para clusters do Redshift e namespaces do Redshift sem servidor.

```
aws lakeformation  create-lake-formation-identity-center-configuration \
--instance-arn <your_idc_instance_arn> \
--service-integrations '[{
  "Redshift": [{
    "RedshiftConnect": {
      "Authorization": "ENABLED"
    }
  }]
}]'
```

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do Lake Formation em https://console.aws.amazon.com/lakeformation/.
2. No painel de navegação esquerdo, selecione Integração com o Centro de Identidade do IAM.
3. Na página de integração do Centro de Identidade do IAM, você pode habilitar a propagação de identidade confiável para o Amazon Redshift Connect. O Lake Formation propaga a

identidade para sistemas downstream com base nas permissões efetivas, de modo que aplicações autorizadas possam acessar os dados em nome dos usuários.

Atualizar a configuração do centro de identidade do Lake Formation

Se você tiver uma aplicação IdC do Lake Formation que não tenha as integrações de serviços necessárias habilitadas, poderá atualizá-la para comportar a propagação de identidade entre serviços e clusters/namespaces.

CLI

Use o comando `update-lake-formation-identity-center-configuration` para habilitar a autorização `Redshift:Connect`. Essas integrações são essenciais para a propagação de identidade do IdC entre serviços e clusters:

```
aws lakeformation update-lake-formation-identity-center-configuration \
--service-integrations '[{

  "Redshift": [{
    "RedshiftConnect": {
      "Authorization": "ENABLED"
    }
  }]
}]'
```

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do Lake Formation em https://console.aws.amazon.com/lakeformation/.
2. No painel de navegação esquerdo, selecione Integração com o Centro de Identidade do IAM.
3. Na página de integração do Centro de Identidade do IAM, você pode habilitar a propagação de identidade confiável para o Amazon Redshift Connect. O Lake Formation propaga a identidade para sistemas downstream com base nas permissões efetivas, de modo que aplicações autorizadas possam acessar os dados em nome dos usuários.

Pré-requisitos do Lake Formation

O cliente precisa de permissões `CREATE_CATALOG` do Lake Formation para habilitar o AWS Glue Data Catalog com as permissões federadas do Amazon Redshift.

1. Se a conta pertencer a um cliente existente do Lake Formation, o administrador do Lake Formation deverá conceder explicitamente a permissão CREATE_CATALOG a cada criador do cluster. Use o seguinte exemplo de comando da CLI:

```
aws lakeformation grant-permissions \
    --cli-input-json \
    '{
        "Principal": {
            "DataLakePrincipalIdentifier": "<PrincipalArn>"
        },
        "Resource": {
            "Catalog": {}
        },
        "Permissions": [
            "CREATE_CATALOG"
        ]
    }'
```

2. Se a conta nunca usou o Lake Formation, verifique se o Catalog Creators está definido como IAMAllowedPrincipals na página Perfis e tarefas administrativas do console do Lake Formation. Se não estiver configurado, defina um administrador de data lake seguindo a seção Criar um administrador de data lake. Como alternativa, você pode criar um administrador do data lake com as políticas mínimas exigidas se você usar somente AWS Glue Data Catalog com as permissões federadas do Amazon Redshift.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Statement1",
            "Effect": "Allow",
            "Action": [
                "lakeformation:PutDataLakeSettings",
                "lakeformation:GrantPermissions",
                "lakeformation:GetDataLakeSettings",
                "lakeformation:BatchGrantPermissions",
                "lakeformation:ListPermissions"
            ],
            "Resource": [
                "*"
            ]
        }
```

```
    ]
}
```

3. Faça com que o administrador do DataLake conceda permissões às IAMAllowedPrincipals para criar um catálogo. O acesso pode ser concedido por meio do botão Conceder para Catalog Creators na página Perfil e tarefas administrativas.

Configuração de rede para o roteamento aprimorado da VPC

Se o roteamento aprimorado da VPC estiver ativado, a VPC deverá conseguir acessar os serviços do Centro de Identidade do AWS IAM. Caso contrário, a propagação de identidade falhará. Para obter mais informações sobre os requisitos de rede, consulte Usar o Centro de Identidade do AWS IAM com o roteamento aprimorado da VPC.

Privilégios Connect

Como parte das permissões federadas do Amazon Redshift, o Amazon Redshift introduziu os privilégios CONNECT para gerenciar o acesso dos usuários federados do Centro de Identidade do AWS IAM aos grupos de trabalho ou clusters do Amazon Redshift. Esse recurso está disponível quando as permissões federadas do Amazon Redshift estão habilitadas no grupo de trabalho ou no cluster.

Esse privilégio permite que os administradores controlem o acesso do usuário por meio de permissões granulares em cada grupo de trabalho ou cluster do Amazon Redshift em que as permissões federadas do Amazon Redshift estão habilitadas. O administrador do Amazon Redshift pode especificar quais usuários ou grupos federados do Centro de Identidade do AWS IAM têm acesso para se conectar diretamente ao grupo de trabalho ou cluster do Amazon Redshift, fornecendo controle refinado sobre o acesso do usuário em cada grupo de trabalho ou cluster do Centro de Identidade do AWS IAM.

Sintaxe

```
GRANT CONNECT [ON WORKGROUP]
TO <prefix>:<username> | ROLE <prefix>:<rolename> | PUBLIC;
```

CONNECT [ON WORKGROUP]

Concede permissão para se conectar a um grupo de trabalho. A permissão CONNECT é aplicável somente para identidades Centro de Identidade do AWS IAM(usuários e perfis).

TO <prefix>:<username>

Indica o usuário federado do Centro de Identidade do AWS IAM que está recebendo as permissões.

TO ROLE <prefix>:<rolename>

Indica o grupo federado do Centro de Identidade do AWS IAM que está recebendo as permissões.

PUBLIC

Concede as permissões CONNECT a todos os usuários federados do Centro de Identidade do AWS IAM, incluindo os usuários criados posteriormente.

Integração

Registro do cluster do Redshift

O Redshift comporta a criação de um cluster ou a restauração de um cluster por meio de um snapshot com registro do AWS Glue Data Catalog (GDC). É possível especificar a parte do nome do catálogo GDC desse registro. Para comportar a propagação da identidade do IdC, você pode especificar um arn da aplicação IdC do Redshift do tipo Lakehouse para permitir a propagação da identidade do IdC.

Criar um cluster com o registro do catálogo de dados Glue

CLI

Para registrar automaticamente o cluster recém-criado no Catálogo de Dados, forneça o nome do catálogo que será usado para criar e registrar o Catálogo de Dados. O parâmetro `redshift-idc-application-arn` é opcional: inclua-o se você quiser vincular o cluster à aplicação IdC do Redshift do tipo Lakehouse. Também é possível estabelecer essa associação da aplicação IdC mais tarde.

```
aws redshift create-cluster \
    --cluster-identifier 'redshift-cluster' \
   --catalog-name 'glue-data-catalog-name' \
   --redshift-idc-application-arn 'arn:aws:redshift:us-
east-1:012345678912:redshiftidcapplication:3f966e50-f1b7-495c-8ace-bd0d6c3c3b17' \
   --<other_configurations_as_needed>
```

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. Navegue até o painel de clusters provisionados e selecione Criar cluster.
3. Configure as definições adicionais de cluster.
4. Na seção Registrar no AWS Glue Data Catalog, selecione Registrar com permissões federadas do Amazon Redshift.
   - Insira um identificador de nome de catálogo.
   - (Recomendado) Selecione as permissões federadas do Amazon Redshift usando Centro de Identidade do AWS IAM para se associar à aplicação IDC do Redshift.
5. Conclua as configurações restantes do cluster e escolha Criar cluster.

Restaurar um novo cluster com registro do AWS Glue Data Catalog

CLI

Para restaurar um snapshot em um novo cluster com registro doAWS Glue Data Catalog, forneça o nome do catálogo que será usado para criar e registrar seu catálogo do AWS Glue. O parâmetro `redshift-idc-application-arn` é opcional: inclua-o se você quiser vincular o cluster à aplicação IdC do Redshift do tipo Lakehouse. Também é possível estabelecer essa associação da aplicação IdC mais tarde.

```
aws redshift restore-from-cluster-snapshot \
   --cluster-identifier 'redshift-cluster' \
   --catalog-name 'glue-data-catalog-name' \
   --snapshot-identifier 'redshift-cluster-snapshot' \
   --redshift-idc-application-arn 'arn:aws:redshift:us-
east-1:012345678912:redshiftidcapplication:3f966e50-f1b7-495c-8ace-bd0d6c3c3b17' \
   --<other_configurations_as_needed>
```

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. Acesse a página de snapshots provisionados. Na tabela de snapshots, selecione Restaurar no cluster provisionado no menu suspenso Restaurar snapshot.

3. Configurar definições adicionais de cluster.
4. Na seção Registrar no AWS Glue Data Catalog, selecione Registrar com permissões federadas do Amazon Redshift.
   - Insira um identificador de nome de catálogo.
   - (Recomendado) Selecione as permissões federadas do Amazon Redshift usando Centro de Identidade do AWS IAM para se associar à aplicação IDC do Redshift.
5. Conclua as configurações restantes do cluster e escolha Criar cluster.

Modificar um cluster existente com o registro do AWS Glue Data Catalog

Se seu cluster do Redshift já estiver associado a uma aplicação IdC do Redshift do tipo não lakehouse, o seguinte ocorre durante o registro do AWS Glue Data Catalog:

- Quando nenhum ARN da aplicação IdC do Redshift for fornecido, a aplicação IdC do Redshift existente em seu catálogo será definido como desabilitado.
- Quando uma aplicação IdC do Redshift do tipo Lakehouse de uma instância do Centro de Identidade do AWS IAM diferente é especificada, o provedor de IdC atual fica desabilitado.
- Quando uma aplicação IdC do Redshift do tipo Lakehouse da mesma instância do Centro de Identidade do AWS IAM é fornecida.
  - O ARN da aplicação IdC do Redshift em seu catálogo será alterado para o ARN da aplicação IdC do Redshift do tipo Lakehouse. O catálogo atualizado pode ser conferido consultando svv_identity_providers. Para acessar mais informações sobre svv_identity_providers, consulte svv_identity_providers.
  - Centro de Identidade do AWS IAMUsuários federados do que já tinham acesso ao cluster do Redshift devem receber explicitamente os privilégios CONNECT dos administradores para acessar o cluster. Para acessar mais informações sobre como conceder privilégios CONNECT, consulte the section called “Privilégios Connect”.
  - Depois de se registrar no AWS Glue Data Catalog, suas identidades federadas do Centro de Identidade do AWS IAM existentes e seus recursos próprios permanecem inalterados. As associações de namespace para essas identidades federadas também são preservadas.

CLI

Você pode usar o comando `modify-lakehouse-configuration` para registrar seu cluster no AWS Glue Data Catalog, então o `catalog-name` é usado para criar e registrar seu catálogo do

AWS Glue. Para comportar a propagação da identidade do IdC, especifique o arn do seu tipo de lakehouse RedshiftIdcApplication. Isso requer uma aplicação IdC do Redshift do tipo Lakehouse. Consulte Criar uma aplicação IdC do Redshift do tipo Lakehouse: configuração da aplicação do Centro de Identidade para o warehouse do Redshift com permissões federadas.

```
aws redshift modify-lakehouse-configuration \
    --cluster-identifier 'redshift-cluster' \
    --lakehouse-registration Register \
    --catalog-name 'glue-data-catalog-name' \
    --lakehouse-idc-registration Associate \
    --lakehouse-idc-application-arn 'arn:aws:redshift:us-
east-1:012345678912:redshiftidcapplication:3f966e50-f1b7-495c-8ace-bd0d6c3c3b17' \
```

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. Acesse o cluster provisionado que você deseja registrar e selecione-o.
3. Na página de detalhes do cluster, selecione Registrar no AWS Glue Data Catalog no menu suspenso Ações.
4. Selecione a opção Registrar com as permissões federadas do Amazon Redshift e

   - Insira um identificador de nome de catálogo.
   - (Recomendado) Selecione as permissões federadas do Amazon Redshift usando Centro de Identidade do AWS IAM para se associar à aplicação IDC do Redshift e escolha Registrar.

Registro do namespace do Redshift sem servidor

O Redshift sem servidor permite que namespaces sem servidor conectados ao grupo de trabalho sejam registrados no AWS Glue Data Catalog. Observe que seu banco de dados será reiniciado durante essa atualização.

Se seu namespace do Redshift sem servidor já estiver associado a uma aplicação IdC do Redshift do tipo não lakehouse, o seguinte ocorrerá durante o registro do Catálogo de Dados do Glue:

- Quando nenhum ARN da aplicação IdC do Redshift for fornecido, a aplicação IdC do Redshift existente em seu catálogo será definido como desabilitado.

- Quando uma aplicação IdC do Redshift do tipo Lakehouse de uma instância do Centro de Identidade do AWS IAM diferente é especificada, o provedor de IdC atual fica desabilitado.
- Quando uma aplicação IdC do Redshift do tipo Lakehouse da mesma instância do Centro de Identidade do AWS IAM é fornecida.
  - O ARN da aplicação IdC do Redshift em seu catálogo será alterado para o ARN da aplicação IdC do Redshift do tipo Lakehouse. O catálogo atualizado pode ser conferido consultando svv_identity_providers. Para acessar mais informações sobre svv_identity_providers, consulte svv_identity_providers.
  - Centro de Identidade do AWS IAMUsuários federados do que já tinham acesso ao cluster do Redshift devem receber explicitamente os privilégios CONNECT dos administradores para acessar o cluster. Para acessar mais informações sobre como conceder privilégios CONNECT, consulte the section called “Privilégios Connect”.
  - Depois de se registrar no AWS Glue Data Catalog, suas identidades federadas do Centro de Identidade do AWS IAM existentes e seus recursos próprios permanecem inalterados. As associações de namespace para essas identidades federadas também são preservadas.

CLI

É possível usar o comando `update-lakehouse-configuration` para registrar seu namespace do Redshift sem servidor no AWS Glue Data Catalog. O `catalog-name` é usado para criar e registrar seu catálogo do Glue. Para comportar a propagação da identidade do IDC, especifique o arn de uma aplicação Idc do Redshift do tipo Lakehouse.

```
aws redshift-serverless update-lakehouse-configuration \
    --namespace-name 'serverless-namespace-name' \
    --lakehouse-registration Register \
    --catalog-name 'glue-data-catalog-name' \
    --lakehouse-idc-registration Associate \
    --lakehouse-idc-application-arn 'arn:aws:redshift:us-
east-1:012345678912:redshiftidcapplication:3f966e50-f1b7-495c-8ace-bd0d6c3c3b17'
```

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. Acesse o cluster provisionado que você deseja registrar e selecione-o.

3. Na página de detalhes do cluster, selecione Registrar no AWS Glue Data Catalog no menu suspenso Ações.
4. Selecione a opção Registrar com as permissões federadas do Amazon Redshift e

   - Insira um identificador de nome de catálogo.
   - (Recomendado) Selecione as permissões federadas do Amazon Redshift usando Centro de Identidade do AWS IAM para se associar à aplicação IDC do Redshift e escolha Registrar.

Habilitar a propagação de identidades do Centro de Identidade do AWS IAM

O Amazon Redshift aceita a propagação de identidades do centro de identidade (IdC) para transmitir facilmente as identidades dos usuários do IdC entre instâncias do Redshift e os serviços AWS Lake Formation/AWS Glue.

Pré-requisitos

- Você criou uma aplicação IdC do Amazon Redshift do tipo Lakehouse, consulte a Configuração da aplicação Centro de Identidade do AWS IAM para o warehouse do Redshift com permissões federadas.
- Você tem um cluster do Amazon Redshift ou um namespace do Amazon Redshift sem servidor registrado no AWS Glue Data Catalog.
  - O namespace do Redshift sem servidor requer um grupo de trabalho anexado para realizar as operações relacionadas.

Se o cluster do Redshift ou o namespace do Redshift sem servidor já estiver associado a uma aplicação IdC do Redshift do tipo não lakehouse, o seguinte ocorrerá durante o registro do AWS Glue Data Catalog:

- Quando nenhum ARN da aplicação IdC do Redshift for fornecido, a aplicação IdC do Redshift existente em seu catálogo será definido como desabilitado.
- Quando uma aplicação IdC do Redshift do tipo Lakehouse de uma instância do Centro de Identidade do AWS IAM diferente é especificada, o provedor de IdC atual fica desabilitado.
- Quando uma aplicação IdC do Redshift do tipo Lakehouse da mesma instância do Centro de Identidade do AWS IAM é fornecida.

- O ARN da aplicação IdC do Redshift em seu catálogo será alterado para o ARN da aplicação IdC do Redshift do tipo Lakehouse. O catálogo atualizado pode ser conferido consultando svv_identity_providers. Para acessar mais informações sobre svv_identity_providers, consulte svv_identity_providers.
- Centro de Identidade do AWS IAMUsuários federados do que já tinham acesso ao cluster do Redshift devem receber explicitamente os privilégios CONNECT dos administradores para acessar o cluster. Para acessar mais informações sobre como conceder privilégios CONNECT, consulte the section called “Privilégios Connect”.
- Depois de se registrar no AWS Glue Data Catalog, suas identidades federadas do Centro de Identidade do AWS IAM existentes e seus recursos próprios permanecem inalterados. As associações de namespace para essas identidades federadas também são preservadas.

Habilitar a propagação de identidade do Centro de Identidade do AWS IAM para clusters provisionados do Amazon Redshift

Para o cluster provisionado do Amazon Redshift que registrou seu namespace no AWS Glue Data Catalog, ele requer a aplicação IdC do Amazon Redshift do Lakehouse, que não exige a atribuição explícita do usuário de identidade do Centro de Identidade do AWS IAM à aplicação, o privilégio de login dos usuários do IdC é gerenciado pelo privilégio CONNECT no warehouse do Redshift.

CLI

É possível usar o comando `modify-lakehouse-configuration` para habilitar a propagação da identidade do IdC para seus clusters com permissões federadas do Redshift, especifique o arn do seu lakehouse tipo RedshiftIdcApplication. Isso requer uma aplicação IdC Lakehouse do Redshift. Consulte Criar uma aplicação IdC do Redshift do tipo Lakehouse: configuração da aplicação do Centro de Identidade para o warehouse do Redshift com permissões federadas.

```
aws redshift modify-lakehouse-configuration \
    --cluster-identifier 'redshift-cluster' \
    --lakehouse-idc-registration Associate \
    --lakehouse-idc-application-arn 'arn:aws:redshift:us-
east-1:012345678912:redshiftidcapplication:3f966e50-f1b7-495c-8ace-bd0d6c3c3b17' \
```

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. Acesse o cluster provisionado que você deseja registrar e selecione-o.
3. Na página de detalhes do cluster, selecione Registrar no AWS Glue Data Catalog no menu suspenso Ações.
4. Selecione Habilitar nas permissões federadas do Amazon Redshift usando o menu suspenso do Centro de Identidade do AWS IAM para associar a aplicação IDC e escolha Salvar alterações.

Habilitar a propagação de identidade do Centro de Identidade do AWS IAM para namespaces do Amazon Redshift sem servidor

CLI

É possível usar o comando `modify-lakehouse-configuration` para habilitar a propagação da identidade do IdC para seu namespace com permissões federadas do Redshift, especifique o arn do seu lakehouse tipo RedshiftIdcApplication. Isso requer uma aplicação IdC Lakehouse do Redshift. Consulte Criar uma aplicação IdC do Redshift do tipo Lakehouse: configuração da aplicação do Centro de Identidade para o warehouse do Redshift com permissões federadas.

```
aws redshift modify-lakehouse-configuration \
    --cluster-identifier 'redshift-cluster' \
    --lakehouse-idc-registration Associate \
    --lakehouse-idc-application-arn 'arn:aws:redshift:us-
east-1:012345678912:redshiftidcapplication:3f966e50-f1b7-495c-8ace-bd0d6c3c3b17' \
```

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. Acesse o namespace sem servidor cujo registro você deseja editar e selecione-o.
3. Na página de detalhes do cluster, selecione Registrar registro do AWS Glue Data Catalog no menu suspenso Ações.

4. Selecione Habilitar nas permissões federadas do Amazon Redshift usando o menu suspenso do Centro de Identidade do AWS IAM para associar a aplicação IDC e escolha Salvar alterações.

ALTER USER SET GLOBAL IDENTITY

Além do IAM e das credenciais do Centro de Identidade do AWS IAM, o usuário que executa consultas nos warehouses do Redshift com permissões federadas pode se autenticar usando um perfil do IAM. Um superusuário pode definir um perfil do IAM para outro usuário não federado associar automaticamente no estabelecimento da sessão, e esse perfil do IAM será assumido ao fazer consultas nos warehouses do Redshift com permissões federadas. Essa funcionalidade é fornecida para permitir que os usuários do IdC da AWS se autentiquem de forma não interativa.

Esse recurso é útil para os seguintes casos de uso:

- Clientes que têm configurações grandes e complexas com o usuário existente do warehouse local, além de usuários com identidade global.
- Clientes que usam o IdC, mas que desejam poder fazer login automaticamente sem a ação interativa do navegador para fazer login.

Requisitos e limitações:

- Somente o superusuário pode definir o perfil do IAM por `ALTER USER`.
- É necessário anexar um perfil do IAM ao cluster.
- O perfil do IAM deve ter permissões para acessar os recursos necessários para executar consultas nos warehouses do Redshift com permissões federadas. Recomendamos o uso de políticas `AmazonRedshiftFederatedAuthorization` gerenciadas pela AWS.
- Os usuários que se autenticam usando o perfil do IAM GLOBAL IDENTITY podem consultar visualizações em data warehouses do Redshift com permissões federadas, mas não podem executar CREATE, ALTER, REFRESH ou DROP nessas visualizações.

Sintaxe

A sintaxe a seguir descreve o comando `ALTER USER SET GLOBAL IDENTITY` usado para definir o perfil do IAM para um usuário de banco de dados não federado executar consultas nos warehouses do Redshift com permissões federadas.

```
ALTER USER username SET
GLOBAL IDENTITY IAM_ROLE 'arn:aws:iam::<AWS-account-id>:role/<role-name>'
```

Agora, quando autenticado como usuário de destino (conectando-se diretamente como *nome de usuário* ou usando SET SESSION AUTHORIZATION), você pode conferir o perfil de identidade global usando

```
SHOW GLOBAL IDENTITY
```

Observe que o perfil de identidade global está associado ao usuário no estabelecimento da sessão. Se você definir a identidade global para o usuário atualmente conectado, esse usuário precisará se reconectar para que a identidade global tenha efeito.

O comando a seguir pode ser usado para remover o perfil do IAM associado.

```
ALTER USER username RESET GLOBAL IDENTITY
```

Parâmetros

username

Nome do usuário. Não pode ser um usuário federado, como usuário do IAM ou usuário do IdC da AWS.

IAM_ROLE 'arn:aws:iam::<account-id>:role/<role-name>'

Use o nome do recurso da Amazon (ARN) de um perfil do IAM que seu cluster usa para autenticação e autorização quando o usuário *nome do usuário* executa consultas em warehouses do Redshift com permissões federadas. Esse perfil deve ter as permissões necessárias para executar a consulta. Recomendamos o uso da política gerenciadas pela AWS `AmazonRedshiftFederatedAuthorization`.

Consultar catálogos registrados no AWS Glue Data Catalog com permissões federadas do Redshift

Depois que você registra um data warehouse do Amazon Redshift no AWS Glue Data Catalog usando permissões federadas do Amazon Redshift, os bancos de dados nesse namespace são

automaticamente montados em todas as instâncias do Amazon Redshift nesse Conta da AWS e região. Isso permite a consulta em ambientes de vários warehouses, mantendo a segurança por meio de identidades globais e políticas refinadas de controle de acesso (FGAC).

Pré-requisitos

Antes de consultar bancos de dados federados, verifique se você tem:

- A política AmazonRedshiftFederatedAuthorization gerenciada pela AWS foi anexada ao seu perfil ou usuário do IAM. Para ter um controle de acesso refinado, é possível conceder ações específicas do IAM dessa política em vez de anexar a política completa.
- Permissões adequadas configuradas para os objetos do catálogo no data warehouse de origem
- Acesso ao data warehouse do Amazon Redshift na mesma Conta da AWS e região.

Consultar bancos de dados federados

Você pode consultar objetos nos bancos de dados nos catálogos registrados usando sintaxes de três partes para acessar tabelas:

```
database@namespace_catalog.schema.table
```

Exemplo de consulta

```
SELECT * FROM my_database@my_namespace.sales.transactions
WHERE transaction_date >= '2024-01-01';
```

Além disso, você pode acessar objetos de bancos de dados federados por meio do banco de dados USE.

Os bancos de dados federados também estão disponíveis para conexão direta.

Segurança e controle de acesso

Identidades globais e aplicação do FGAC

Quando você habilita o data warehouse do Amazon Redshift com permissões federadas, as concessões e as políticas do FGAC definidas nesses objetos são automaticamente aplicadas nos warehouses de consumo. Isso garante segurança consistente em seu ambiente de vários warehouses.

Recursos de segurança aceitos

Você pode configurar os controles de segurança a seguir em objetos no data warehouse do Amazon Redshift, que serão aplicados em todos os warehouses de consumo.

- Privilégios em nível de coluna (CLP): conceda ou restrinja o acesso a colunas específicas.
- Segurança por linha (RLS): controle o acesso a linhas específicas com base nos atributos do usuário.
- Mascaramento dinâmico de dados (DDM): mascare automaticamente dados sensíveis com base nas permissões do usuário.

Gerenciar o controle de acesso no catálogo de permissões federadas do Amazon Redshift

Com as permissões federadas do Amazon Redshift, os usuários podem definir controles de acesso granulares e refinados de qualquer warehouse do Redshift na Conta da AWS. Permissões gerais gerenciam o acesso a tabelas, visualizações e objetos de banco de dados, incluindo permissões com escopo definido, enquanto controles refinados permitem privilégios em nível de coluna e a aplicação de políticas de segurança, como segurança por linha (RLS) e mascaramento dinâmico de dados (DDM).

Concessão/revogação

Com as permissões federadas, você pode definir permissões nos acessos em nível de tabela, como acesso para ler e gravar dados em tabelas e visualizações no banco de dados de permissões federadas do Redshift a partir de qualquer warehouse do Redshift.

```
GRANT SELECT ON "sales_db@finance-catalog".sales_schema.sales_table TO
 "IAMR:sales_analyst";

GRANT INSERT ON "sales_db@finance-catalog".sales_schema.sales_view TO
 "IAMR:sales_data_engineer";
```

```
REVOKE UPDATE ON "sales_db@finance-catalog".sales_schema.us_sales_view FROM
 "IAMR:us_sales_analyst";

REVOKE DELETE ON "sales_db@finance-catalog".sales_schema.us_sales_view FROM
 "IAMR:us_sales_analyst";
```

As permissões federadas do Amazon Redshift aceitam permissões com escopo para permitir que você conceda ou revogue permissões em todos os objetos de um tipo em um banco de dados ou esquema.

```
GRANT SELECT FOR TABLES IN SCHEMA "sales_db@finance-catalog".sales_schema TO
 "IAMR:sales_manager";

REVOKE UPDATE FOR TABLES IN SCHEMA sales_schmea DATABASE "sales_db@finance-catalog"
 FROM "IAMR:sales_analyst";
```

Você pode definir permissões para conceder/revogar acesso no banco de dados.

```
GRANT CREATE ON DATABASE "sales_db@finance-catalog" TO "IAMR:sales_admin";

REVOKE TEMP ON DATABASE "sales_db@finance-catalog" FROM "IAMR:sales_analyst";
```

Você pode conceder a permissão DEBUG para permitir que entidades selecionadas vejam registros de registro em log seguros não suprimidos de um banco de dados que use controle de acesso refinado em um warehouse do Amazon Redshift com permissões federadas. Você pode conceder a permissão DEBUG a uma identidade global, a um grupo de IdC ou aos administradores de uma conta de consumidor. Para obter mais informações, consulte Observações de uso para conceder a permissão DEBUG.

```
GRANT DEBUG ON DATABASE "sales_db@finance-catalog" TO "IAM:sales_auditor";

GRANT DEBUG ON DATABASE "sales_db@finance-catalog" TO "IAMR:audit_role";

GRANT DEBUG ON DATABASE "sales_db@finance-catalog" TO ROLE "myidc:audit_team";

GRANT DEBUG ON DATABASE "sales_db@finance-catalog" TO ADMINS OF ACCOUNT '123456789012';


REVOKE DEBUG ON DATABASE "sales_db@finance-catalog" FROM "IAM:sales_auditor";
```

Para acessar mais informações sobre a sintaxe aceita para permissões federadas do Amazon Redshift para concessão, consulte the section called “As permissões que você concede no catálogo de permissões federadas do Amazon Redshift.”.

Controle de acesso refinado

Você pode definir um controle de acesso refinado sobre dados sensíveis usando o controle de acesso em nível de coluna, a segurança por linha (RLS) e o mascaramento dinâmico de dados (DDM) em um banco de dados com permissões federadas do Amazon Redshift. O controle de acesso em nível de coluna permite definir privilégios de acesso mais refinados em nível de coluna em tabelas e visualizações. Superusuários e usuários ou perfis com o perfil `sys:secadmin` no banco de dados podem criar políticas de RLS e DDM, anexar essas políticas a relações específicas e habilitar o RLS em uma relação.

Controle de acesso em nível de coluna:

Você pode conceder ou revogar privilégios específicos em colunas individuais de uma tabela ou visualização.

```
GRANT SELECT ON "sales_db@finance-catalog".sales_schema.sales_table(order_number,
 sales_date, purchase_amount, sale_amount) TO "IAMR:sales_revenue_analyst";

REVOKE UPDATE ON "sales_db@finance-catalog".sales_schema.us_sales_view(order_number,
 sales_date, purchase_amount, sale_amount) FROM "IAMR:sales_revenue_analyst";
```

Segurança por linha (RLS):

É possível ativar ou desativar a segurança por linha para uma relação.

```
ALTER TABLE "sales_db@finance-catalog".sales_schema.sales_table ROW LEVEL SECURITY ON;
```

É possível criar, alterar ou descartar políticas de RLS no banco de dados.

```
-- Create an RLS policy
CREATE RLS POLICY "sales_db@finance-catalog".policy_america
WITH (region VARCHAR(10))
USING (region = 'USA');

-- Alter an RLS policy
ALTER RLS POLICY "sales_db@finance-catalog".policy_america
USING (region IN ('USA', 'CANADA', 'Mexico'));
```

```
-- Drop an RLS policy
DROP RLS POLICY "sales_db@finance-catalog".policy_america;
```

Você pode anexar ou desanexar uma política de RLS em uma relação.

```
-- Attach an RLS policy
ATTACH RLS POLICY "sales_db@finance-catalog".policy_america
ON "sales_db@finance-catalog".sales_schema.sales_table
TO "IAMR:america_sales_analyst";

-- Detach an RLS policy
DETACH RLS POLICY "sales_db@finance-catalog".policy_america
ON "sales_db@finance-catalog".sales_schema.sales_view_america
FROM "IAMR:global_sales_analyst";
```

Mascaramento dinâmico de dados (DDM):

É possível criar, alterar e descartar políticas de mascaramento no banco de dados.

```
-- Create a masking policy
CREATE MASKING POLICY "sales_db@finance-catalog".hash_credit
WITH (credit_card varchar(256))
USING (sha2(credit_card + 'testSalt', 256));

-- Alter an masking policy
ALTER MASKING POLICY "sales_db@finance-catalog".hash_credit
USING (sha2(credit_card + 'otherTestSalt', 256));

-- Drop an masking policy
DROP MASKING POLICY "sales_db@finance-catalog".hash_credit;
```

Você pode anexar ou desanexar uma política de mascaramento em uma relação.

```
-- Attach a masking policy
 ATTACH MASKING POLICY hash_credit
ON "sales_db@finance-catalog".sales_schema.transaction_table (credit_card)
TO "IAMR:sales_analyst" PRIORITY 30;

-- Detach a masking policy
DETACH MASKING POLICY hash_credit
ON "sales_db@finance-catalog".sales_schema.transaction_view (credit_card)
```

```
FROM "IAMR:transaction_analyst";
```

Um superusuário ou usuário com o privilégio `sys:secadmin` pode visualizar as políticas de RLS e DDM, bem como seus anexos em uma relação com permissões federadas, usando o comando SHOW POLICIES.

> **Note**
>
> - Funções definidas pelo usuário (UDF) no RLS, as definições de políticas do DDM não são aceitas com as permissões federadas do Amazon Redshift.
> - As funções SQL do Redshift user_is_member_of, role_is_member_of, user_is_member_of_role não são aceitas com permissões federadas do Amazon Redshift.

As permissões que você concede no catálogo de permissões federadas do Amazon Redshift.

Instruções SQL para apoiar o gerenciamento de permissões no banco de dados com a permissão federada do Amazon Redshift em recursos em diferentes níveis de granularidade. A sintaxe aceita referências de tabelas locais (quando conectadas ao banco de dados que contém o recurso ou quando o banco de dados de origem está em use) e referências totalmente qualificadas entre bancos de dados.

Observação

- *username* pode se referir a um usuário ou perfil do IAM ou usuário do IdC (Centro de Identidade do AWS IAM).
- *role_name* pode se referir a um grupo do IdC. Grupos do IAM não são aceitos.

Permissões de baixa granularidade

```
GRANT { SELECT | INSERT | UPDATE | DELETE | TRUNCATE} ON
[ TABLE ] { table_name | database@catalog.schema_name.table_name }
TO { username | ROLE role_name | PUBLIC }

REVOKE { SELECT | INSERT | UPDATE | DELETE | TRUNCATE } ON
[ TABLE ] { table_name | database@catalog.schema_name.table_name }
FROM { username | ROLE role_name | PUBLIC }
```

Privilégios em nível de coluna

```
GRANT { { SELECT | UPDATE | DELETE }
( column_name [, ...] ) [, ...] | ALL [ PRIVILEGES ] ( column_name [,...] ) }
ON { table_name | database@catalog.schema_name.table_name }
TO { username | ROLE role_name | PUBLIC }

REVOKE { { SELECT | UPDATE | DELETE }
( column_name [, ...] ) [, ...] | ALL [ PRIVILEGES ] ( column_name [,...] ) }
ON { table_name | database@catalog.schema_name.table_name }
FROM { username | ROLE role_name | PUBLIC }
```

Permissão DEBUG de registro em log seguro

A permissão DEBUG controla a visibilidade de registros de registro em log seguros não suprimidos em um banco de dados que use controle de acesso refinado em um warehouse do Amazon Redshift com permissões federadas. Você pode concedê-la a uma identidade global (um usuário do IAM, um perfil do IAM ou um usuário do IdC especificado por um nome com prefixo do provedor), a um grupo do IdC (especificado com ROLE) ou aos administradores de uma conta de consumidor. Para um destinatário de conta, você pode especificar apenas a mesma conta que pertence ao produtor. Para ver notas e restrições de uso, consulte Observações de uso para conceder a permissão DEBUG.

```
GRANT DEBUG
ON DATABASE database@catalog
TO { prefix:name | ROLE idc_prefix:group_name | ADMINS OF ACCOUNT 'account_id' }

REVOKE DEBUG
ON DATABASE database@catalog
FROM { prefix:name | ROLE idc_prefix:group_name | ADMINS OF ACCOUNT 'account_id' }
```

Permissões com escopo

```
GRANT { CREATE | USAGE | ALTER | DROP } [,...] | ALL [ PRIVILEGES ] }
FOR SCHEMAS IN
DATABASE database@catalog
TO { username [ WITH GRANT OPTION ] | ROLE role_name } [, ...]
```

```
GRANT
{ { SELECT | INSERT | UPDATE | DELETE | DROP | ALTER | TRUNCATE | REFERENCES }
 [, ...] } | ALL [ PRIVILEGES ] } }
FOR TABLES IN
{ SCHEMA schema_name [DATABASE database@catalog ] | DATABASE database@catalog }
TO { username [ WITH GRANT OPTION ] | ROLE role_name } [, ...]

REVOKE [ GRANT OPTION ] { CREATE | USAGE | ALTER | DROP } [,...] | ALL [ PRIVILEGES ] }

FOR SCHEMAS IN
DATABASE database@catalog
FROM { username | ROLE role_name } [, ...]

REVOKE [ GRANT OPTION ] { { SELECT | INSERT | UPDATE | DELETE | DROP | ALTER | TRUNCATE
 | REFERENCES } [, ...] } | ALL [ PRIVILEGES ] } }
FOR TABLES IN
{ SCHEMA schema_name [ DATABASE database@catalog ] | DATABASE database@catalog }
FROM { username | ROLE role_name } [, ...]
```

Permissões federadas do Amazon Redshift: exemplos completos

Veja a seguir um exemplo completo que mostra como criar e gerenciar políticas de governança de dados abrangentes usando permissões federadas do Amazon Redshift. Essas políticas incluem segurança por linha (RLS), mascaramento dinâmico de dados (DDM) e permissões em nível de coluna (CLP) que trabalham juntas para controlar o acesso aos dados com base nos perfis e condições do usuário.

Você deve ser um `superuser` ou ter o perfil sys:secadmin para executar este exemplo.

Pré-requisitos

Os exemplos pressupõem que os perfis “IAMR:role_name” abaixo já estejam presentes na conta. Caso contrário, crie-os. Além disso, o data warehouse do Redshift é registrado no AWS Glue Data Catalog com o nome “catalog_name” e tem um banco de dados de “db_name”.

Na instância do catálogo, conceda o perfil sys:secadmin ao perfil do IAM correspondente.

```
-- Grant sys:secadmin role to relevant user (must be run on Redshift catalog instance)
```

```
        GRANT ROLE sys:secadmin TO "IAMR:AccountSecurityAdminrole_name";
```

O restante será executado no warehouse de computação do Redshift.

Se você estiver usando usuários do IAM ou do IdC, poderá ignorar as duas etapas a seguir para criar usuários de warehouse local e mapeamento de identidade global.

Etapa 1: criar o warehouse local com os usuários necessários para testar as políticas de governança

```
-- Create test users.
CREATE USER alice WITH PASSWORD 'Alice_pass_1';
CREATE USER oscar WITH PASSWORD 'Oscar_pass_1';
CREATE USER sierra WITH PASSWORD 'Sierra_pass_1';
```

Etapa 2: configurar o mapeamento de perfis do IAM de identidade global

```
-- Map local users to IAM roles (executed by superuser).

-- Make user sierra a sys:secadmin by setting the global identity
-- to `IAMR:AccountSecurityAdmin`.
-- This role has been granted secadmin privilege on Redshift catalog instance.

ALTER USER sierra SET GLOBAL IDENTITY
IAM_ROLE 'arn:aws:iam::123456789012:role/AccountSecurityAdmin';

ALTER USER alice SET GLOBAL IDENTITY
IAM_ROLE 'arn:aws:iam::123456789012:role/Analyst';

ALTER USER oscar SET GLOBAL IDENTITY
IAM_ROLE 'arn:aws:iam::123456789012:role/Operator';

-- Verify global identity settings.
SET SESSION AUTHORIZATION sierra;
SHOW GLOBAL IDENTITY;

SET SESSION AUTHORIZATION alice;
SHOW GLOBAL IDENTITY;

SET SESSION AUTHORIZATION oscar;
SHOW GLOBAL IDENTITY;

-- Reset to default session.
```

```
RESET SESSION AUTHORIZATION;
```

Configurar o ambiente

Primeiro, crie tabelas e preencha-as com exemplos de dados de clientes e tabelas de pesquisa para políticas de governança.

```
-- Create the main customer table.
CREATE TABLE db_name@catalog_name.public.customers (
    id INTEGER,
    name VARCHAR(50),
    email VARCHAR(100),
    region VARCHAR(20),
    revenue DECIMAL(10,2)
);

-- Populate with sample customer data.
INSERT INTO db_name@catalog_name.public.customers VALUES
(1, 'John Smith', 'john@email.com', 'US', 1000.00),
(2, 'Jane Doe', 'jane@email.com', 'EU', 500.00),
(3, 'Mike Johnson', 'mike@email.com', 'US', 2000.00);

-- Grant basic table access.
GRANT ALL ON db_name@catalog_name.public.customers TO PUBLIC;

-- Create lookup table for region-based policies.
CREATE TABLE db_name@catalog_name.public.lookup_regions (
    region_code VARCHAR(20),
    allowed BOOLEAN
);

INSERT INTO db_name@catalog_name.public.lookup_regions VALUES
('US', TRUE),
('EU', FALSE),
('APAC', TRUE);

GRANT ALL ON db_name@catalog_name.public.lookup_regions TO PUBLIC;

-- Create lookup table for revenue-based masking.
CREATE TABLE db_name@catalog_name.public.lookup_revenue_tiers (base_revenue INTEGER);

INSERT INTO db_name@catalog_name.public.lookup_revenue_tiers VALUES (1000), (2000);
```

```
GRANT ALL ON db_name@catalog_name.public.lookup_revenue_tiers TO PUBLIC;
```

Configurar permissões em nível de coluna (CLP)

Configure o acesso em nível de coluna para diferentes usuários a fim de controlar quais colunas os usuários podem acessar.

```
-- Grant specific column access to Analyst i.e. `alice`.
GRANT SELECT (id, region) ON db_name@catalog_name.public.customers
    TO "IAMR:Analyst";

-- Grant different column access to Operator i.e. `oscar`.
GRANT SELECT (id, name, revenue) ON db_name@catalog_name.public.customers
    TO "IAMR:operator";
```

Criar políticas de segurança por linha (RLS)

Crie políticas de RLS para controlar quais linhas os usuários podem ver com base em suas concessões e condições de dados.

```
-- Switch to admin user to create policies.
SET SESSION AUTHORIZATION sierra;

-- Create simple RLS policy: Analysts see only US customers.
CREATE RLS POLICY db_name@catalog_name.us_only
WITH (region VARCHAR(20))
USING (region = 'US');

-- Attach the policy to the Analyst i.e. `alice`.
ATTACH RLS POLICY db_name@catalog_name.us_only
ON db_name@catalog_name.public.customers
TO "IAMR:Analyst";

-- Enable row level security on the table.
ALTER TABLE db_name@catalog_name.public.customers ROW LEVEL SECURITY ON;

-- Create advanced RLS policy using lookup table.
CREATE RLS POLICY db_name@catalog_name.region_lookup_policy
WITH (region VARCHAR(20)) AS r
USING (r.region IN (
    SELECT region_code
```

```
    FROM public.lookup_regions
    WHERE allowed = TRUE
));

-- Attach the lookup-based policy to Operator i.e. `oscar`.
ATTACH RLS POLICY db_name@catalog_name.region_lookup_policy
ON db_name@catalog_name.public.customers
TO "IAMR:Operator";
```

Criar políticas de mascaramento dinâmico de dados (DDM)

Crie políticas de mascaramento para ofuscar dados sensíveis com base nos perfis e condições do usuário.

```
-- Create masking policy for PII data (names and emails).
CREATE MASKING POLICY db_name@catalog_name.mask_pii
WITH (DATA VARCHAR(100))
USING (SHA2(DATA + 'secret', 256)::TEXT);

-- Attach masking to name and email columns for all users.
ATTACH MASKING POLICY db_name@catalog_name.mask_pii
ON db_name@catalog_name.public.customers (name)
TO PUBLIC;

ATTACH MASKING POLICY db_name@catalog_name.mask_pii
ON db_name@catalog_name.public.customers (email)
TO PUBLIC;

-- Create conditional masking policy for revenue data.
CREATE MASKING POLICY db_name@catalog_name.conditional_mask
WITH (revenue DECIMAL(10,2))
USING (CASE WHEN revenue IN (SELECT base_revenue
                                    FROM public.lookup_revenue_tiers)
            THEN revenue ELSE 0.00 END);

-- Attach conditional masking to Analyst i.e. `alice` with priority.
ATTACH MASKING POLICY db_name@catalog_name.conditional_mask
ON db_name@catalog_name.public.customers (revenue)
TO "IAMR:Analyst"
PRIORITY 20;
```

Auditar políticas disponíveis/aplicadas

Use os comandos SHOW para confirmar se suas políticas de governança estão configuradas corretamente.

```
-- Show all RLS policies in the database.
SHOW RLS POLICIES FROM DATABASE db_name@catalog_name LIMIT 10;

-- Show RLS policies for specific users and tables.
SHOW RLS POLICIES ON db_name@catalog_name.public.customers FOR "IAMR:Analyst" LIMIT 10;

SHOW RLS POLICIES ON db_name@catalog_name.public.customers FOR "IAMR:Operator" LIMIT
 10;
SHOW RLS POLICIES ON db_name@catalog_name.public.customers FOR PUBLIC LIMIT 10;

-- Show all masking policies in the database.
SHOW MASKING POLICIES FROM DATABASE db_name@catalog_name LIMIT 10;

-- Show Masking policies for specific users and tables.
SHOW MASKING POLICIES ON db_name@catalog_name.public.customers FOR "IAMR:Analyst" LIMIT
 10;
SHOW MASKING POLICIES ON db_name@catalog_name.public.customers FOR "IAMR:Operator"
 LIMIT 10;
SHOW MASKING POLICIES ON db_name@catalog_name.public.customers FOR PUBLIC LIMIT 10;
```

Testar padrões de acesso

Teste como diferentes usuários veem os dados com base em seus perfis e políticas aplicadas.

```
-- Test as analyst: Only US customers, only id/region columns,
-- conditional revenue masking.
SET SESSION AUTHORIZATION alice;
SELECT id, region FROM db_name@catalog_name.public.customers ORDER BY id;
SELECT id, region, revenue FROM db_name@catalog_name.public.customers ORDER BY id;

-- Test as operator: Allowed regions only, masked names, specific columns.
SET SESSION AUTHORIZATION oscar;
SELECT id, name, revenue FROM db_name@catalog_name.public.customers ORDER BY id;
SELECT id, name, region FROM db_name@catalog_name.public.customers ORDER BY id;

-- Test as admin: Full access to all data.
SET SESSION AUTHORIZATION sierra;
```

```
SELECT * FROM db_name@catalog_name.public.customers ORDER BY id;
```

Alterar políticas

Modifique as políticas existentes para mudar seu comportamento sem recriá-las.

```
-- Switch back to admin user.
SET SESSION AUTHORIZATION sierra;

-- Alter the PII masking policy to use simple string replacement.
ALTER MASKING POLICY db_name@catalog_name.mask_pii
USING ('***MASKED***'::TEXT);

-- Alter the conditional masking policy to use different threshold.
ALTER MASKING POLICY db_name@catalog_name.conditional_mask
USING (CASE WHEN revenue >= 500.00 THEN revenue ELSE -1.00 END);

-- Alter the RLS policy to show only disallowed regions.
ALTER RLS POLICY db_name@catalog_name.region_lookup_policy
USING (r.region IN (
    SELECT region_code
    FROM db_name@catalog_name.public.lookup_regions
    WHERE allowed = FALSE
));
```

Políticas de desvinculação e descarte

Remova as políticas quando não são mais necessárias.

```
-- Detach RLS policies from users.
DETACH RLS POLICY db_name@catalog_name.us_only
ON db_name@catalog_name.public.customers
FROM "IAMR:Analyst";

DETACH RLS POLICY db_name@catalog_name.region_lookup_policy
ON db_name@catalog_name.public.customers
FROM "IAMR:Operator";

-- Detach masking policies.
DETACH MASKING POLICY db_name@catalog_name.mask_pii
ON db_name@catalog_name.public.customers (name)
FROM PUBLIC;
```

```
DETACH MASKING POLICY db_name@catalog_name.mask_pii
ON db_name@catalog_name.public.customers (email)
FROM PUBLIC;

DETACH MASKING POLICY db_name@catalog_name.conditional_mask
ON db_name@catalog_name.public.customers (revenue)
FROM "IAMR:Analyst";
```

Limpeza: desative políticas de RLS e descarte.

```
-- Turn off Row-level security.
ALTER TABLE db_name@catalog_name.public.customers ROW LEVEL SECURITY OFF;

-- Drop policies.
DROP RLS POLICY db_name@catalog_name.us_only CASCADE;
DROP RLS POLICY db_name@catalog_name.region_lookup_policy CASCADE;
DROP MASKING POLICY db_name@catalog_name.mask_pii CASCADE;
DROP MASKING POLICY db_name@catalog_name.conditional_mask CASCADE;

-- Drop tables.
DROP TABLE db_name@catalog_name.public.customers;
DROP TABLE db_name@catalog_name.public.lookup_regions;
DROP TABLE db_name@catalog_name.public.lookup_revenue_tiers;
```

Redefinir a identidade global

```
-- TO rest the global identity.
ALTER USER alice RESET GLOBAL IDENTITY;
ALTER USER oscar RESET GLOBAL IDENTITY;
ALTER USER sierra RESET GLOBAL IDENTITY;
```

Configuração do usuário federado

Com as permissões federadas do Amazon Redshift, os usuários autenticados com as credenciais do IAM ou do Centro de Identidade do IAM (IdC) podem ter uma experiência consistente em todos os seus warehouses do Amazon Redshift.

Quando um usuário se conecta a qualquer cluster do Redshift, seus parâmetros de configuração, como formato de data, caminho de pesquisa e preferências de fuso horário, são aplicados

automaticamente à sessão. Isso garante que a mesma experiência do usuário seja mantida em todos os clusters, sem a necessidade de reconfiguração manual, proporcionando uma experiência perfeita e personalizada.

Exemplo

Pense em Alex, um analista de dados que configurou seu formato de data e fuso horário preferidos em seu banco de dados principal do Redshift. Quando Alex executa consultas lá, todos os valores de data aparecem no formato preferido.

Agora, quando Alex se conecta a um banco de dados compartilhado ou montado automaticamente em outro warehouse do Redshift, essas mesmas preferências são aplicadas de forma consistente. Alex vê os resultados no mesmo formato de data e fuso horário, garantindo uma experiência consistente em todos os ambientes.

Configurações de usuário que são consistentes em todos os warehouses do Amazon Redshift

As seguintes configurações em nível de usuário são sincronizadas automaticamente nos clusters do Redshift para usuários autenticados com credenciais do IAM ou do Centro de Identidade do AWS IAM (IdC):

- datestyle
- enable_case_sensitive_identifier
- extra_float_digits
- search_path
- cast_super_null_on_error
- default_geometry_encoding
- describe_field_name_in_uppercase
- downcase_delimited_identifier
- enable_case_sensitive_super_attribute
- enable_numeric_rounding
- enable_result_cache_for_session
- error_on_nondeterministic_update
- interval_forbid_composite_literals
- json_serialization_enable

- json_serialization_parse_nested_strings
- mv_enable_aqmv_for_session
- navigate_super_null_on_error
- parse_super_null_on_error
- spectrum_enable_pseudo_columns
- enable_spectrum_oid
- spectrum_query_maxerror
- stored_proc_log_min_messages
- analyze_threshold_percent
- enable_vacuum_boost
- pg_federation_repeatable_read

Gerenciamento de conexão e configuração

- Ao se conectar ao Amazon Redshift usando o JDBC, algumas configurações em nível de sessão também podem ser aplicadas automaticamente. Para acessar mais detalhes, consulte , verifique a documentação sobre configuração em nível de sessão.
- Os usuários também podem definir configurações persistentes em nível de usuário usando o ALTER USER.

Desabilitar a propagação do Centro de Identidade do AWS IAM

Antes de desabilitar a propagação do Centro de Identidade do AWS IAM, você deve ter o cluster do Amazon Redshift ou o namespace do Amazon Redshift sem servidor registrado no AWS Glue Data Catalog e associado a uma aplicação IdC do Redshift do Lakehouse. Um namespace do Amazon Redshift sem servidor requer um grupo de trabalho anexado para realizar as operações relacionadas.

Desabilitar a propagação de identidade do Centro de Identidade do AWS IAM para clusters provisionados do Amazon Redshift

Ao desabilitar a propagação de identidade do Centro de Identidade do AWS IAM para seus clusters provisionados do Amazon Redshift, a aplicação IdC do Redshift do Lakehouse anexada a ele será marcada como desabilitado no cluster.

CLI

Você pode usar o comando `modify-lakehouse-configuration` para desabilitar a propagação de identidade do IdC para seus clusters com as permissões federadas do Redshift. Observe que ele não exclui o provedor do IdC do seu cluster, mas o coloca no modo desabilitado.

```
aws redshift modify-lakehouse-configuration \
    --cluster-identifier 'redshift-cluster' \
    --lakehouse-idc-registration Disassociate \
```

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. Acesse o cluster provisionado cujo registro você deseja editar e selecione-o.
3. Na página de detalhes do cluster, selecione Registrar registro do AWS Glue Data Catalog no menu suspenso Ações.
4. Selecione Desabilitar nas permissões federadas do Amazon Redshift usando o menu suspenso do Centro de Identidade do AWS IAM para desassociar a aplicação IDC e escolha Salvar alterações.

Desabilitar a propagação de identidade do Centro de Identidade do AWS IAM para namespaces do Amazon Redshift sem servidor

CLI

Você pode usar o comando `modify-lakehouse-configuration` para desabilitar a propagação de identidade do IdC para seu namespace com as permissões federadas do Redshift. Observe que ele não exclui o provedor do IdC do seu cluster, mas o coloca no modo desabilitado.

```
aws redshift-serverless update-lakehouse-configuration \
--namespace-name 'serverless-namespace-name' \
--lakehouse-idc-registration Disassociate \
```

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. Acesse o namespace sem servidor cujo registro você deseja editar e selecione-o.
3. Na página de detalhes do namespace, selecione Editar registro do AWS Glue Data Catalog no menu suspenso Ações.
4. Selecione Desabilitar nas permissões federadas do Amazon Redshift usando o menu suspenso do Centro de Identidade do AWS IAM para desassociar a aplicação IDC e escolha Salvar alterações.

Cancelar o registro de AWS Glue Data Catalog

Requisitos da política do IAM para o cancelamento do registro de permissões federadas do Amazon Redshift.

Para cancelar o registro do seu cluster ou do namespace sem servidor do AWS Glue Data Catalog, são necessárias as permissões do IAM a seguir.

Para clusters provisionados do Redshift

- `redshift:ModifyLakehouseConfiguration`
- `redshift:DregisterNamespace`

Para o Redshift sem servidor

- `redshift-serverless:UpdateLakehouseConfiguration`
- `redshift:DregisterNamespace`

Para integração do AWS Glue Data Catalog

- `glue:DeleteCatalog`
- `glue:GetCatalog`

Para registro de recursos do Lake Formation

- `lakeformation:DeregisterResource`

Cancele o registro do Redshift do . AWS Glue Data Catalog

CLI

Você pode usar o comando `modify-lakehouse-configuration` para cancelar o registro do cluster do AWS Glue Data Catalog. Se você tiver um provedor de IdC associado ao seu cluster, ele colocará o provedor de IdC no cluster no modo desabilitado.

```
aws redshift modify-lakehouse-configuration \
--cluster-identifier 'redshift-cluster' \
--lakehouse-registration Deregister
```

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. Acesse o cluster provisionado cujo registro você deseja cancelar e selecione-o.
3. Na página de detalhes do cluster, selecione Cancelar registro do AWS Glue Data Catalog no menu suspenso Ações e escolha Cancelar registro.

Cancele o registro do namespace do Redshift sem servidor do . AWS Glue Data Catalog

CLI

Você pode usar o comando `update-lakehouse-configuration` para cancelar o registro do namespace do Redshift sem servidor do AWS Glue Data Catalog. Se você tiver um provedor de IdC associado ao seu cluster, ele colocará o provedor de IdC no cluster no modo desabilitado.

```
aws redshift modify-lakehouse-configuration \
--cluster-identifier 'redshift-cluster' \
--lakehouse-registration Deregister
```

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.

2. Acesse o cluster ou o namespace sem servidor cujo registro você deseja cancelar e selecione-o.
3. Na página de detalhes do cluster, selecione Cancelar registro do AWS Glue Data Catalog no menu suspenso Ações e escolha Cancelar registro.

Amazon Redshift Spectrum

Esta seção descreve como usar o Redshift Spectrum para ler dados do Amazon S3 de forma eficiente.

Usando o Amazon Redshift Spectrum, é possível executar consultas e recuperar dados estruturados e semiestruturados dos arquivos no Amazon S3 com eficiência, sem a necessidade de carregar os dados em tabelas do Amazon Redshift. As consultas do Redshift Spectrum aplicam o paralelismo em massa para executar com muita rapidez grandes conjuntos de dados. Grande parte do processamento ocorre na camada do Redshift Spectrum e a maioria dos dados permanece no Amazon S3. Vários clusters podem consultar simultaneamente o mesmo conjunto de dados no Amazon S3 sem precisar criar cópias dos dados de cada cluster.

Tópicos

- Visão geral do Amazon Redshift Spectrum
- Conceitos básicos do Amazon Redshift Spectrum
- Consultar Tabelas do Amazon S3 por meio do Amazon Redshift
- Políticas do IAM do Amazon Redshift Spectrum
- Redshift Spectrum e AWS Lake Formation
- Arquivos de dados para consultas no Amazon Redshift Spectrum
- Esquemas externos no Amazon Redshift Spectrum
- Tabelas externas para o Redshift Spectrum
- Uso de tabelas do Apache Iceberg com o Amazon Redshift
- Performance de consulta do Amazon Redshift Spectrum
- Opções de tratamento de dados
- Exemplo: Executar subconsultas correlacionadas no Redshift Spectrum
- Métricas no Amazon Redshift Spectrum
- Solução de problemas de consulta no Amazon Redshift Spectrum
- Tutorial: Consultar dados aninhados com o Amazon Redshift Spectrum

Visão geral do Amazon Redshift Spectrum

Esta seção apresenta detalhes para usar o Redshift Spectrum para ler dados do Amazon S3 de forma eficiente.

> **Note**
>
> As instâncias DC2 e RA3 do Amazon Redshift usam o Redshift Spectrum para executar consultas de data lake em uma frota de servidores dedicados fora do cluster. As instâncias RG do Amazon Redshift e o Amazon Redshift sem servidor incluem um mecanismo de consulta de data lake integrado que é executado nos recursos computacionais do próprio cluster. Para obter informações, consulte Consulta do data lake.

O Amazon Redshift Spectrum reside em servidores dedicados do Amazon Redshift que são independentes do seu cluster. O Amazon Redshift envia várias tarefas de computação intensiva para a camada do Redshift Spectrum, como a filtragem e a agregação de predicados. Por isso, as consultas do Redshift Spectrum utilizam muito menos a capacidade de processamento do seu cluster do que outras consultas. O Redshift Spectrum também dimensiona de forma inteligente. Com base nas demandas das consultas, o Redshift Spectrum pode usar milhares de instâncias para usufruir do processamento paralelo maciço.

As tabelas do Redshift Spectrum são criadas por meio da definição da estrutura dos arquivos e do seu registro como tabelas em um catálogo de dados externos. O catálogo de dados externos pode ser AWS Glue, o catálogo de dados que vem com o Amazon Athena ou sua própria metastore do Apache Hive. É possível criar e gerenciar tabelas externas a partir do Amazon Redshift usando comandos da linguagem de definição de dados (DDL) ou qualquer outra ferramenta que se conecte ao catálogo de dados externos. As alterações feitas no catálogo de dados externos são disponibilizadas instantaneamente para todos os clusters do Amazon Redshift.

Opcionalmente, você pode dividir as tabelas externas em partições de uma ou mais colunas. Definir partições como parte da tabela externa pode melhorar a performance. A melhora ocorre porque o otimizador de consultas do Amazon Redshift elimina as partições que não contêm dados para consulta.

As visões materializadas nas tabelas do Spectrum podem melhorar consideravelmente o custo e o desempenho. Para obter mais informações, consulte Visões materializadas em tabelas externas de data lake no Amazon Redshift Spectrum.

Após definir suas tabelas do Redshift Spectrum, você pode consultar e unir as tabelas da mesma forma como faz com qualquer outra tabela do Amazon Redshift. O Redshift Spectrum não é compatível com operações de atualização em tabelas externas. Você pode adicionar tabelas do Redshift Spectrum a vários clusters do Amazon Redshift e consultar os mesmos dados no Amazon S3 a partir de qualquer cluster na mesma região da AWS. Quando você atualiza os arquivos de dados do Amazon S3, os dados são disponibilizados instantaneamente para consulta a partir de qualquer um dos clusters do Amazon Redshift.

O AWS Glue Data Catalog que você acessa pode ser criptografado para aumentar a segurança. Se o catálogo do AWS Glue estiver criptografado, você precisará da chave do AWS Key Management Service (AWS KMS) para o AWS Glue acessar o catálogo do AWS Glue. A criptografia do catálogo do AWS Glue não está disponível em todas as regiões da AWS. Para obter uma lista de regiões da AWS compatíveis, consulte Criptografia e acesso seguro para AWS Glue no AWS GlueGuia do desenvolvedor. Para obter mais informações sobre AWS GlueCriptografia do catálogo de dados, consulte Criptografar seu AWS Glue Data Catalog no AWS Glue Guia do desenvolvedor.

> **Note**
>
> Não é possível visualizar os detalhes das tabelas do Redshift Spectrum usando os mesmos recursos das tabelas padrão do Amazon Redshift, como PG_TABLE_DEF, STV_TBL_PERM, PG_CLASS, ou information_schema. Caso sua ferramenta de business intelligence ou análise não reconheça as tabelas externas do Redshift Spectrum, configure sua aplicação para consultar SVV_EXTERNAL_TABLES e SVV_EXTERNAL_COLUMNS.

Regiões do Amazon Redshift Spectrum

O Redshift Spectrum está disponível nas Regiões da AWS onde o Amazon Redshift está disponível, a menos que especificado de outra forma na documentação específica da região. Para a disponibilidade de Região da AWS em regiões comerciais, consulte Endpoints de serviço para a API do Redshift na Referência geral da Amazon Web Services.

Limitações do Amazon Redshift Spectrum

Este tópico descreve as limitações referentes ao uso do Redshift Spectrum.

Observe as seguintes considerações sobre o Redshift Spectrum:

- Para clusters provisionados RA3 e DC2 que usam o Redshift Spectrum, o cluster e o bucket do Amazon S3 devem estar na mesma região da AWS. Os clusters RG provisionados e o Amazon Redshift sem servidor incluem um mecanismo de consulta de data lake integrado que é executado nos recursos de computação do próprio cluster e oferece suporte à consulta de dados do Amazon S3 em regiões da AWS.
- Quando o roteamento da VPC aprimorado é ativado, as consultas do Redshift Spectrum em tabelas de data lake no Amazon S3 exigem endpoints da VPC para o Amazon S3 e AWS Glue, para que o cluster possa acessar seus dados e catálogo. Observe que o Spectrum verifica seus dados do Amazon S3 em uma frota gerenciada pela AWS fora da sua VPC, portanto, esse tráfego de verificação de dados não passa pela sua VPC, mesmo com o roteamento aprimorado da VPC ativado. Para ver as etapas de configuração, consulte Consultar tabelas de data lake com roteamento de VPC aprimorado no Guia de gerenciamento do Amazon Redshift.
- O Redshift Spectrum é compatível com aliases de ponto de acesso do Amazon S3. Para obter mais informações, consulte Usar um alias em estilo de bucket para seu ponto de acesso no Guia do usuário do Amazon Simple Storage Service. Porém, o Redshift Spectrum não oferece suporte à VPC com aliases de ponto de acesso do Amazon S3. Para acessar mais informações, veja Consultar tabelas de data lake com roteamento de VPC aprimorado no Guia de gerenciamento do Amazon Redshift.
- Não é possível realizar operações de atualização ou exclusão em tabelas externas. Para criar uma nova tabela externa no esquema especificado, você pode usar CREATE EXTERNAL TABLE. Para obter mais informações sobre CREATE EXTERNAL TABLE AS, consulte CREATE EXTERNAL TABLE. Para inserir os resultados de uma consulta SELECT em tabelas externas existentes ou em catálogos externos, você pode usar INSERT (tabela externa). Para obter mais informações sobre INSERT (tabela externa), consulte INSERT (tabela externa).
- A não ser que você esteja usando um AWS Glue Data Catalog habilitado para AWS Lake Formation, não é possível controlar as permissões do usuário em uma tabela externa. Em vez disso, é possível conceder e revogar permissões no esquema externo. Para obter mais informações sobre como trabalhar com o AWS Lake Formation, consulte Redshift Spectrum e AWS Lake Formation.
- Para executar as consultas do Redshift Spectrum, o usuário do banco de dados precisa ter permissão para criar tabelas temporárias no banco de dados. O exemplo a seguir concede permissão temporária no banco de dados `spectrumdb` para o grupo de usuários `spectrumusers`.

```
grant temp on database spectrumdb to group spectrumusers;
```

Para obter mais informações, consulte GRANT.

- Ao usar o catálogo de dados do Athena ou do AWS Glue como um armazenamento de metadados, consulte "Cotas e limites" no Guia de gerenciamento de clusters do Amazon Redshift.
- O Redshift Spectrum não oferece suporte ao Amazon EMR com Kerberos.

Conceitos básicos do Amazon Redshift Spectrum

Neste tutorial, você aprenderá como usar o Amazon Redshift Spectrum para consultar dados diretamente dos arquivos no Amazon S3. Se você já tiver um cluster e um cliente SQL, poderá concluir este tutorial com configuração mínima.

> **Note**
>
> Em clusters RG provisionados, as consultas de data lake são executadas nos recursos computacionais do próprio cluster, em vez de em servidores dedicados do Redshift Spectrum. Da mesma forma, o Amazon Redshift sem servidor processa consultas de data lake usando a própria capacidade computacional. Para obter informações sobre o uso de consultas integradas de data lake no RG e no Redshift sem servidor, consulte Consulta do data lake.

> **Note**
>
> As consultas do Redshift Spectrum estão sujeitas a cobranças adicionais. O custo da execução de consultas de exemplo neste tutorial é nominal. Para obter mais informações sobre preços, consulte Definição de preços do Amazon Redshift Spectrum.

Pré-requisitos

Para usar o Redshift Spectrum, é necessário um cluster do Amazon Redshift e um cliente SQL conectado ao cluster, de modo que você possa executar comandos de SQL. O cluster e os arquivos de dados do Amazon S3 devem estar localizados na mesma Região da AWS.

Consulte informações sobre como criar um cluster do Amazon Redshift em Clusters provisionados do Amazon Redshift no Guia de conceitos básicos do Amazon Redshift. Para ter informações sobre

formas de se conectar a um cluster, consulte Connecting to Amazon Redshift data warehouses no Guia de conceitos básicos do Amazon Redshift.

Em alguns dos exemplos a seguir, os dados de exemplo estão na região Leste dos EUA (Norte da Virgínia) (`us-east-1`), então você precisa de um cluster que também esteja em `us-east-1`. Ou você pode usar o Amazon S3 para copiar objetos de dados dos seguintes buckets e pastas para o seu bucket na Região da AWS onde o cluster está localizado:

- `s3://redshift-downloads/tickit/spectrum/customers/*`
- `s3://redshift-downloads/tickit/spectrum/sales_partition/*`
- `s3://redshift-downloads/tickit/spectrum/sales/*`
- `s3://redshift-downloads/tickit/spectrum/salesevent/*`

Execute um comando do Amazon S3 semelhante ao apresentado a seguir para copiar dados de exemplo localizados na região Leste dos EUA (Norte da Virgínia) para sua Região da AWS. Antes de executar o comando, crie o bucket e as pastas no bucket para que correspondam ao comando copy do Amazon S3. A saída do comando copy do Amazon S3 confirma que os arquivos são copiados para o *nome-do-bucket* na Região da AWS desejada.

```
aws s3 cp s3://redshift-downloads/tickit/spectrum/ s3://bucket-name/tickit/spectrum/ --
copy-props none --recursive
```

Conceitos básicos do Amazon Redshift Spectrum usando o AWS CloudFormation

Como alternativa às etapas a seguir, é possível acessar o template Redshift Spectrum DataLake AWS CloudFormation para criar uma pilha com um bucket do Amazon S3 que você pode consultar. Para obter mais informações, consulte Iniciar a pilha do AWS CloudFormation e consultar seus dados no Amazon S3.

Conceitos básicos detalhados do Amazon Redshift Spectrum

Para começar a usar o Amazon Redshift Spectrum, siga as seguintes etapas:

- Etapa 1. Criar uma função do IAM para o Amazon Redshift
- Etapa 2: Associar uma função do IAM ao cluster

- Etapa 3: Criar um esquema e uma tabela externos
- Etapa 4: Consultar os dados no Amazon S3

Etapa 1. Criar uma função do IAM para o Amazon Redshift

O cluster precisa de autorização para acessar o catálogo de dados externo no AWS Glue ou no Amazon Athena e os arquivos de dados no Amazon S3. Para fornecer essa autorização, referencie a função do AWS Identity and Access Management (IAM) que é associada ao cluster. Para obter mais informações sobre como usar funções com o Amazon Redshift, consulte Autorização das operações COPY e UNLOAD usando as funções do IAM.

> **Note**
>
> Em determinados casos, é possível migrar o catálogo de dados do Athena para um catálogo de dados do AWS Glue. Isso poderá ser feito se o cluster estiver em uma região da AWS compatível com o AWS Glue e se você tiver tabelas externas do Redshift Spectrum no catálogo de dados do Athena. Para usar o catálogo de dados do AWS Glue com o Redshift Spectrum, talvez seja necessário alterar as políticas do IAM. Para obter mais informações, consulte Atualizar para o catálogo de dados do AWS Glue no Manual do usuário do Athena.

Ao criar uma função para o Amazon Redshift, escolha uma das seguintes abordagens:

- Se você estiver usando o Redshift Spectrum com um catálogo de dados do Athena ou um catálogo de dados do AWS Glue, siga as etapas descritas em Para criar uma função do IAM para o Amazon Redshift.
- Se você estiver usando o Redshift Spectrum com um AWS Glue Data Catalog habilitado para AWS Lake Formation, siga as etapas descritas nestes procedimentos:
  - Para criar uma função do IAM para o Amazon Redshift usando um AWS Glue Data Catalog habilitado para o AWS Lake Formation
  - Para conceder permissões SELECT na tabela para consultar no banco de dados do Lake Formation

Para criar uma função do IAM para o Amazon Redshift

1. Abra o console do IAM.

2. No painel de navegação, escolha Perfis.
3. Escolha Criar Perfil.
4. Selecione Serviço da AWS como a entidade confiável e, depois, Redshift como caso de uso.
5. Em Caso de uso para outros Serviços da AWS, selecione Redshift: personalizável e Próximo.
6. A página Add permissions policy (Adicionar política de permissões) é exibida. Escolha `AmazonS3ReadOnlyAccess` e `AWSGlueConsoleFullAccess` se estiver usando o catálogo de dados do AWS Glue. Ou escolha `AmazonAthenaFullAccess` se estiver usando o catálogo de dados do Athena. Escolha Próximo.

   > **Note**
   >
   > A política `AmazonS3ReadOnlyAccess` concede ao seu cluster acesso somente leitura a todos os buckets do Amazon S3. Para conceder acesso apenas ao bucket de dados de amostra da AWS, crie uma nova política e adicione as permissões a seguir.
   >
   > JSON

   ```
   {
       "Version":"2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Action": [
                   "s3:Get*",
                   "s3:List*"
               ],
               "Resource": "arn:aws:s3:::redshift-downloads/*"
           }
       ]
   }
   ```

7. Para Role name (Nome da função), digite um nome para sua função, por exemplo, **`myspectrum_role`**.
8. Revise as informações e escolha Create role.
9. No painel de navegação, escolha Perfis. Escolha o nome de sua nova função para visualizar o resumo e copie o Nome de recurso da Amazon (ARN) da função em sua área de transferência. Esse valor é o nome de recurso da Amazon (ARN) para a função que você acaba de criar. Esse

valor é usado quando tabelas externas são criadas para referenciar os arquivos de dados no Amazon S3.

Para criar uma função do IAM para o Amazon Redshift usando um AWS Glue Data Catalog habilitado para o AWS Lake Formation

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Policies.

   Se essa for a primeira vez que você escolhe Políticas, a página Bem-vindo às políticas gerenciadas será exibida. Escolha Começar.
3. Escolha Criar política.
4. Opte por criar a política na guia JSON.
5. Cole no documento de política JSON a seguir, que concede acesso ao catálogo de dados, mas nega as permissões de administrador para o Lake Formation.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "RedshiftPolicyForLF",
            "Effect": "Allow",
            "Action": [
                "glue:*",
                "lakeformation:GetDataAccess"
            ],
            "Resource": "*"
        }
    ]
}
```

6. Ao terminar, escolha Review (Revisar) para revisar a política. O validador de política indica se há qualquer erro de sintaxe.
7. Na página Review policy (Revisar política), em Name (Nome), insira **`myspectrum_policy`** para dar um nome à política que você está criando. Insira uma Description (Descrição) (opcional).

Revise o Resumo da política para ver as permissões que são concedidas pela política. Em seguida, escolha Criar política para salvar seu trabalho.

Depois de criar uma política, você pode conceder acesso aos usuários.

Para conceder acesso, adicione as permissões aos seus usuários, grupos ou perfis:

- Usuários e grupos no Centro de Identidade do AWS IAM:

  Crie um conjunto de permissões. Siga as instruções em Criação de um conjunto de permissões no Guia do usuário do Centro de Identidade do AWS IAM.
- Usuários gerenciados no IAM com provedor de identidades:

  Crie um perfil para a federação de identidades. Siga as instruções em Criando um perfil para um provedor de identidades de terceiros (federação) no Guia do Usuário do IAM.
- Usuários do IAM:
  - Crie um perfil que seu usuário possa assumir. Siga as instruções em Criação de um perfil para um usuário do IAM no Guia do usuário do IAM.
  - (Não recomendado) Vincule uma política diretamente a um usuário ou adicione um usuário a um grupo de usuários. Siga as instruções em Adição de permissões a um usuário (console) no Guia do usuário do IAM.

Para conceder permissões SELECT na tabela para consultar no banco de dados do Lake Formation

1. Abra o console do Lake Formation em https://console.aws.amazon.com/lakeformation/.
2. No painel de navegação, escolha Permissões do data lake e Conceder.
3. Siga as instruções em Concessão das permissões da tabela usando o nome do recurso nomeado no Guia do desenvolvedor do AWS Lake Formation. Forneça as informações a seguir:
   - Para a Função do IAM, selecione a função do IAM criada, `myspectrum_role`. Ao executar o editor de consulta do Amazon Redshift, ele usa essa função do IAM para conceder permissão aos dados.

   > **Note**
   >
   > Para conceder permissão SELECT na tabela em um catálogo de dados habilitado para o Lake Formation para consulta, faça o seguinte:

- Registre o caminho dos dados no Lake Formation.
- Conceda aos usuários permissão para esse caminho no Lake Formation.
- As tabelas criadas podem ser encontradas no caminho registrado no Lake Formation.

4. Selecione Conceder.

> **Important**
>
> Como prática recomendada, permita o acesso somente aos objetos subjacentes do Amazon S3 por meio de permissões do Lake Formation. Para impedir o acesso não aprovado, remova qualquer permissão concedida aos objetos do Amazon S3 fora do Lake Formation. Se você já tiver acessado objetos do Amazon S3 antes de configurar o Lake Formation, remova políticas do IAM ou permissões de bucket que tenham sido configuradas anteriormente. Para obter mais informações, consulte Atualizar permissões de dados do AWS Glue para o modelo do AWS Lake Formation e Permissões do Lake Formation.

Etapa 2: Associar uma função do IAM ao cluster

Agora, você tem uma função do IAM que autoriza o Amazon Redshift a acessar o catálogo de dados externo e o Amazon S3 para você. Nesse ponto, você deve associar essa função a seu cluster do Amazon Redshift.

Para associar uma função do IAM a um cluster

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. No menu de navegação, escolha Clusters e o nome do cluster que deseja atualizar.
3. Em Actions (Ações), escolha Manage IAM roles (Gerenciar funções do IAM). A página IAM roles (Funções do IAM) é exibida.
4. Escolha Enter ARN (Digitar ARN) e insira um ARN ou um perfil do IAM ou escolha um perfil do IAM na lista. Depois, escolha Add IAM role (Adicionar função do IAM) para adicioná-la à lista Attached IAM roles (Funções do IAM anexadas).
5. Escolha Done (Concluído) para associar a função do IAM ao cluster. O cluster é modificado para concluir a alteração.

Etapa 3: Criar um esquema e uma tabela externos

Crie tabelas externas em um esquema externo. O esquema externo faz referência a um banco de dados no catálogo de dados externo e fornece o ARN da função do IAM que autoriza o cluster a acessar o Amazon S3 em seu nome. É possível criar um banco de dados externo em um catálogo de dados do Amazon Athena, no AWS Glue Data Catalog ou em uma metastore do Apache Hive, como o Amazon EMR. Neste exemplo, você cria o banco de dados externo em um catálogo de dados do Amazon Athena ao criar o esquema externo do Amazon Redshift. Para obter mais informações, consulte Esquemas externos no Amazon Redshift Spectrum.

Para criar um esquema e uma tabela externos

1. Para criar um esquema externo, substitua o nome de recurso da Amazon (ARN) da função do IAM no comando a seguir pelo ARN da função que você criou na etapa 1. Execute o comando no cliente SQL.

```
create external schema myspectrum_schema
from data catalog
database 'myspectrum_db'
iam_role 'arn:aws:iam::123456789012:role/myspectrum_role'
create external database if not exists;
```

2. Para criar uma tabela externa, execute o seguinte comando CREATE EXTERNAL TABLE.

> **Note**
>
> O cluster e o bucket do Amazon S3 devem estar na mesma Região da AWS. Neste exemplo, o comando CREATE EXTERNAL TABLE, o bucket do Amazon S3 com os dados de exemplo está localizado na Leste dos EUA (Norte da Virgínia) Região da AWS. Para ver os dados de origem, baixe o arquivo `sales_ts.000`.
> Você pode modificar este exemplo para ser executado em uma diferente Região da AWS. Crie um bucket do Amazon S3 na desejada Região da AWS. Copie os dados de vendas com um comando copy do Amazon S3. Depois, atualize a opção de local do bucket no comando `CREATE EXTERNAL TABLE` de exemplo.
>
> ```
> aws s3 cp s3://redshift-downloads/tickit/spectrum/sales/ s3://bucket-name/
> tickit/spectrum/sales/ --copy-props none --recursive
> ```

A saída do comando copy do Amazon S3 confirma que o arquivo foi copiado para o *nome-do-bucket* na Região da AWS desejada.

```
copy: s3://redshift-downloads/tickit/spectrum/sales/sales_ts.000 to
 s3://bucket-name/tickit/spectrum/sales/sales_ts.000
```

```
create external table myspectrum_schema.sales(
salesid integer,
listid integer,
sellerid integer,
buyerid integer,
eventid integer,
dateid smallint,
qtysold smallint,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp)
row format delimited
fields terminated by '\t'
stored as textfile
location 's3://redshift-downloads/tickit/spectrum/sales/'
table properties ('numRows'='172000');
```

Etapa 4: Consultar os dados no Amazon S3

Após criar as tabelas externas, você pode consultá-las usando as mesmas instruções SELECT usadas nas consultas de outras tabelas do Amazon Redshift. Essas consultas com instruções SELECT incluem junções de tabelas, agregação de dados e filtragem de predicados.

Para consultar os dados no Amazon S3

1. Obtenha o número de linhas na tabela MYSPECTRUM_SCHEMA.SALES.

```
select count(*) from myspectrum_schema.sales;
```

```
count
------
```

```
172462
```

2. Mantenha suas tabelas de fatos maiores no Amazon S3 e suas tabelas de dimensões menores no Amazon Redshift como prática recomendada. Se você tiver carregado os dados de exemplo apresentados em Carregar dados de amostra, já deverá ter uma tabela chamada EVENT no banco de dados. Caso contrário, crie a tabela EVENT usando o comando a seguir.

```
create table event(
eventid integer not null distkey,
venueid smallint not null,
catid smallint not null,
dateid smallint not null sortkey,
eventname varchar(200),
starttime timestamp);
```

3. Carregue a tabela EVENT substituindo o ARN da função do IAM no comando COPY a seguir pelo ARN da função que você criou em Etapa 1. Criar uma função do IAM para o Amazon Redshift. Opcionalmente, você pode baixar e visualizar os dados de origem para o `allevents_pipe.txt` de um bucket do Amazon S3 na Região da AWS `us-east-1`.

```
copy event from 's3://redshift-downloads/tickit/allevents_pipe.txt'
iam_role 'arn:aws:iam::123456789012:role/myspectrum_role'
delimiter '|' timeformat 'YYYY-MM-DD HH:MI:SS' region 'us-east-1';
```

O exemplo a seguir une a tabela externa MYSPECTRUM_SCHEMA.SALES do Amazon S3 com a tabela local EVENT do Amazon Redshift para encontrar o total de vendas para os dez eventos principais.

```
select top 10 myspectrum_schema.sales.eventid,
 sum(myspectrum_schema.sales.pricepaid) from myspectrum_schema.sales, event
where myspectrum_schema.sales.eventid = event.eventid
and myspectrum_schema.sales.pricepaid > 30
group by myspectrum_schema.sales.eventid
order by 2 desc;
```

```
eventid | sum
--------+---------
    289 | 51846.00
   7895 | 51049.00
   1602 | 50301.00
```

```
   851 | 49956.00
  7315 | 49823.00
  6471 | 47997.00
  2118 | 47863.00
   984 | 46780.00
  7851 | 46661.00
  5638 | 46280.00
```

4. Visualize o plano da consulta para a consulta anterior. As etapas `S3 Seq Scan`, `S3 HashAggregate` e `S3 Query Scan` foram executadas com os dados do Amazon S3.

```
explain
select top 10 myspectrum_schema.sales.eventid,
 sum(myspectrum_schema.sales.pricepaid)
from myspectrum_schema.sales, event
where myspectrum_schema.sales.eventid = event.eventid
and myspectrum_schema.sales.pricepaid > 30
group by myspectrum_schema.sales.eventid
order by 2 desc;
```

```
QUERY PLAN


------------------------------------------------------------------------------
XN Limit  (cost=1001055770628.63..1001055770628.65 rows=10 width=31)


  ->  XN Merge  (cost=1001055770628.63..1001055770629.13 rows=200 width=31)


        Merge Key: sum(sales.derived_col2)


        ->  XN Network  (cost=1001055770628.63..1001055770629.13 rows=200 width=31)


              Send to leader


              ->  XN Sort  (cost=1001055770628.63..1001055770629.13 rows=200
 width=31)
```

```
                  Sort Key: sum(sales.derived_col2)


                  ->  XN HashAggregate  (cost=1055770620.49..1055770620.99
rows=200 width=31)

                        ->  XN Hash Join DS_BCAST_INNER
(cost=3119.97..1055769620.49 rows=200000 width=31)

                              Hash Cond: ("outer".derived_col1 = "inner".eventid)


                              ->  XN S3 Query Scan sales  (cost=3010.00..5010.50
rows=200000 width=31)

                                    ->  S3 HashAggregate  (cost=3010.00..3010.50
rows=200000 width=16)

                                          ->  S3 Seq Scan myspectrum_schema.sales
location:"s3://redshift-downloads/tickit/spectrum/sales" format:TEXT
(cost=0.00..2150.00 rows=172000 width=16)
                                                Filter: (pricepaid > 30.00)


                              ->  XN Hash  (cost=87.98..87.98 rows=8798 width=4)


                                    ->  XN Seq Scan on event  (cost=0.00..87.98
rows=8798 width=4)
```

Iniciar a pilha do AWS CloudFormation e consultar seus dados no Amazon S3

Depois de criar um cluster do Amazon Redshift e se conectar ao cluster, você pode instalar o template do AWS CloudFormation Redshift Spectrum DataLake e consultar seus dados.

O CloudFormation instala o modelo Redshift Spectrum Getting Started DataLake e cria uma pilha contendo:

- Uma função chamada `myspectrum_role` associada ao cluster do Redshift
- Um esquema externo chamado `myspectrum_schema`

- Uma tabela externa chamada `sales` em um bucket do Amazon S3
- Uma tabela do Redshift chamada `event` carregada com dados

Para iniciar sua pilha Redshift Spectrum Getting Started DataLake CloudFormation

1. Selecione Launch CFN stack (Iniciar pilha do CFN). O console do CloudFormation abrirá com o modelo Datalake.yml selecionado.

   Você também pode baixar e personalizar o modelo do CFN, depois abrir o console do CloudFormation (https://console.aws.amazon.com/cloudformation) e criar uma pilha com o modelo personalizado.
2. Escolha Próximo.
3. Em Parameters (Parâmetros), digite o nome do cluster do Amazon Redshift, o nome do banco de dados e o nome de usuário do banco de dados.
4. Escolha Próximo.

   São exibidas as opções de pilha.
5. Selecione Next (Próximo) para aceitar as configurações padrão.
6. Revise as informações e, em Recursos, selecione Estou ciente de que o AWS CloudFormation pode criar recursos do IAM.
7. Selecione Criar pilha.

Se ocorrer um erro durante a criação da pilha, consulte estas informações:

- Acesse a guia Events (Eventos) do CloudFormation para obter informações que podem ajudar você a solucionar o erro.
- Exclua a pilha do DataLake CloudFormation antes de tentar a realizar a operação novamente.
- Verifique se você está conectado ao banco de dados do Amazon Redshift.
- Verifique se inseriu as informações corretas do nome do cluster, nome do banco de dados e nome de usuário do banco de dados do Amazon Redshift.

Consultar seus dados no Amazon S3

Consulte tabelas externas usando as mesmas instruções SELECT usadas nas consultas de outras tabelas do Amazon Redshift. Essas consultas com instruções SELECT incluem junções de tabelas, agregação de dados e filtragem de predicados.

A consulta a seguir retorna o número de linhas na tabela externa `myspectrum_schema.sales`.

```
select count(*) from myspectrum_schema.sales;
```

```
count
------
172462
```

Unir uma tabela externa com uma tabela local

O exemplo a seguir une a tabela externa `myspectrum_schema.sales` e a tabela local `event` para encontrar as vendas totais para os 10 eventos principais.

```
select top 10 myspectrum_schema.sales.eventid, sum(myspectrum_schema.sales.pricepaid)
 from myspectrum_schema.sales, event
where myspectrum_schema.sales.eventid = event.eventid
and myspectrum_schema.sales.pricepaid > 30
group by myspectrum_schema.sales.eventid
order by 2 desc;
```

```
eventid | sum
--------+---------
    289 | 51846.00
   7895 | 51049.00
   1602 | 50301.00
    851 | 49956.00
   7315 | 49823.00
   6471 | 47997.00
   2118 | 47863.00
    984 | 46780.00
   7851 | 46661.00
   5638 | 46280.00
```

Visualizar o plano de consulta

Visualize o plano da consulta para a consulta anterior. Observe que as etapas `S3 Seq Scan`, `S3 HashAggregate` e `S3 Query Scan` foram executadas com os dados do Amazon S3.

```
explain
select top 10 myspectrum_schema.sales.eventid, sum(myspectrum_schema.sales.pricepaid)
from myspectrum_schema.sales, event
where myspectrum_schema.sales.eventid = event.eventid
and myspectrum_schema.sales.pricepaid > 30
group by myspectrum_schema.sales.eventid
order by 2 desc;
```

```
QUERY PLAN


------------------------------------------------------------------------------
XN Limit  (cost=1001055770628.63..1001055770628.65 rows=10 width=31)


  ->  XN Merge  (cost=1001055770628.63..1001055770629.13 rows=200 width=31)


        Merge Key: sum(sales.derived_col2)


        ->  XN Network  (cost=1001055770628.63..1001055770629.13 rows=200 width=31)


              Send to leader


              ->  XN Sort  (cost=1001055770628.63..1001055770629.13 rows=200 width=31)


                    Sort Key: sum(sales.derived_col2)


                    ->  XN HashAggregate  (cost=1055770620.49..1055770620.99 rows=200
 width=31)
```

```
                          ->  XN Hash Join DS_BCAST_INNER  (cost=3119.97..1055769620.49
rows=200000 width=31)

                                Hash Cond: ("outer".derived_col1 = "inner".eventid)


                                ->  XN S3 Query Scan sales  (cost=3010.00..5010.50
rows=200000 width=31)

                                      ->  S3 HashAggregate  (cost=3010.00..3010.50
rows=200000 width=16)

                                            ->  S3 Seq Scan spectrum.sales
location:"s3://redshift-downloads/tickit/spectrum/sales" format:TEXT
(cost=0.00..2150.00 rows=172000 width=16)
                                                  Filter: (pricepaid > 30.00)


                                ->  XN Hash  (cost=87.98..87.98 rows=8798 width=4)


                                      ->  XN Seq Scan on event  (cost=0.00..87.98
rows=8798 width=4)
```

Consultar Tabelas do Amazon S3 por meio do Amazon Redshift

O Amazon Redshift integra-se aos buckets de tabelas do Amazon S3, permitindo acessar recursos de tabelas do S3 usando o Amazon Redshift. Se você está apenas começando ou gerencia milhares de tabelas em seu ambiente do Iceberg, os buckets de tabelas simplificam o gerenciamento de data lakes em qualquer escala. Para obter mais informações, consulte Buckets de tabela.

Este tópico descreve como começar a usar Tabelas do Amazon S3 e o Redshift e acessar objetos Tabelas do S3 usando o Amazon Redshift.

Pré-requisitos

Antes de consultar Tabelas do S3 do Amazon Redshift, você deve integrar as tabelas do S3 ao AWS Glue Data Catalog. Para receber instruções, consulte Integrar Tabelas do Amazon S3 ao AWS Glue Data Catalog

Depois que as Tabelas do S3 são integradas ao AWS Glue Data Catalog, as entidades principais do IAM com as Tabelas do S3 e as permissões do AWS Glue IAM necessárias podem descobrir as Tabelas do S3 por meio do AWS Glue Data Catalog.

> **Note**
>
> O método 3 (awsdatacatalog montado automaticamente) tem pré-requisitos adicionais. Para obter mais detalhes, consulte the section called “Método 3: awsdatacatalog montado automaticamente”.

Consultar Tabelas do S3 por meio do Amazon Redshift

Para começar a consultar Tabelas do S3, siga estas etapas:

- Etapa 1: criar um perfil do IAM para o Amazon Redshift
- Etapa 2: anexar um perfil do IAM ao seu cluster do Amazon Redshift
- Etapa 3: consultar Tabelas do S3 por meio do Amazon Redshift

Etapa 1: criar um perfil do IAM para o Amazon Redshift

Seu cluster precisa de autorização para acessar o catálogo externo de Tabelas do S3 no AWS Glue. Para fornecer essa autorização, o Amazon Redshift usa um perfil do IAM que é anexado ao cluster. Criar um perfil do IAM com as permissões de política a seguir.

> **Note**
>
> Na política e nos exemplos abaixo, substitua `us-west-2` pela sua região da AWS e `111122223333` pelo ID da sua conta da AWS.

Para criar uma política:

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Policies.
3. Escolha Criar política.
4. Escolha a guia JSON.

5. Cole no seguinte documento de política JSON:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "GlueDataCatalogPermissions",
      "Effect": "Allow",
      "Action": [
        "glue:GetCatalog",
        "glue:GetDatabase",
        "glue:GetTable",
        "glue:GetTables",
        "glue:UpdateTable",
        "glue:DeleteTable"
      ],
      "Resource": [
        "arn:aws:glue:us-west-2:111122223333:catalog",
        "arn:aws:glue:us-west-2:111122223333:catalog/s3tablescatalog",
        "arn:aws:glue:us-west-2:111122223333:catalog/s3tablescatalog/*",
        "arn:aws:glue:us-west-2:111122223333:database/s3tablescatalog/*/*",
        "arn:aws:glue:us-west-2:111122223333:table/s3tablescatalog/*/*/*",
        "arn:aws:glue:us-west-2:111122223333:database/*",
        "arn:aws:glue:us-west-2:111122223333:table/*/*"
      ]
    },
    {
      "Sid": "S3TablesDataAccessPermissions",
      "Effect": "Allow",
      "Action": [
        "s3tables:GetTableBucket",
        "s3tables:GetNamespace",
        "s3tables:GetTable",
        "s3tables:GetTableMetadataLocation",
        "s3tables:GetTableData",
        "s3tables:ListTableBuckets",
        "s3tables:CreateTable",
        "s3tables:PutTableData",
        "s3tables:UpdateTableMetadataLocation",
        "s3tables:ListNamespaces",
        "s3tables:ListTables",
        "s3tables:DeleteTable"
      ],
```

```
      "Resource": [
        "arn:aws:s3tables:us-west-2:111122223333:bucket/*",
        "arn:aws:s3tables:us-west-2:111122223333:bucket/*/table/*"
      ]
    }
  ]
}
```

6. Selecione Revisar política.
7. Na página Revisar política, insira `GlueCatalogS3Tables_Policy` para Nome. Como opção, insira uma descrição. Revise o resumo da política e escolha Criar política.

Para criar um perfil do IAM para o Amazon Redshift:

1. Abra o console do IAM.
2. No painel de navegação, escolha Perfis.
3. Escolha Criar Perfil.
4. Selecione Serviço da AWS como a entidade confiável e, depois, Redshift como caso de uso.
5. Em Caso de uso para outros serviços da AWS, escolha Redshift: personalizável e, depois, escolha Próximo.
6. Na página Adicionar permissões, anexe a política `GlueCatalogS3Tables_Policy` que você criou anteriormente. Escolha Próximo.
7. Para Role name (Nome da função), digite um nome para sua função, por exemplo, `RedshiftS3TablesRole`.
8. Revise as informações e escolha Criar perfil.
9. No painel de navegação, escolha Perfis. Escolha o nome de seu novo perfil para visualizar o resumo e copie o ARN do perfil em sua área de transferência. Você usará esse ARN ao criar esquemas externos para namespaces de Tabelas do S3.

> **Note**
>
> Limite o escopo a recursos específicos substituindo o caractere curinga por nomes dos recursos da Amazon (ARNs) exatos.

Etapa 2: anexar um perfil do IAM ao seu cluster do Amazon Redshift

Associe o perfil do IAM configurado na Etapa 1 ao seu cluster do Amazon Redshift.

Usar o Console de Gerenciamento da AWS:

1. Faça login no Console de Gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. No menu de navegação, escolha Clusters, depois selecione o cluster que deseja atualizar.
3. Em Ações, escolha Gerenciar perfis do IAM para exibir a lista atual de perfis do IAM associados ao cluster.
4. Na página Gerenciar perfis do IAM, escolha o perfil do IAM para adição e selecione Adicionar perfil do IAM.
5. Escolha Done (Concluído) para salvar as alterações.

Usar a AWS CLI:

Execute o comando a seguir para associar um perfil do IAM a um cluster ou namespace existente. Substitua `my-redshift-cluster` ou `my-redshift-namespace` pelo identificador do cluster ou namespace e `111122223333` pelo ID da conta da AWS.

```
aws redshift modify-cluster-iam-roles \
    --cluster-identifier my-redshift-cluster \
    --add-iam-roles arn:aws:iam::111122223333:role/RedshiftS3TablesRole

-- for serverless
aws redshift-serverless update-namespace \
    --namespace-name my-redshift-namespace \
    --iam-roles "arn:aws:iam::111122223333:role/RedshiftS3TablesRole"
```

Para obter mais informações, consulte Associar perfis do IAM a clusters no Guia de gerenciamento do Amazon Redshift.

Etapa 3: consultar Tabelas do S3 por meio do Amazon Redshift

Quando você integra as Tabelas do S3 ao AWS Glue Data Catalog, o serviço cria uma estrutura de catálogo federado que associa os recursos das Tabelas do S3 a objetos do catálogo do AWS Glue:

- Um bucket de tabelas do S3 se torna um catálogo no AWS Glue Data Catalog.

- Um namespace do S3 se torna um banco de dados do AWS Glue.
- Uma tabela do S3 se torna um objeto de tabela do AWS Glue.

A integração cria a seguinte hierarquia:

- Catálogo federado:`s3tablescatalog` (criado automaticamente).
- Catálogos secundários: cada bucket de tabelas do S3 se torna um catálogo secundário em `s3tablescatalog`.
- Bancos de dados: cada namespace do S3 em um bucket de tabelas se torna um banco de dados.
- Tabelas: cada tabela do S3 em um namespace se torna uma tabela.

Por exemplo, se você tiver um bucket de tabelas do S3 denominado `analytics-bucket` com um namespace `sales` contendo uma tabela `transactions`, o caminho completo no AWS Glue Data Catalog será: `s3tablescatalog/analytics-bucket/sales/transactions`.

Crie um link de recurso

Antes de usar qualquer um dos três métodos de consulta abaixo, você deve criar um link de recurso no AWS Glue Data Catalog. Os links de recursos permitem que o Amazon Redshift faça referência aos bancos de dados de Tabelas do S3 por meio do catálogo padrão.

Usar o console do AWS Glue:

1. Abra o console do AWS Glue em https://console.aws.amazon.com/glue/.
2. No painel de navegação, escolha Bancos de dados.
3. Escolha Criar e, depois, escolha Link de recurso.
4. Na página Criar link de recurso, forneça as seguintes informações:
   - Nome do link de recurso: insira um nome para o link de recurso (por exemplo, `sales_resource_link`).
   - Banco de dados compartilhado: insira o caminho do banco de dados de Tabelas do S3 (por exemplo, `s3tablescatalog/analytics-bucket/sales`).
   - Proprietário do banco de dados compartilhado: insira o ID da conta da AWS.
   - ID do catálogo do banco de dados compartilhado: insira o ID do catálogo no formato `<account-id>:s3tablescatalog/<bucket-name>`.
5. Escolha Criar.

Usar a AWS CLI:

```
aws glue create-database \
  --region us-west-2 \
  --cli-input-json '{
        "CatalogId": "111122223333",
        "DatabaseInput": {
            "Name": "sales_resource_link",
            "TargetDatabase": {
                "CatalogId": "111122223333:s3tablescatalog/analytics-bucket",
                "DatabaseName": "sales"
            }
        }
  }'
```

Esse comando cria um link de recurso chamado `sales_resource_link` em seu catálogo de dados do AWS Glue padrão que aponta para o banco de dados `sales` no bucket de tabelas do S3 `analytics-bucket`.

Depois que os links de recursos são criados, o Amazon Redshift fornece três métodos para consultar as Tabelas do S3. Escolha o método mais adequado ao seu caso de uso.

> **Note**
>
> Para criar um link de recurso em nível de banco de dados, o administrador do Redshift deve ter a permissão `AWS Glue:CreateDatabase` no catálogo padrão e no banco de dados que está sendo criado.

Método 1: CREATE EXTERNAL SCHEMA

Use `CREATE EXTERNAL SCHEMA` para criar um esquema externo que faça referência ao seu banco de dados de Tabelas do S3. Esse método fornece controle explícito sobre a nomenclatura e a configuração do esquema.

Para receber detalhes completos da sintaxe, consulte CREATE EXTERNAL SCHEMA no Guia do desenvolvedor de banco de dados do Amazon Redshift.

Exemplo

Use o nome do banco de dados e o ID do catálogo da etapa 3. Substitua `111122223333` pelo ID de sua conta da AWS.

```
CREATE EXTERNAL SCHEMA s3tables_schema
FROM DATA CATALOG DATABASE 'sales_resource_link'
IAM_ROLE 'arn:aws:iam::111122223333:role/RedshiftS3TablesRole'
REGION 'us-west-2'
CATALOG_ID '111122223333';

SELECT * FROM s3tables_schema.transactions;
```

Método 2: CREATE DATABASE FROM ARN

Use `CREATE DATABASE` com a cláusula `FROM ARN` para criar um banco de dados federado que faça referência direta ao seu link de recurso do AWS Glue. Esse método associa automaticamente o banco de dados do AWS Glue a um banco de dados do Redshift.

Para receber detalhes completos da sintaxe, consulte CREATE DATABASE no Guia do desenvolvedor de banco de dados do Amazon Redshift.

Exemplo

Substitua `111122223333` pelo ID de sua conta da AWS.

```
CREATE DATABASE s3tables_db
FROM ARN 'arn:aws:glue:us-west-2:111122223333:database/sales_resource_link'
WITH DATA CATALOG SCHEMA analytics_schema
IAM_ROLE 'arn:aws:iam::111122223333:role/RedshiftS3TablesRole';

SELECT * FROM s3tables_db.analytics_schema.transactions;
```

Método 3: awsdatacatalog montado automaticamente

O Amazon Redshift pode montar automaticamente bancos de dados do AWS Glue Data Catalog, incluindo links de recursos de Tabelas do S3, por meio do banco de dados `awsdatacatalog`. Esse método exige que o acesso federado ao Spectrum (FAS) seja habilitado em seu cluster.

Pré-requisitos

Para usar o banco de dados `awsdatacatalog` montado automaticamente, você deve habilitar o acesso federado ao Spectrum. Isso permite que o Amazon Redshift use credenciais de identidade federadas para acessar o AWS Glue Data Catalog e fontes de dados externas.

Como habilitar o acesso federado ao Spectrum:

1. Conecte-se ao seu cluster do Redshift usando uma identidade do IAM com as seguintes permissões:
   - `redshift:GetClusterCredentialsWithIAM` (para clusters provisionados) ou `redshift-serverless:GetCredentials` (para tecnologia sem servidor).
   - `AmazonS3ReadOnlyAccess`
   - `AWSGlueConsoleFullAccess`
   - Permissões de Tabelas do S3 (conforme definido na Etapa 1)
2. Quando você se conecta com uma identidade do IAM, o Amazon Redshift cria automaticamente um usuário de banco de dados prefixado com `IAM:` (para usuários) ou `IAMR:` (para perfis).
3. Como administrador do cluster, conceda ao usuário federado permissões para acessar o esquema externo. Substitua `my_user` pelo nome de usuário ou o perfil do IAM:

```
GRANT ALL ON SCHEMA awsdatacatalog TO "IAMR:my_user";
```

Para receber informações detalhadas sobre como configurar um acesso federado, consulte Usar uma identidade federada para gerenciar o acesso do Amazon Redshift aos recursos locais e a tabelas externas do Amazon Redshift Spectrum no Guia de gerenciamento do Amazon Redshift.

Consultar Tabelas do S

Depois que o acesso federado for configurado, verifique os esquemas montados e consulte suas Tabelas do S3.

Verifique os esquemas montados:

```
SHOW SCHEMAS FROM DATABASE awsdatacatalog;
```

Consulte as Tabelas do S3 usando o nome do link do recurso da Etapa 3:

```
SELECT * FROM awsdatacatalog.sales_resource_link.transactions;
```

Políticas do IAM do Amazon Redshift Spectrum

Este tópico descreve as permissões necessárias do IAM para usar o Redshift Spectrum.

Por padrão, o Amazon Redshift Spectrum usa o AWS Glue Data Catalog em regiões da AWS compatíveis com o AWS Glue. Em outras regiões da AWS, o Redshift Spectrum usa o catálogo de dados do Athena. Seu cluster precisa de autorização para acessar o catálogo de dados externos no AWS Glue ou no Athena e os arquivos de dados no Amazon S3. Essa autorização é fornecida por meio de referência à função do AWS Identity and Access Management (IAM) que é associada ao cluster. Se você usa uma metastore do Apache Hive para gerenciar o catálogo de dados, não precisa conceder acesso ao Athena.

Você pode encadear funções para que seu cluster possa assumir outras funções não anexadas ao cluster. Para obter mais informações, consulte Encadeamento de funções do IAM no Amazon Redshift Spectrum.

O catálogo do AWS Glue que você acessa pode ser criptografado para aumentar a segurança. Se o catálogo do AWS Glue estiver criptografado, será necessária a chave do AWS KMS para que o AWS Glue acesse o catálogo de dados do AWS Glue. Para obter informações, consulte Criptografia de seu catálogo de dados do AWS Glue no Guia do desenvolvedor do AWS Glue.

Tópicos

- Permissões do Amazon S3
- Permissões entre contas do Amazon S3
- Políticas para conceder ou restringir acesso ao Redshift Spectrum
- Políticas para conceder permissões mínimas
- Encadeamento de funções do IAM no Amazon Redshift Spectrum
- Controlar o acesso ao Catálogo de dados do AWS Glue

Permissões do Amazon S3

No mínimo, seu cluster precisa dos acessos GET e LIST para o bucket do Amazon S3. Se o bucket não estiver na mesma conta da AWS que o cluster, o bucket também deverá conceder autorização para que o cluster acesse os dados. Para obter mais informações, consulte Autorização para o Amazon Redshift acessar outros serviços da AWS em seu nome.

> **Note**
>
> O bucket do Amazon S3 não pode usar uma política de bucket que restrinja o acesso somente a partir de endpoints da VPC específicos.

A política a seguir concede os acessos GET e LIST a um bucket do Amazon S3. A política permite o acesso aos buckets do Amazon S3 para o Redshift Spectrum, bem como às operações COPY.

JSON

```
{
 "Version":"2012-10-17",
 "Statement": [{
  "Effect": "Allow",
  "Action": ["s3:Get*", "s3:List*"],
  "Resource": "*"
 }]
}
```

A política a seguir concede os acessos GET e LIST ao bucket do Amazon S3 denominado `amzn-s3-demo-bucket`.

JSON

```
{
 "Version":"2012-10-17",
 "Statement": [{
  "Effect": "Allow",
  "Action": ["s3:Get*", "s3:List*"],
  "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
 }]
}
```

Permissões entre contas do Amazon S3

Para conceder ao Redshift Spectrum permissão para acessar dados em um bucket do Amazon S3 que pertence a outra conta da AWS, adicione a política a seguir ao bucket do Amazon S3. Para obter mais informações, consulte Conceder permissões de bucket entre contas.

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Example permissions",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/spectrumrole"
            },
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetObject",
                "s3:ListMultipartUploadParts",
                "s3:ListBucket",
                "s3:ListBucketMultipartUploads"
            ],
            "Resource": [
                "arn:aws:s3:::bucketname",
                "arn:aws:s3:::bucketname/*"
            ]
        }
    ]
}
```

Políticas para conceder ou restringir acesso ao Redshift Spectrum

A política a seguir permite acesso aos buckets do Amazon S3 para o Amazon Redshift. Ao adicionar essa política a um perfil do IAM, recomendamos usar um perfil do IAM que seja utilizado somente para o Amazon Redshift. Também recomendamos que você não reutilize perfis existentes nem adicione permissões extras a esse perfil do IAM.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListBucketVersions",
                "s3:ListBucket"
```

```
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        }
    ]
}
```

Políticas para conceder permissões mínimas

A política a seguir concede as permissões mínimas necessárias para usar o Redshift Spectrum com o Amazon S3, o AWS Glue e o Athena.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket",
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::bucketname",
                "arn:aws:s3:::bucketname/folder1/folder2/*"
             ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "glue:CreateDatabase",
                "glue:DeleteDatabase",
                "glue:GetDatabase",
                "glue:GetDatabases",
                "glue:UpdateDatabase",
                "glue:CreateTable",
                "glue:DeleteTable",
```

```
                "glue:BatchDeleteTable",
                "glue:UpdateTable",
                "glue:GetTable",
                "glue:GetTables",
                "glue:BatchCreatePartition",
                "glue:CreatePartition",
                "glue:DeletePartition",
                "glue:BatchDeletePartition",
                "glue:UpdatePartition",
                "glue:GetPartition",
                "glue:GetPartitions",
                "glue:BatchGetPartition"
            ],
            "Resource": "*"
        }
    ]
}
```

Se você utilizar o Athena para o catálogo de dados em vez do AWS Glue, a política exigirá acesso total ao Athena. A política a seguir concede acesso aos recursos do Athena. Se seu banco de dados externo for uma metastore do Hive, você não precisará de acesso ao Athena.

JSON

```
{
 "Version":"2012-10-17",
 "Statement": [{
  "Effect": "Allow",
  "Action": ["athena:*"],
  "Resource": ["*"]
 }]
}
```

Encadeamento de funções do IAM no Amazon Redshift Spectrum

Ao anexar uma função ao cluster, seu cluster poderá assumir essa função para acessar o Amazon S3, o Athena e o AWS Glue em seu nome. Se uma função anexada ao cluster não tiver acesso aos recursos necessários, você poderá encadear outra função, possivelmente pertencente a outra conta. O cluster assumirá a função encadeada temporariamente para acessar os dados. Você também

pode conceder acesso entre contas com o encadeamento de funções. Você pode encadear um máximo de 10 funções. Cada função em cadeia assume a próxima função na cadeia, até que o cluster assuma a função no final da cadeia.

Para encadear funções, você estabelece uma relação de confiança entre as funções. Uma função que assume uma outra função deve ter uma política de permissões que permita que ela assuma a função especificada. Por sua vez, a função que passa permissões deve ter uma política de confiança que permita que ela passe suas permissões para outra função. Para obter mais informações, consulte Encadeamento de funções do IAM no Amazon Redshift.

Quando você executa o comando do CREATE EXTERNAL SCHEMA, pode encadear funções incluindo uma lista separada por vírgulas de ARNs de funções.

> **Note**
>
> A lista de funções encadeadas não deve incluir espaços.

No exemplo a seguir, `MyRedshiftRole` está anexada ao cluster. `MyRedshiftRole` assume a função `AcmeData`, que pertence à conta `111122223333`.

```
create external schema acme from data catalog
database 'acmedb' region 'us-west-2'
iam_role 'arn:aws:iam::123456789012:role/MyRedshiftRole,arn:aws:iam::111122223333:role/
AcmeData';
```

Controlar o acesso ao Catálogo de dados do AWS Glue

Se usar o AWS Glue em seu catálogo de dados, você poderá aplicar um controle de acesso refinado ao catálogo de dados do AWS Glue com sua política do IAM. Por exemplo, você pode querer expor apenas alguns bancos de dados e tabelas para uma função específica do IAM.

As seções a seguir descrevem as políticas do IAM para vários níveis de acesso aos dados armazenados no AWS Glue Data Catalog.

Tópicos

- Política para operações de banco de dados
- Política para operações de tabela
- Política para operações de partição

Política para operações de banco de dados

Se quiser conceder aos usuários permissões para exibir e criar um banco de dados, eles precisarão de direitos de acesso ao banco de dados e ao AWS Glue Data Catalog.

A consulta de exemplo a seguir cria um banco de dados.

```
CREATE EXTERNAL SCHEMA example_db
FROM DATA CATALOG DATABASE 'example_db' region 'us-west-2'
IAM_ROLE 'arn:aws:iam::redshift-account:role/spectrumrole'
CREATE EXTERNAL DATABASE IF NOT EXISTS
```

A seguinte política do IAM fornece as permissões mínimas necessárias para criar um banco de dados.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetDatabase",
                "glue:CreateDatabase"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:111122223333:database/example_db",
                "arn:aws:glue:us-west-2:111122223333:catalog"
            ]
        }
    ]
}
```

A consulta de exemplo a seguir lista os bancos de dados atuais.

```
SELECT * FROM SVV_EXTERNAL_DATABASES WHERE
```

```
databasename = 'example_db1' or databasename = 'example_db2';
```

A seguinte política do IAM fornece as permissões mínimas necessárias para listar os bancos de dados atuais.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetDatabases"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:111122223333:database/example_db1",
                "arn:aws:glue:us-west-2:111122223333:database/example_db2",
                "arn:aws:glue:us-west-2:111122223333:catalog"
            ]
        }
    ]
}
```

Política para operações de tabela

Para conceder aos usuários permissões para exibir, criar, descartar, alterar ou executar outras ações em tabelas, eles precisarão de vários tipos de acesso. Eles precisam de acesso às próprias tabelas, aos bancos de dados ao qual elas pertencem e ao catálogo.

A consulta de exemplo a seguir cria uma tabela externa.

```
CREATE EXTERNAL TABLE example_db.example_tbl0(
    col0 INT,
    col1 VARCHAR(255)
) PARTITIONED BY (part INT) STORED AS TEXTFILE
LOCATION 's3://test/s3/location/';
```

A seguinte política do IAM fornece as permissões mínimas necessárias para criar uma tabela externa.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:CreateTable"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:111122223333:catalog",
                "arn:aws:glue:us-west-2:111122223333:database/example_db",
                "arn:aws:glue:us-west-2:111122223333:table/example_db/
example_tbl0"
            ]
        }
    ]
}
```

O exemplo a seguir consulta cada lista das tabelas externas atuais.

```
SELECT * FROM svv_external_tables
WHERE tablename = 'example_tbl0' OR
tablename = 'example_tbl1';
```

```
SELECT * FROM svv_external_columns
WHERE tablename = 'example_tbl0' OR
tablename = 'example_tbl1';
```

```
SELECT parameters FROM svv_external_tables
WHERE tablename = 'example_tbl0' OR
```

```
tablename = 'example_tbl1';
```

A seguinte política do IAM fornece as permissões mínimas necessárias para listar as tabelas externas atuais.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetTables"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:111122223333:catalog",
                "arn:aws:glue:us-west-2:111122223333:database/example_db",
                "arn:aws:glue:us-west-2:111122223333:table/example_db/
example_tbl0",
                "arn:aws:glue:us-west-2:111122223333:table/example_db/
example_tbl1"
            ]
        }
    ]
}
```

A consulta de exemplo a seguir altera uma tabela existente.

```
ALTER TABLE example_db.example_tbl0
SET TABLE PROPERTIES ('numRows' = '100');
```

A seguinte política do IAM fornece as permissões mínimas necessárias para alterar uma tabela existente.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetTable",
                "glue:UpdateTable"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:111122223333:catalog",
                "arn:aws:glue:us-west-2:111122223333:database/example_db",
                "arn:aws:glue:us-west-2:111122223333:table/example_db/
example_tbl0"
            ]
        }
    ]
}
```

A consulta de exemplo a seguir descarta uma tabela existente.

```
DROP TABLE example_db.example_tbl0;
```

A seguinte política do IAM fornece as permissões mínimas necessárias para descartar uma tabela existente.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:DeleteTable"
            ],
```

```
            "Resource": [
                "arn:aws:glue:us-west-2:111122223333:catalog",
                "arn:aws:glue:us-west-2:111122223333:database/example_db",
                "arn:aws:glue:us-west-2:111122223333:table/example_db/
example_tbl0"
            ]
        }
    ]
}
```

Política para operações de partição

Se quiser conceder aos usuários permissões para executar operações no nível da partição (exibir, criar, soltar, alterar e assim por diante), eles precisarão de permissões para as tabelas às quais as partições pertencem. Eles também precisarão de permissões para os bancos de dados relacionados e o catálogo de dados do AWS Glue.

A consulta de exemplo a seguir cria uma partição.

```
ALTER TABLE example_db.example_tbl0
ADD PARTITION (part=0) LOCATION 's3://test/s3/location/part=0/';
ALTER TABLE example_db.example_t
ADD PARTITION (part=1) LOCATION 's3://test/s3/location/part=1/';
```

A seguinte política do IAM fornece as permissões mínimas necessárias para criar uma partição.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetTable",
                "glue:BatchCreatePartition"
            ],
            "Resource": [
```

```
                "arn:aws:glue:us-west-2:111122223333:catalog",
                "arn:aws:glue:us-west-2:111122223333:database/example_db",
                "arn:aws:glue:us-west-2:111122223333:table/example_db/
example_tbl0"
            ]
        }
    ]
}
```

A consulta de exemplo a seguir lista as partições atuais.

```
SELECT * FROM svv_external_partitions
WHERE schemname = 'example_db' AND
tablename = 'example_tbl0'
```

A seguinte política do IAM fornece as permissões mínimas necessárias para listar as partições atuais.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetPartitions",
                "glue:GetTables",
                "glue:GetTable"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:111122223333:catalog",
                "arn:aws:glue:us-west-2:111122223333:database/example_db",
                "arn:aws:glue:us-west-2:111122223333:table/example_db/
example_tbl0"
            ]
        }
    ]
```

```
}
```

A consulta de exemplo a seguir altera uma partição existente.

```
ALTER TABLE example_db.example_tbl0 PARTITION(part='0')
SET LOCATION 's3://test/s3/new/location/part=0/';
```

A seguinte política do IAM fornece as permissões mínimas necessárias para alterar uma partição existente.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetPartition",
                "glue:UpdatePartition"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:111122223333:catalog",
                "arn:aws:glue:us-west-2:111122223333:database/example_db",
                "arn:aws:glue:us-west-2:111122223333:table/example_db/
example_tbl0"
            ]
        }
    ]
}
```

A consulta de exemplo a seguir descarta uma partição existente.

```
ALTER TABLE example_db.example_tbl0 DROP PARTITION(part='0');
```

A seguinte política do IAM fornece as permissões mínimas necessárias para destacar uma partição existente.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:DeletePartition"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:111122223333:catalog",
                "arn:aws:glue:us-west-2:111122223333:database/example_db",
                "arn:aws:glue:us-west-2:111122223333:table/example_db/
example_tbl0"
            ]
        }
    ]
}
```

Redshift Spectrum e AWS Lake Formation

Este tópico descreve como usar o Redshift Spectrum com o Lake Formation. O Lake Formation é um serviço para compartilhar dados de analytics.

É possível usar o AWS Lake Formation para definir e impor de forma centralizada políticas de acesso no nível do banco de dados, da tabela e da coluna para os dados armazenados no Amazon S3. Depois que os dados forem registrados com um AWS Glue Data Catalog habilitado com o Lake Formation, será possível consultá-los usando vários serviços, incluindo o Redshift Spectrum.

O Lake Formation fornece a segurança e a governança do catálogo de dados. No Lake Formation, é possível conceder e revogar permissões para os objetos do catálogo de dados, como bancos de dados, tabelas, colunas e armazenamento subjacente do Amazon S3.

> **Important**
>
> Você só pode usar o Redshift Spectrum com um catálogo de dados habilitado para Lake Formation nas regiões da AWS onde o Lake Formation está disponível. Para obter uma lista de regiões disponíveis, consulte Endpoints e cotas do AWS Lake Formation na Referência geral da AWS.

Usando o Redshift Spectrum com Lake Formation, você pode fazer o seguinte:

- Use o Lake Formation como um local central para conceder e revogar permissões e políticas de controle de acesso em todos os dados do data lake. O Lake Formation fornece uma hierarquia de permissões para controlar o acesso aos bancos de dados e tabelas em um catálogo de dados. Para obter mais informações, consulte "Overview of Lake Formation permissions" (Visão geral das permissões do Lake Formation) no Guia do desenvolvedor do AWS Lake Formation.
- Crie tabelas externas e execute consultas em dados no data lake. Antes que os usuários da conta possam executar consultas, um administrador da conta do data lake registra os caminhos existentes do Amazon S3 que contêm dados da fonte com o Lake Formation. O administrador também cria tabelas e concede permissões aos usuários. O acesso pode ser concedido em bancos de dados, tabelas ou colunas. O administrador pode usar filtros de dados no Lake Formation para conceder controle de acesso granular sobre seus dados confidenciais armazenados no Amazon S3. Para obter mais informações, consulte Usar filtros de dados para segurança em nível de linha e de célula.

  Depois que os dados forem registrados no catálogo de dados, sempre que os usuários tentarem executar consultas, o Lake Formation verificará o acesso à tabela para essa entidade principal específica. O Lake Formation fornece credenciais temporárias para o Redshift Spectrum e a consulta é executada.
- Execute consultas do Redshift Spectrum em um AWS Glue Data Catalog montado automaticamente usando credenciais do IAM obtidas com `GetCredentials` ou `GetClusterCredentials` e gerencie as permissões do Lake Formation por usuário do banco de dados (IAMR:username ou IAM:username).

Quando você usa o Redshift Spectrum com um Catálogo de Dados habilitado para o Lake Formation, uma das seguintes opções deve estar em vigor:

- Um perfil do IAM associado ao cluster que tem permissão para o Catálogo de Dados.

- Uma identidade federada do IAM configurada para gerenciar o acesso a recursos externos. Para obter mais informações, consulte Usar uma identidade federada para gerenciar o acesso do Amazon Redshift aos recursos locais e às tabelas externas do Amazon Redshift Spectrum.

> **Important**
>
> Não é possível encadear funções do IAM ao usar o Redshift Spectrum com um catálogo de dados habilitado para o Lake Formation.

Para saber mais sobre as etapas necessárias para configurar o AWS Lake Formation a ser usado com o Redshift Spectrum, consulte Tutorial: Creating a data lake from a JDBC source in Lake Formation no Guia do desenvolvedor do AWS Lake Formation. Especificamente, confira "Query the data in the data lake using Amazon Redshift Spectrum" (Consultar os dados no data lake usando o Amazon Redshift Spectrum) para obter detalhes sobre a integração com o Redshift Spectrum. Os dados e recursos da AWS usados neste tópico dependem das etapas anteriores do tutorial.

Usar filtros de dados para segurança em nível de linha e de célula

Você pode definir filtros de dados no AWS Lake Formation para controlar o acesso em nível de linha e célula de consultas do Redshift Spectrum aos dados definidos em seu catálogo de dados. Para configurar isso, execute as seguintes tarefas:

- Crie um filtro de dados no Lake Formation com as seguintes informações:
  - Uma especificação de coluna com uma lista de colunas a serem incluídas ou excluídas dos resultados da consulta.
  - Uma expressão de filtro de linha que especifica as linhas a serem incluídas nos resultados da consulta.

  Para obter mais informações sobre como criar um filtro de dados, consulte "Data filters in Lake Formation" (Filtros de dados no Lake Formation) no Guia do desenvolvedor do AWS Lake Formation.
- Crie uma tabela externa no Amazon Redshift que faça referência a uma tabela em seu catálogo de dados habilitado para o Lake Formation. Para obter detalhes sobre como consultar uma tabela do Lake Formation usando o Redshift Spectrum, confira "Query the data in the data lake using Amazon Redshift Spectrum" (Consultar os dados no data lake usando o Amazon Redshift Spectrum) no Guia do desenvolvedor do AWS Lake Formation.

Depois que a tabela for definida no Amazon Redshift, você poderá consultar a tabela do Lake Formation e acessar somente as linhas e colunas permitidas pelo filtro de dados.

Para obter um guia detalhado sobre como configurar a segurança por linha e célula no Lake Formation, depois consultar usando o Redshift Spectrum, consulte Usar o Amazon Redshift Spectrum com políticas de segurança por linha e célula definidas no AWS Lake Formation.

Arquivos de dados para consultas no Amazon Redshift Spectrum

Esta seção descreve como criar arquivos de dados no Amazon S3 em um formato compatível com o Redshift Spectrum.

Os arquivos de dados que você usa para consultas no Amazon Redshift Spectrum geralmente são dos mesmos tipos dos arquivos usados para outras aplicações. Por exemplo, os mesmos tipos de arquivo são usados com o Amazon Athena, o Amazon EMR e o Quick. Você pode consultar os dados em seu formato original diretamente no Amazon S3. Para isso, os arquivos de dados devem estar em um formato compatível com o Redshift Spectrum e localizados em um bucket do Amazon S3 que possa ser acessado pelo cluster.

O bucket do Amazon S3 com os arquivos de dados deve estar na mesma região da AWS do cluster do Amazon Redshift. Para obter informações sobre as regiões da AWS compatíveis, consulte Regiões do Amazon Redshift Spectrum.

Formatos de dados do Redshift Spectrum

O Redshift Spectrum é compatível com os formatos de dados estruturados e semiestruturados a seguir.

Formato do arquivo	Colunar	Suporte a leituras paralelas	Unidade dividida
Parquet	Sim	Sim	Grupo de linhas
ORC	Sim	Sim	Stripe
RCFile	Sim	Sim	Grupo de linhas
TextFile	Não	Sim	Linha

Formato do arquivo	Colunar	Suporte a leituras paralelas	Unidade dividida
SequenceFile	Não	Sim	Linha ou bloco
RegexSerde	Não	Sim	Linha
OpenCSV	Não	Sim	Linha
AVRO	Não	Sim	Bloco
Ion	Não	Não	N/D
JSON	Não	Não	N/D

Na tabela anterior, os cabeçalhos indicam o seguinte:

- Colunar: se o formato do arquivo armazena dados fisicamente em uma estrutura orientada em colunas, em vez de uma estrutura orientada em linhas.
- Suporte a leituras paralelas: se o formato de arquivo oferece suporte à leitura de blocos individuais dentro do arquivo. A leitura de blocos individuais permite o processamento distribuído de um arquivo em várias solicitações independentes do Redshift Spectrum, em vez de ter de ler o arquivo completo em uma única solicitação.
- Unidade dividida: para formatos de arquivo que podem ser lidos em paralelo, a unidade dividida é o menor bloco de dados que uma única solicitação do Redshift Spectrum consegue processar.

> **Note**
>
> Os valores de data e hora nos arquivos de texto devem estar no formato `yyyy-MM-dd HH:mm:ss.SSSSSS`, como mostra o seguinte valor de data e hora de exemplo: `2017-05-01 11:30:59.000000`.

Recomendamos usar um formato de arquivo colunar para o armazenamento, como o Apache Parquet. O Amazon Redshift é compatível com o formato de arquivo de dados Apache Parquet v1. Com um formato de arquivo colunar para o armazenamento, é possível minimizar a transferência de dados do Amazon S3 selecionando apenas as colunas necessárias.

Tipos de compactação do Redshift Spectrum

Para reduzir o espaço de armazenamento, melhorar a performance e minimizar os custos, é altamente recomendável compactar os arquivos de dados. O Redshift Spectrum reconhece os tipos de compactação de arquivos com base na extensão de arquivo.

O Redshift Spectrum é compatível com os tipos de compactação e extensões a seguir.

Algoritmo de compactação	Extensão de arquivo	Suporte a leituras paralelas
Gzip	.gz	Não
Bzip2	.bz2	Sim
Snappy	.snappy	Não

O Amazon Redshift também permite a compactação Zstandard (zstd) para arquivos Apache Parquet e tabelas Apache Iceberg.

Você pode aplicar compactação em diferentes níveis. O mais comum é compactar um arquivo inteiro ou blocos individuais dentro de um arquivo. A compactação de formatos colunares no nível do arquivo não gera benefícios de performance.

Para que o Redshift Spectrum consiga ler um arquivo em paralelo, as seguintes afirmações devem ser verdadeiras:

- O formato de arquivo oferece suporte a leituras paralelas.
- A compactação no nível do arquivo, se houver, oferece suporte a leituras paralelas.

Não importa se as unidades divididas individuais dentro de um arquivo são compactadas usando um algoritmo de compactação que pode ser lido em paralelo, pois cada unidade dividida é processada por uma única solicitação do Redshift Spectrum. Um exemplo disso são os arquivos Parquet compactados por Snappy. Grupos de linhas individuais dentro do arquivo Parquet são compactados usando Snappy, mas a estrutura de nível superior do arquivo permanece descompactada. Nesse caso, o arquivo pode ser lido em paralelo, pois cada solicitação do Redshift Spectrum consegue ler e processar grupos de linhas individuais do Amazon S3.

Encriptação do Redshift Spectrum

O Redshift Spectrum descriptografa os arquivos de dados de forma transparente usando as seguintes opções de criptografia:

- Criptografia no lado do servidor (SSE-S3) usando uma chave de criptografia AES-256 gerenciada pelo Amazon S3.
- Criptografia no lado do servidor com chaves gerenciadas pelo AWS Key Management Service (SSE-KMS).

O Redshift Spectrum não é compatível com criptografia no lado do cliente no Amazon S3. Para obter mais informações sobre criptografia no lado do servidor, consulte Proteger dados usando criptografia no lado do servidor no Guia do usuário do Amazon Simple Storage Service.

O Amazon Redshift usa o processamento paralelo maciço (MPP) para executar com rapidez consultas complexas que operam com grandes quantidades de dados. O Redshift Spectrum amplia o mesmo princípio para as consultas com dados externos, usando várias instâncias do Redshift Spectrum para fazer a varredura de arquivos. Coloque os arquivos em uma pasta separada para cada tabela.

Você pode otimizar os dados para o processamento em paralelo executando as seguintes ações:

- Se o formato de arquivo ou a compactação não oferecer suporte à leitura em paralelo, divida arquivos grandes em vários arquivos menores. Recomendamos o uso de tamanhos de arquivo entre 64 MB e 1 GB.
- Mantenha todos os arquivos com aproximadamente o mesmo tamanho. Se alguns arquivos forem muito maiores do que os outros, o Redshift Spectrum não poderá distribuir o workload uniformemente.

Esquemas externos no Amazon Redshift Spectrum

Este tópico descreve como criar e usar esquemas externos com o Redshift Spectrum. Esquemas externos são coleções de tabelas que você usa como referência para acessar dados fora do cluster do Amazon Redshift. Essas tabelas contêm metadados sobre os dados externos que o Redshift Spectrum lê.

Todas as tabelas externas devem ser criadas em um esquema externo, o qual você cria usando a instrução CREATE EXTERNAL SCHEMA.

> **Note**
>
> Alguns aplicativos usam os termos banco de dados e esquema de maneira intercambiável. O Amazon Redshift usa o termo “esquema”.

Um esquema externo do Amazon Redshift faz referência a um banco de dados externo em um catálogo de dados externo. Você pode criar o banco de dados externo no Amazon Redshift, no Amazon Athena, no AWS Glue Data Catalog ou em uma metastore do Apache Hive, como o Amazon EMR. Quando você cria um banco de dados externo no Amazon Redshift, o banco de dados reside no catálogo de dados do Athena. Para criar um banco de dados em uma metastore do Hive, é necessário criar o banco de dados na aplicação do Hive.

O Amazon Redshift precisa de autorização para acessar o catálogo de dados no Athena e os arquivos de dados no Amazon S3 em seu nome. Para fornecer essa autorização, crie primeiro uma função do AWS Identity and Access Management (IAM). Em seguida, anexe a função ao seu cluster e forneça o nome do recurso da Amazon (ARN) para a função na instrução `CREATE EXTERNAL SCHEMA` do Amazon Redshift. Para obter mais informações sobre a autorização da , consulte Políticas do IAM do Amazon Redshift Spectrum.

Para criar um banco de dados externo ao mesmo tempo que o esquema externo é criado, especifique `FROM DATA CATALOG` e inclua a cláusula `CREATE EXTERNAL DATABASE` na instrução `CREATE EXTERNAL SCHEMA`.

O exemplo a seguir cria um esquema externo denominado `spectrum_schema` usando o banco de dados externo `spectrum_db`.

```
create external schema spectrum_schema from data catalog
database 'spectrum_db'
iam_role 'arn:aws:iam::123456789012:role/MySpectrumRole'
create external database if not exists;
```

Se você gerencia o catálogo de dados usando o Athena, especifique o nome do banco de dados do Athena e a região da AWS na qual o catálogo de dados do Athena está localizado.

O exemplo a seguir cria um esquema externo usando o banco de dados padrão `sampledb` no catálogo de dados do Athena.

```
create external schema athena_schema from data catalog
database 'sampledb'
```

```
iam_role 'arn:aws:iam::123456789012:role/MySpectrumRole'
region 'us-east-2';
```

> **Note**
>
> O parâmetro `region` faz referência à região da AWS em que o catálogo de dados do Athena está localizado, e não à localização dos arquivos de dados no Amazon S3.

Se você gerencia o catálogo de dados usando uma metastore do Hive, como o Amazon EMR, seus grupos de segurança devem ser configurados para permitir o tráfego entre os clusters.

Na instrução CREATE EXTERNAL SCHEMA, especifique `FROM HIVE METASTORE` e inclua o URI e o número da porta da metastore. O exemplo a seguir cria um esquema externo usando um banco de dados do metastore do Hive denominado `hive_db`.

```
create external schema hive_schema
from hive metastore
database 'hive_db'
uri '172.10.10.10' port 99
iam_role 'arn:aws:iam::123456789012:role/MySpectrumRole'
```

Para visualizar os esquemas externos do seu cluster, consulte a tabela do catálogo PG_EXTERNAL_SCHEMA ou a exibição SVV_EXTERNAL_SCHEMAS. O exemplo a seguir consulta a SVV_EXTERNAL_SCHEMAS, que une as tabelas PG_EXTERNAL_SCHEMA e PG_NAMESPACE.

```
select * from svv_external_schemas
```

Para ver a sintaxe completa do comando e os exemplos, consulte CREATE EXTERNAL SCHEMA.

Trabalhar com catálogos externos no Amazon Redshift Spectrum

Os metadados dos bancos de dados externos e as tabelas externas do Amazon Redshift Spectrum são armazenados em um catálogo de dados externos. Por padrão, os metadados do Redshift Spectrum são armazenados em um catálogo de dados do Athena. É possível visualizar e gerenciar os bancos de dados e as tabelas do Redshift Spectrum no console do Athena.

Também é possível criar e gerenciar bancos de dados externos e tabelas externas usando a linguagem de definição de dados (DDL) do Hive por meio do Athena ou de uma metastore do Hive, como o Amazon EMR.

> **Note**
>
> Recomendamos o uso do Amazon Redshift para criar e gerenciar os bancos de dados externos e as tabelas externas no Redshift Spectrum.

Visualizar bancos de dados do Redshift Spectrum no Athena e no AWS Glue

Você pode criar um banco de dados externo incluindo a cláusula CREATE EXTERNAL DATABASE IF NOT EXISTS como parte da instrução CREATE EXTERNAL SCHEMA. Nesses casos, os metadados do banco de dados externo é armazenado em seu catálogo de dados. Os metadados das tabelas externas criadas que são qualificadas pelo esquema externo também são armazenados no catálogo de dados do .

O Athena e o AWS Glue mantêm um catálogo de dados para cada Região da AWS compatível. Para exibir os metadados da tabela, faça login no Athena ou no console do AWS Glue. No Athena, escolha Data sources (Fontes de dados) e seu AWS Glue e visualize os detalhes do banco de dados. No AWS Glue, escolha Databases (Bancos de dados) e seu banco de dados externo e visualize os respectivos detalhes.

Se você cria e gerencia suas tabelas externas usando o Athena, registre o banco de dados com CREATE EXTERNAL SCHEMA. Por exemplo, o comando a seguir registra o banco de dados do Athena denominado `sampledb`.

```
create external schema athena_sample
from data catalog
database 'sampledb'
iam_role 'arn:aws:iam::123456789012:role/mySpectrumRole'
region 'us-east-1';
```

Ao consultar a exibição de sistema SVV_EXTERNAL_TABLES, você vê as tabelas no banco de dados `sampledb` do Athena, além das tabelas que criou no Amazon Redshift.

```
select * from svv_external_tables;
```

```
schemaname    | tablename        | location

--------------+------------------
+-----------------------------------------------------
```

```
athena_sample | elb_logs          | s3://athena-examples/elb/plaintext
athena_sample | lineitem_1t_csv   | s3://myspectrum/tpch/1000/lineitem_csv

athena_sample | lineitem_1t_part | s3://myspectrum/tpch/1000/lineitem_partition

spectrum      | sales            | s3://redshift-downloads/tickit/spectrum/sales

spectrum      | sales_part       | s3://redshift-downloads/tickit/spectrum/sales_part
```

Registro de um banco de dados da metastore do Apache Hive

Ao criar tabelas externas em uma metastore do Apache Hive, você pode usar CREATE EXTERNAL SCHEMA para registrar essas tabelas no Redshift Spectrum.

Na instrução CREATE EXTERNAL SCHEMA, especifique a cláusula FROM HIVE METASTORE e forneça o URI e o número da porta da metastore do Hive. A função do IAM deve incluir uma permissão para acessar o Amazon S3, mas não precisa de nenhuma permissão para o Athena. O exemplo a seguir registra uma metastore do Hive.

```
create external schema if not exists hive_schema
from hive metastore
database 'hive_database'
uri 'ip-10-0-111-111.us-west-2.compute.internal' port 9083
iam_role 'arn:aws:iam::123456789012:role/mySpectrumRole';
```

Ativar o cluster do Amazon Redshift para acessar o cluster do Amazon EMR

Se a metastore do Hive estiver no Amazon EMR, você deverá conceder ao cluster do Amazon Redshift acesso ao cluster do Amazon EMR. Para isso, crie um grupo de segurança do Amazon EC2. Depois, permita todo o tráfego de entrada para esse grupo de segurança do EC2 recebido do grupo de segurança do cluster do Amazon Redshift e do grupo de segurança do cluster do Amazon EMR. Em seguida, adicione a segurança do EC2 a ambos os clusters do Amazon Redshift e do Amazon EMR.

Visualizar o nome do grupo de segurança no cluster do Amazon Redshift

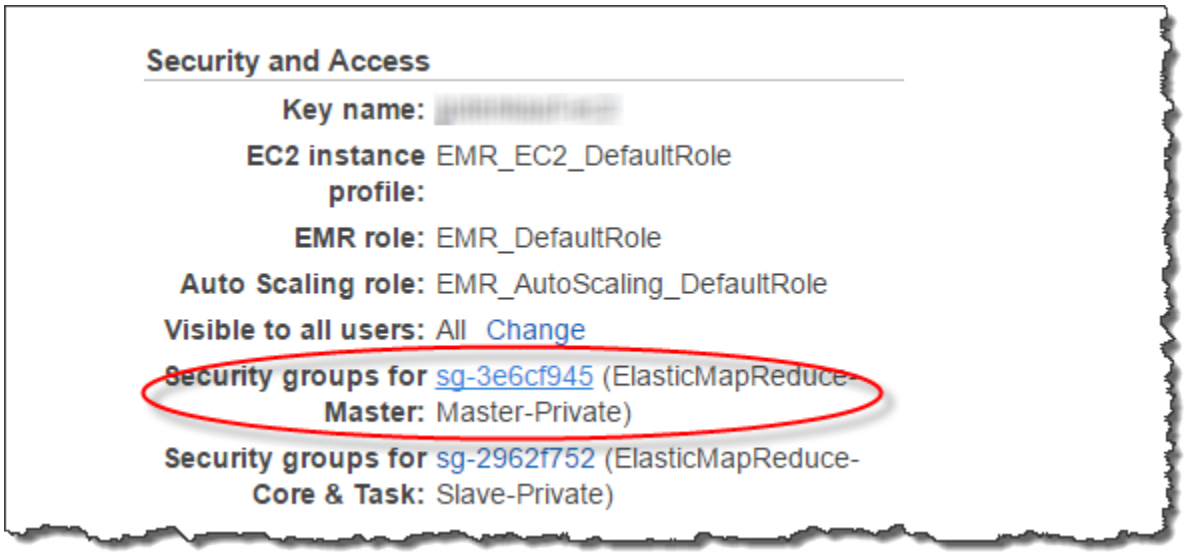
Para exibir o grupo de segurança, faça o seguinte:

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.

2. No menu de navegação, escolha Clusters. Em seguida, escolha o cluster na lista para abrir os respectivos detalhes.
3. Escolha Properties (Propriedades) e visualize a seção Network and security (Rede e segurança).
4. Encontre seu grupo de segurança em VPC security group (Grupo de segurança da VPC) e anote.

Visualizar o nome do grupo de segurança do nó principal do Amazon EMR

1. Abra o cluster do Amazon EMR. Para obter mais informações, consulte Usar configurações de segurança para definir a segurança do cluster no Guia de gerenciamento do Amazon EMR.
2. Em Security and access (Segurança e acesso), anote o nome do grupo de segurança do nó principal do Amazon EMR.



Para criar ou modificar um grupo de segurança do Amazon EC2 para permitir a conexão entre o Amazon Redshift e o Amazon EMR

1. No painel do Amazon EC2, escolha Security Groups (Grupos de segurança). Consulte mais informações em Regras de grupos de segurança no Guia do usuário do Amazon EC2.
2. Escolha Criar grupo de segurança.
3. Se estiver usando a VPC, escolha a VPC na qual os clusters do Amazon Redshift e do Amazon EMR se encontram.
4. Adicione uma regra de entrada.

1. Em Type (Tipo), escolha Custom TCP (TCP personalizada).
2. Em Source, escolha Custom.
3. Insira o nome de seu grupo de segurança do Amazon Redshift.

5. Adicione uma outra regra de entrada.

1. Para Type, escolha TCP.
2. Em Port Range (Intervalo de portas), insira 9083.

> **Note**
>
> A porta padrão para um HMS no EMR é 9083. Se o HMS usa uma porta diferente, especifique essa porta na regra de entrada e na definição do esquema externo.

3. Em Source, escolha Custom.

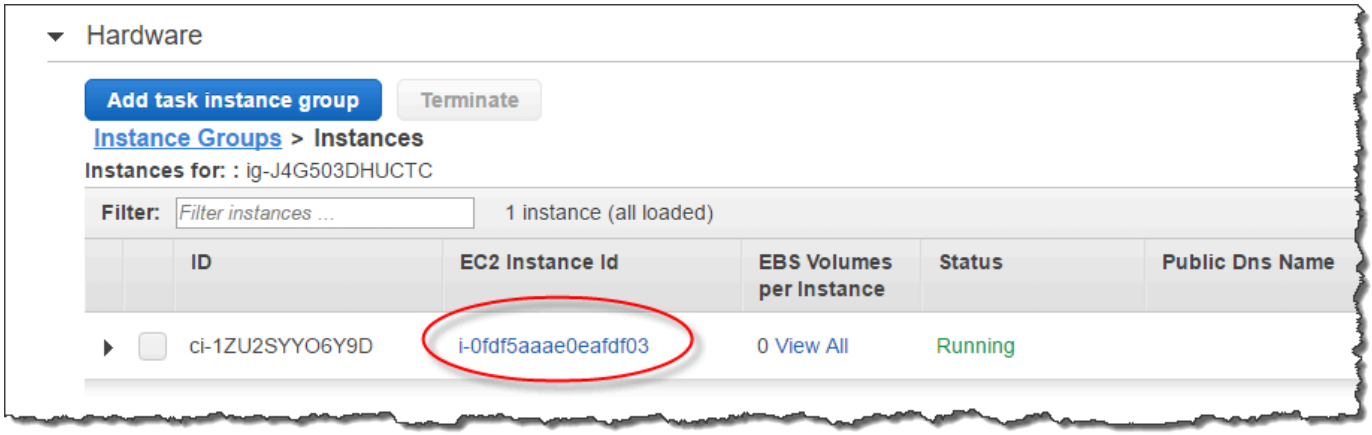
6. Insira o nome e a descrição do grupo de segurança.
7. Escolha Criar grupo de segurança.

Para adicionar o grupo de segurança do Amazon EC2 que você criou na etapa anterior ao cluster do Amazon Redshift e ao cluster do Amazon EMR

1. No Amazon Redshift, escolha seu cluster.
2. Escolha Properties.
3. Visualize a seção Network and security (Rede e segurança) e escolha Edit (Editar).
4. Em VPC security group (Grupo de segurança da VPC), escolha o novo nome do grupo de segurança.
5. Escolha Salvar alterações.

Para adicionar o grupo de segurança do Amazon EC2 ao cluster do Amazon EMR

1. No Amazon EMR, escolha seu cluster. Para obter mais informações, consulte Usar configurações de segurança para definir a segurança do cluster no Guia de gerenciamento do Amazon EMR.
2. Em Hardware, escolha o link para o nó principal.
3. Escolha o link na coluna EC2 Instance ID (ID da instância do EC2).



4. Escolha Actions (Ações), Security (Segurança) e Change security groups (Alterar grupos de segurança).
5. Em Associated sercurity groups (Grupos de segurança associados), escolha o novo grupo de segurança e escolha Add security group (Adicionar grupo de segurança).
6. Escolha Salvar.

Tabelas externas para o Redshift Spectrum

Este tópico descreve como criar e usar tabelas externas com o Redshift Spectrum. Tabelas externos são coleções de tabelas que você usa como referência para acessar dados fora do cluster do Amazon Redshift. Essas tabelas contêm metadados sobre os dados externos que o Redshift Spectrum lê.

Crie uma tabela externa em um esquema externo. Para criar tabelas externas, você deve ser proprietário do esquema externo ou um superusuário. Para transferir a propriedade de um esquema externo, use ALTER SCHEMA para alterar o proprietário. O exemplo a seguir altera o proprietário do esquema `spectrum_schema` para `newowner`.

```
alter schema spectrum_schema owner to newowner;
```

Para executar uma consulta do Redshift Spectrum, você precisa das seguintes permissões:

- Permissão de uso no esquema
- Permissão para criar tabelas temporárias no banco de dados atual

O exemplo a seguir concede permissão de uso no esquema `spectrum_schema` para o grupo de usuários `spectrumusers`.

```
grant usage on schema spectrum_schema to group spectrumusers;
```

O exemplo a seguir concede permissão temporária no banco de dados `spectrumdb` para o grupo de usuários `spectrumusers`.

```
grant temp on database spectrumdb to group spectrumusers;
```

Você pode criar uma tabela externa no Amazon Redshift, AWS Glue, Amazon Athena ou em uma metastore do Apache Hive. Para obter mais informações, consulte Conceitos básicos do uso do AWS Glue no Guia do desenvolvedor do AWS Glue, Conceitos básicos no Manual do usuário do Amazon Athena ou Apache Hive no Guia do desenvolvedor do Amazon EMR.

Se sua tabela externa estiver definida no AWS Glue, no Athena ou em uma metastore do Hive, crie primeiro um esquema externo que faça referência ao banco de dados externo. Em seguida, inclua uma referência à tabela externa na instrução SELECT prefixando o nome da tabela com o nome do esquema, sem precisar criar a tabela no Amazon Redshift. Para obter mais informações, consulte Esquemas externos no Amazon Redshift Spectrum.

Para permitir que o Amazon Redshift visualize tabelas no AWS Glue Data Catalog, adicione o `glue:GetTable` para a função do IAM do Amazon Redshift. Caso contrário, poderá ocorrer um erro semelhante ao seguinte.

```
RedshiftIamRoleSession is not authorized to perform: glue:GetTable on resource: *;
```

Por exemplo, suponha que você tem uma tabela externa denominada `lineitem_athena` definida em um catálogo externo do Athena. Nesse caso, você pode definir um esquema externo denominado `athena_schema` e, em seguida, consultar a tabela usando o comando SELECT a seguir.

```
select count(*) from athena_schema.lineitem_athena;
```

Para definir uma tabela externa no Amazon Redshift, use o comando CREATE EXTERNAL TABLE. A instrução da tabela externa define as colunas da tabela, o formato dos arquivos de dados e a localização dos dados no Amazon S3. O Redshift Spectrum faz uma varredura dos arquivos na pasta especificada e em todas as subpastas. O Redshift Spectrum ignora os arquivos ocultos e os

arquivos que começam com um ponto, um sublinhado ou um símbolo do jogo da velha (. , _, ou #) ou terminam com um til (~).

O exemplo a seguir cria uma tabela chamada SALES no esquema externo do Amazon Redshift denominado `spectrum`. Os dados estão em arquivos de texto delimitados por tabulação.

```
create external table spectrum.sales(
salesid integer,
listid integer,
sellerid integer,
buyerid integer,
eventid integer,
dateid smallint,
qtysold smallint,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp)
row format delimited
fields terminated by '\t'
stored as textfile
location 's3://redshift-downloads/tickit/spectrum/sales/'
table properties ('numRows'='172000');
```

Para visualizar as tabelas externas, consulte a exibição do sistema SVV_EXTERNAL_TABLES.

Pseudocolunas

Por padrão, o Amazon Redshift cria tabelas externas com as pseudocolunas `$path`, `$size` e `$spectrum_oid`. Selecione a coluna `$path` para exibir o caminho para os arquivos de dados no Amazon S3 e selecione a coluna `$size` para exibir o tamanho dos arquivos de dados em cada linha retornada por uma consulta. A coluna `$spectrum_oid` possibilita realizar consultas correlacionadas com o Redshift Spectrum. Para ver um exemplo, consulte Exemplo: Executar subconsultas correlacionadas no Redshift Spectrum. Você precisa delimitar os nomes de colunas `$path`, `$size` e `$spectrum_oid` com aspas duplas. A cláusula SELECT * não retornará as pseudocolunas. Você deve incluir explicitamente os nomes de coluna `$path`, `$size` e `$spectrum_oid` na consulta, como mostra o exemplo a seguir.

```
select "$path", "$size", "$spectrum_oid"
from spectrum.sales_part where saledate = '2008-12-01';
```

Você pode desabilitar a criação de pseudocolunas em uma sessão. Basta definir o parâmetro de configuração `spectrum_enable_pseudo_columns` como `false`. Para obter mais informações, consulte spectrum_enable_pseudo_columns. Você também pode desabilitar somente a pseudocoluna `$spectrum_oid` definindo o parâmetro `enable_spectrum_oid` como `false`. Para obter mais informações, consulte enable_spectrum_oid. No entanto, desabilitar a pseudocoluna `$spectrum_oid` também desabilita a compatibilidade de consultas correlacionadas com o Redshift Spectrum.

> **Important**
>
> A seleção de `$size`, `$path` ou `$spectrum_oid` gera cobranças porque o Redshift Spectrum verifica os arquivos de dados no Amazon S3 para determinar o tamanho do conjunto de resultados. Para obter mais informações, consulte Preço do Amazon Redshift.

Exemplo de pseudocolunas

O exemplo a seguir retorna o tamanho total de arquivos de dados relacionados de uma tabela externa.

```
select distinct "$path", "$size"
from spectrum.sales_part;

 $path                                                                      | $size
----------------------------------------------------------------------------+-------
s3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-01/ |  1616
s3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-02/ |  1444
s3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-03/ |  1644
```

Dividir as tabelas externas do Redshift Spectrum

Ao dividir seus dados em partições, você pode restringir a quantidade de varreduras de dados do Redshift Spectrum filtrando pela chave de partição. Você pode dividir seus dados em partições usando qualquer chave.

Uma prática comum é dividir os dados com base no tempo. Por exemplo, você pode escolher a partição por ano, mês, data e hora. Se você tiver dados vindos de várias origens, pode dividi-los em partições por um identificador de origem dos dados e por data.

O procedimento a seguir descreve como dividir os dados em partições.

Para dividir seus dados em partições

1. Armazene os dados em pastas no Amazon S3 de acordo com sua chave de partição.

   Crie uma pasta para cada valor de partição e nomeie a pasta com a chave e o valor da partição. Por exemplo, se você dividir por data, pode ter pastas denominadas `saledate=2017-04-01`, `saledate=2017-04-02`, e assim por diante. O Redshift Spectrum faz uma varredura dos arquivos na pasta da partição e em todas as subpastas. O Redshift Spectrum ignora os arquivos ocultos e os arquivos que começam com um ponto, um sublinhado ou um símbolo do jogo da velha (. , _, ou #) ou terminam com um til (~).

2. Crie uma tabela externa e especifique a chave de partição na cláusula PARTITIONED BY.

   A chave de partição não pode ser o nome de uma coluna da tabela. O tipo dos dados pode ser SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE PRECISION, BOOLEAN, CHAR, VARCHAR, DATE ou TIMESTAMP.

3. Adicione as partições.

   Usando o ALTER TABLE … ADD PARTITION, adicione cada partição especificando a coluna e a chave-valor de partição, além do local da pasta de partição no Amazon S3. Você pode adicionar várias partições em um único comando ALTER TABLE… ADD. O exemplo a seguir inclui partições para `'2008-01'` e `'2008-03'`.

```
alter table spectrum.sales_part add
partition(saledate='2008-01-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/
saledate=2008-01/'
partition(saledate='2008-03-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/
saledate=2008-03/';
```

> **Note**
>
> Se usar o catálogo do AWS Glue, você poderá adicionar até 100 partições usando um único comando ALTER TABLE.

Exemplos de particionamento de dados

Neste exemplo, você cria uma tabela externa que é particionada por uma única chave de partição e uma tabela externa que é particionada por duas chaves de partição.

Os dados deste exemplo estão localizados em um bucket do Amazon S3 que oferece acesso de leitura a todos os usuários autenticados da AWS. O cluster e os arquivos de dados externos devem estar localizados na mesma Região da AWS. O bucket de dados de exemplo está na região Leste dos EUA (Norte da Virgínia) (us-east-1). Para acessar os dados usando o Redshift Spectrum, seu cluster também deve estar em us-east-1. Para listar as pastas no Amazon S3, execute o comando a seguir.

```
aws s3 ls s3://redshift-downloads/tickit/spectrum/sales_partition/
```

```
PRE saledate=2008-01/
PRE saledate=2008-03/
PRE saledate=2008-04/
PRE saledate=2008-05/
PRE saledate=2008-06/
PRE saledate=2008-12/
```

Se você ainda não tiver um esquema externo, execute o seguinte comando. Substitua o nome de recurso da Amazon (ARN) para a sua função do AWS Identity and Access Management (IAM).

```
create external schema spectrum
from data catalog
database 'spectrumdb'
iam_role 'arn:aws:iam::123456789012:role/myspectrumrole'
create external database if not exists;
```

Exemplo 1: particionamento com uma única chave de partição

No exemplo a seguir, você cria uma tabela externa que é particionada pelo mês.

Para criar uma tabela externa particionada pelo mês, execute o seguinte comando.

```
create external table spectrum.sales_part(
salesid integer,
listid integer,
sellerid integer,
```

```
buyerid integer,
eventid integer,
dateid smallint,
qtysold smallint,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp)
partitioned by (saledate char(10))
row format delimited
fields terminated by '|'
stored as textfile
location 's3://redshift-downloads/tickit/spectrum/sales_partition/'
table properties ('numRows'='172000');
```

Para adicionar as partições, execute o seguinte comando ALTER TABLE.

```
alter table spectrum.sales_part add
partition(saledate='2008-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-01/'

partition(saledate='2008-03')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-03/'

partition(saledate='2008-04')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-04/';
```

Para selecionar dados na tabela particionada, execute a consulta a seguir.

```
select top 5 spectrum.sales_part.eventid, sum(spectrum.sales_part.pricepaid)
from spectrum.sales_part, event
where spectrum.sales_part.eventid = event.eventid
  and spectrum.sales_part.pricepaid > 30
  and saledate = '2008-01'
group by spectrum.sales_part.eventid
order by 2 desc;
```

```
eventid | sum
--------+---------
   4124 | 21179.00
   1924 | 20569.00
   2294 | 18830.00
   2260 | 17669.00
```

```
   6032 | 17265.00
```

Para visualizar as partições da tabela externa, consulte a exibição do sistema SVV_EXTERNAL_PARTITIONS.

```
select schemaname, tablename, values, location from svv_external_partitions
where tablename = 'sales_part';
```

```
schemaname | tablename  | values      | location

-----------+------------+-------------
+-------------------------------------------------------------------------
spectrum   | sales_part | ["2008-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-01
spectrum   | sales_part | ["2008-03"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-03
spectrum   | sales_part | ["2008-04"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-04
```

Exemplo 2: particionamento com várias chaves de partição

Para criar uma tabela externa particionada por `date` e por `eventid`, execute o seguinte comando.

```
create external table spectrum.sales_event(
salesid integer,
listid integer,
sellerid integer,
buyerid integer,
eventid integer,
dateid smallint,
qtysold smallint,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp)
partitioned by (salesmonth char(10), event integer)
row format delimited
fields terminated by '|'
stored as textfile
location 's3://redshift-downloads/tickit/spectrum/salesevent/'
table properties ('numRows'='172000');
```

Para adicionar as partições, execute o seguinte comando ALTER TABLE.

```
alter table spectrum.sales_event add
partition(salesmonth='2008-01', event='101')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-01/
event=101/'

partition(salesmonth='2008-01', event='102')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-01/
event=102/'

partition(salesmonth='2008-01', event='103')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-01/
event=103/'

partition(salesmonth='2008-02', event='101')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-02/
event=101/'

partition(salesmonth='2008-02', event='102')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-02/
event=102/'

partition(salesmonth='2008-02', event='103')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-02/
event=103/'

partition(salesmonth='2008-03', event='101')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-03/
event=101/'

partition(salesmonth='2008-03', event='102')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-03/
event=102/'

partition(salesmonth='2008-03', event='103')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-03/
event=103/';
```

Execute a consulta a seguir para selecionar os dados da tabela particionada.

```
select spectrum.sales_event.salesmonth, event.eventname,
 sum(spectrum.sales_event.pricepaid)
from spectrum.sales_event, event
where spectrum.sales_event.eventid = event.eventid
```

```
  and salesmonth = '2008-02'
 and (event = '101'
 or event = '102'
 or event = '103')
group by event.eventname, spectrum.sales_event.salesmonth
order by 3 desc;
```

```
salesmonth | eventname       | sum
-----------+-----------------+--------
2008-02    | The Magic Flute | 5062.00
2008-02    | La Sonnambula   | 3498.00
2008-02    | Die Walkure     |  534.00
```

Mapeamento de colunas de tabela externa para colunas do ORC

Você usa tabelas externas do Amazon Redshift Spectrum para consultar dados de arquivos no formato ORC. O formato de coluna de linha otimizada (ORC) é um formato de arquivo de armazenamento colunar que oferece suporte para estruturas de dados aninhadas. Para obter mais informações sobre como consultar dados aninhados, consulte Consultar dados aninhados com o Amazon Redshift Spectrum.

Ao criar uma tabela externa que faz referência a dados em um arquivo ORC, você mapeia cada coluna na tabela externa para uma coluna nos dados ORC. Para fazer isso, você usa um dos seguintes métodos:

- Mapeamento por posição
- Mapeamento por nome de coluna

O mapeamento por nome de coluna é o padrão.

Mapeamento por posição

Com o mapeamento por posição, a primeira coluna definida na tabela externa é mapeada para a primeira coluna no arquivo de dados ORC, a segundo para a segundo e assim por diante. O mapeamento por posição requer que a ordem das colunas na tabela externa e no arquivo ORC corresponda. Se a ordem das colunas não corresponder, você poderá mapear as colunas por nome.

> **Important**
>
> Em releases anteriores, o Redshift Spectrum usava o mapeamento de posição por padrão. Se você precisar continuar usando o mapeamento por posição para tabelas existentes, defina a propriedade de tabela `orc.schema.resolution` como `position`, como mostra o exemplo a seguir.
>
> ```
> alter table spectrum.orc_example
> set table properties('orc.schema.resolution'='position');
> ```

Por exemplo, a tabela `SPECTRUM.ORC_EXAMPLE` é definida da seguinte maneira.

```
create external table spectrum.orc_example(
int_col int,
float_col float,
nested_col struct<
  "int_col" : int,
  "map_col" : map<int, array<float >>
   >
) stored as orc
location 's3://example/orc/files/';
```

A estrutura da tabela pode ser abstraída como segue.

- 'int_col' : int
- 'float_col' : float
- 'nested_col' : struct
  - o 'int_col' : int
  - o 'map_col' : map
    - key : int
    - value : array
      - value : float

O arquivo ORC subjacente tem a seguinte estrutura de arquivos.

- ORC file root(id = 0)
  - o 'int_col' : int (id = 1)
  - o 'float_col' : float (id = 2)
  - o 'nested_col' : struct (id = 3)

```
      - 'int_col' : int (id = 4)
      - 'map_col' : map (id = 5)
         - key : int (id = 6)
         - value : array (id = 7)
            - value : float (id = 8)
```

Neste exemplo, você pode mapear cada coluna na tabela externa para uma coluna no arquivo ORC estritamente por posição. Veja o mapeamento a seguir.

Nome da coluna de tabela externa	ID de coluna ORC	Nome da coluna ORC
int_col	1	int_col
float_col	2	float_col
nested_col	3	nested_col
nested_col.int_col	4	int_col
nested_col.map_col	5	map_col
nested_col.map_col.key	6	NA
nested_col.map_col.value	7	NA
nested_col.map_col.value.item	8	NA

Mapeamento por nome de coluna

Usando o mapeamento de nomes, você mapeia colunas em uma tabela externa para colunas nomeadas em arquivos ORC no mesmo nível, com o mesmo nome.

Por exemplo, suponha que você queira mapear a tabela do exemplo anterior, `SPECTRUM.ORC_EXAMPLE`, com um arquivo ORC que usa a seguinte estrutura de arquivo.

```
• ORC file root(id = 0)
   o 'nested_col' : struct (id = 1)
      - 'map_col' : map (id = 2)
         - key : int (id = 3)
```

```
        - value : array (id = 4)
           - value : float (id = 5)
     - 'int_col' : int (id = 6)
  o 'int_col' : int (id = 7)
  o 'float_col' : float (id = 8)
```

Usando mapeamento por posição, o Redshift Spectrum tenta o seguinte mapeamento.

Nome da coluna de tabela externa	ID de coluna ORC	Nome da coluna ORC
int_col	1	struct
float_col	7	int_col
nested_col	8	float_col

Quando você consulta uma tabela com o mapeamento de posição anterior, o comando SELECT falha na validação de tipo, pois as estruturas são diferentes.

Você pode mapear a mesma tabela externa para ambas as estruturas de arquivo mostradas nos exemplos anteriores usando o mapeamento por nome de coluna. As colunas da tabela `int_col`, `float_col` e `nested_col` são mapeadas por nome de coluna para colunas com os mesmos nomes no arquivo ORC. A coluna denominada `nested_col` na tabela externa é uma coluna `struct` com subcolunas denominadas `map_col` e `int_col`. As subcolunas também são mapeadas corretamente para as colunas correspondentes no arquivo ORC por nome da coluna.

Criar tabelas externas para dados gerenciados no Apache Hudi

Para consultar dados no formato Apache Hudi Copy On Write (CoW), use tabelas externas do Amazon Redshift Spectrum. Uma tabela do Delta Lake é uma coleção de arquivos do Apache Parquet armazenados no Amazon S3. É possível ler tabelas de cópia em gravação (CoW) nas versões 0.5.2, 0.6.0, 0.7.0, 0.8.0, 0.9.0, 0.10.0, 0.10.1, 0.11.0 e 0.11.1 do Apache Hudi que são criadas e modificadas com operações de gravação insert, delete e upsert. Por exemplo, tabelas de bootstrap não são compatíveis. Para obter mais informações, consulte Tabela do Copy On Write na documentação de código aberto do Apache Hudi.

Ao criar uma tabela externa que faz referência a dados em um formato Hudi CoW, você mapeia cada coluna na tabela externa para uma coluna nos dados do Hudi. O mapeamento é feito por coluna.

As instruções de linguagem de definição de dados (DDL) para tabelas particionadas e não particionadas do Hudi são semelhantes às de outros formatos de arquivo do Apache Parquet. Para tabelas do Hudi, defina `INPUTFORMAT` como `org.apache.hudi.hadoop.HoodieParquetInputFormat`. O parâmetro `LOCATION` deve apontar para a pasta base da tabela do Hudi que contém a pasta `.hoodie`, necessária para estabelecer a linha do tempo de confirmação do Hudi. Em alguns casos, uma operação SELECT em uma tabela do Hudi pode falhar, exibindo a mensagem Nenhuma linha de tempo de confirmação válida do Hudi foi encontrada. Se for o caso, verifique se a pasta `.hoodie` está no local correto e contém uma linha de tempo de confirmação do Hudi válida.

> **Note**
>
> O formato Apache Hudi só é compatível quando você usa um AWS Glue Data Catalog. Ele não é compatível quando você usa uma metastore do Apache Hive como o catálogo externo.

O DDL para definir uma tabela não particionada tem o formato a seguir.

```
CREATE EXTERNAL TABLE tbl_name (columns)
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS
INPUTFORMAT 'org.apache.hudi.hadoop.HoodieParquetInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat'
LOCATION 's3://s3-bucket/prefix'
```

O DDL para definir uma tabela particionada tem o formato a seguir.

```
CREATE EXTERNAL TABLE tbl_name (columns)
PARTITIONED BY(pcolumn1 pcolumn1-type[,...])
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS
INPUTFORMAT 'org.apache.hudi.hadoop.HoodieParquetInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat'
LOCATION 's3://s3-bucket/prefix'
```

Para adicionar partições a uma tabela do Hudi particionada, execute um comando ALTER TABLE ADD PARTITION, no qual o parâmetro `LOCATION` aponta para a subpasta do Amazon S3 com os arquivos que pertencem à partição.

O DDL para adicionar partições tem o formato a seguir.

```
ALTER TABLE tbl_name
ADD IF NOT EXISTS PARTITION(pcolumn1=pvalue1[,...])
LOCATION 's3://s3-bucket/prefix/partition-path'
```

Criar tabelas externas para dados gerenciados no Delta Lake

Para consultar dados em tabelas do Delta Lake, você pode usar tabelas externas do Amazon Redshift Spectrum.

Para acessar uma tabela do Delta Lake a partir do Redshift Spectrum, gere um manifesto antes da consulta. Um manifesto do Delta Lake contém uma lista de arquivos que compõem um snapshot consistente da tabela do Delta Lake. Em uma tabela particionada, há um manifesto por partição. Uma tabela do Delta Lake é uma coleção de arquivos do Apache Parquet armazenados no Amazon S3. Para obter mais informações, consulte Delta Lake na documentação de código aberto do Delta Lake.

Ao criar uma tabela externa que faz referência a dados em tabelas do Delta Lake, você mapeia cada coluna na tabela externa para uma coluna na tabela do Delta Lake. O mapeamento é feito pelo nome da coluna.

O DDL para tabelas particionadas e não particionadas do Delta Lake é semelhante ao dos outros formatos de arquivo do Apache Parquet. Para tabelas do Delta Lake, defina `INPUTFORMAT` como `org.apache.hadoop.hive.ql.io.SymlinkTextInputFormat` e `OUTPUTFORMAT` como `org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat`. O `LOCATION` deve apontar para a pasta de manifesto na pasta base da tabela. Se uma operação SELECT em uma tabela do Delta Lake falhar, consulte para obter os possíveis motivos Limitações e solução de problemas para tabelas do Delta Lake.

O DDL para definir uma tabela não particionada tem o formato a seguir.

```
CREATE EXTERNAL TABLE tbl_name (columns)
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS
INPUTFORMAT 'org.apache.hadoop.hive.ql.io.SymlinkTextInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION 's3://s3-bucket/prefix/_symlink_format_manifest'
```

O DDL para definir uma tabela particionada tem o formato a seguir.

```
CREATE EXTERNAL TABLE tbl_name (columns)
```

```
PARTITIONED BY(pcolumn1 pcolumn1-type[,...])
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS
INPUTFORMAT 'org.apache.hadoop.hive.ql.io.SymlinkTextInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION 's3://s3-bucket>/prefix/_symlink_format_manifest'
```

Para adicionar partições a uma tabela particionada do Delta Lake, execute um comando ALTER TABLE ADD PARTITION no qual o parâmetro LOCATION aponte para a subpasta do Amazon S3 que contém o manifesto para a partição.

O DDL para adicionar partições tem o formato a seguir.

```
ALTER TABLE tbl_name
ADD IF NOT EXISTS PARTITION(pcolumn1=pvalue1[,...])
LOCATION
's3://s3-bucket/prefix/_symlink_format_manifest/partition-path'
```

Ou execute o DDL que aponte diretamente para o arquivo manifesto do Delta Lake.

```
ALTER TABLE tbl_name
ADD IF NOT EXISTS PARTITION(pcolumn1=pvalue1[,...])
LOCATION
's3://s3-bucket/prefix/_symlink_format_manifest/partition-path/manifest'
```

Limitações e solução de problemas para tabelas do Delta Lake

Considere o seguinte ao consultar tabelas do Delta Lake do Redshift Spectrum:

- Se um manifesto apontar para um snapshot ou partição que não existe mais, as consultas falharão até que um novo manifesto válido seja gerado. Por exemplo, isso pode resultar de uma operação VACUUM na tabela subjacente,
- Os manifestos do Delta Lake fornecem apenas consistência em nível de partição.

A tabela a seguir explica alguns potenciais motivos para determinados erros quando você consulta uma tabela do Delta Lake.

A mensagem de erro	Possível motivo
O manifesto do Delta Lake no bucket s3-bucket -1 não pode conter entradas no bucket s3-bucket-2.	As entradas de manifesto apontam para arquivos em um bucket do Amazon S3 diferente do especificado.
Os arquivos do Delta Lake devem estar na mesma pasta.	As entradas de manifesto apontam para arquivos que têm um prefixo do Amazon S3 diferente do especificado.
Não foi possível encontrar o arquivo filename listado no manifesto do Delta Lake manifest-path.	Um arquivo listado no manifesto não foi encontrado no Amazon S3.
Erro ao buscar o manifesto do Delta Lake.	O manifesto não foi encontrado no Amazon S3.
Caminho do S3 inválido.	Uma entrada no arquivo manifesto não é um caminho válido do Amazon S3 ou o arquivo manifesto foi corrompido.

Uso de tabelas do Apache Iceberg com o Amazon Redshift

> **Note**
>
> Para obter o melhor desempenho ao usar tabelas do Apache Iceberg com o Amazon Redshift, você deve gerar estatísticas de colunas para as tabelas que estão usando o AWS Glue. Para ter mais informações, consulte Gerar estatísticas de colunas para tabelas Iceberg no Guia do desenvolvedor do AWS Glue.

Este tópico descreve como utilizar tabelas no formato Apache Iceberg com o Amazon Redshift. Apache Iceberg é um formato de tabela de código aberto de alto desempenho para data lakes. Para obter mais informações, consulte Apache Iceberg na documentação do Apache Iceberg.

É possível consultar tabelas do Apache Iceberg catalogadas no AWS Glue Data Catalog com o Amazon Redshift. Os tipos de instância RG e o Redshift sem servidor usam a própria computação

para processar consultas de data lake, enquanto os tipos de instância RA3 usam o Redshift Spectrum. Para obter mais informações, consulte Consulta do data lake.

O Amazon Redshift fornece consistência transacional para consultar tabelas do Apache Iceberg. É possível manipular os dados nas tabelas usando serviços compatíveis com ACID (atomicidade, consistência, isolamento, durabilidade), como Amazon Athena e Amazon EMR, enquanto executa consultas usando o Amazon Redshift. O Amazon Redshift pode usar as estatísticas da tabela armazenadas nos metadados do Apache Iceberg para otimizar os planos de consulta e reduzir a verificação de arquivos durante o processamento da consulta. Com o Amazon Redshift SQL, é possível unir tabelas do Redshift com tabelas de data lake.

Como começar a usar tabelas do Iceberg com o Amazon Redshift:

1. Crie uma tabela do Apache Iceberg em um banco de dados do AWS Glue Data Catalog usando um serviço compatível, como Amazon Athena ou Amazon EMR. Para criar uma tabela do Iceberg usando o Athena, consulte Usar tabelas do Apache Iceberg no Guia do usuário do Amazon Athena.
2. Crie um cluster do Amazon Redshift ou um grupo de trabalho do Redshift sem servidor com um perfil do IAM associado que permita o acesso ao data lake. Consulte informações sobre como criar clusters ou grupos de trabalho em Clusters provisionados do Amazon Redshift e em Criar um data warehouse com o Amazon Redshift sem servidor no Guia de conceitos básicos do Amazon Redshift.
3. Conecte-se ao cluster ou grupo de trabalho usando o editor de consultas v2 ou um cliente SQL de terceiros. Para ter informações sobre como se conectar usando o Editor de Consultas v2, consulte Conectar-se a um data warehouse do Amazon Redshift usando ferramentas de cliente SQL no Guia de gerenciamento do Amazon Redshift.
4. Crie um esquema externo no banco de dados do Amazon Redshift para um banco de dados específico do Catálogo de Dados que inclui as tabelas do Iceberg. Para obter mais informações sobre como criar um esquema externo, consulte Esquemas externos no Amazon Redshift Spectrum.
5. Execute consultas SQL para acessar as tabelas do Iceberg no esquema externo que você criou.

Considerações ao usar tabelas do Apache Iceberg com o Amazon Redshift

Considere o seguinte ao usar o Amazon Redshift com tabelas do Iceberg:

- Suporte à versão do Iceberg: o Amazon Redshift é compatível com a execução de consultas nas seguintes versões das tabelas do Iceberg:
  - A versão 1, que define como tabelas analíticas grandes são gerenciadas usando arquivos de dados imutáveis.
  - A versão 2 adiciona a capacidade de dar suporte a atualizações e exclusões no nível da linha, ao mesmo tempo em que mantém os arquivos de dados existentes inalterados e processa alterações de dados da tabela usando arquivos de exclusão.
  - A versão 3 introduz valores padrão de colunas, rastreamento da linhagem de linhas e vetores de exclusão. Os valores padrão permitem definir valores iniciais para as colunas, que são aplicados quando os valores não são fornecidos explicitamente. A linhagem de linhas fornece pseudocolunas que rastreiam a identidade da linha e o histórico de modificações. Os vetores de exclusão oferecem uma representação mais compacta das exclusões em nível de linha, permitindo leituras e gravações mais rápidas em comparação com os arquivos de exclusão do Iceberg v2. Para obter mais informações, consulte the section called “Recursos do Iceberg v3”.

  Para obter informações sobre as diferenças entre as tabelas das versões 1, 2 e 3, consulte Format version changes na documentação do Apache Iceberg.

  Para obter mais informações sobre os recursos do Iceberg v3 compatíveis com o Amazon Redshift e as limitações atuais, consulte the section called “Recursos do Iceberg v3”.
- Adicionar partições: você não precisa adicionar partições manualmente a tabelas do Apache Iceberg. Novas partições nas tabelas do Apache Iceberg são detectadas automaticamente pelo Amazon Redshift e nenhuma operação manual é necessária para atualizar as partições na definição da tabela. As alterações na especificação da partição também são aplicadas automaticamente às consultas sem nenhuma intervenção do usuário.
- Ingestão de dados do Iceberg no Amazon Redshift: é possível usar os comandos INSERT INTO ou CREATE TABLE AS para importar dados da tabela do Iceberg para uma tabela local do Amazon Redshift. No momento, é possível usar o comando COPY para ingerir o conteúdo de uma tabela do Apache Iceberg em uma tabela local do Amazon Redshift.
- Visões materializadas: você pode criar visões materializadas em tabelas do Apache Iceberg como qualquer outra tabela externa no Amazon Redshift. As mesmas considerações para outros formatos de tabela de data lake se aplicam às tabelas do Apache Iceberg. No momento, não é possível usar regravação automática de consultas e visões materializadas automáticas em tabelas de data lake.
- Controle de acesso detalhado do AWS Lake Formation: o Amazon Redshift permite um controle de acesso detalhado do AWS Lake Formation em tabelas do Apache Iceberg.

- Parâmetros de tratamento de dados definidos pelo usuário: o Amazon Redshift permite parâmetros de tratamento de dados definidos pelo usuário nas tabelas do Apache Iceberg. Você usa parâmetros de tratamento de dados definidos pelo usuário em arquivos existentes para personalizar os dados que estão sendo consultados em tabelas externas e evitar erros de verificação. Esses parâmetros fornecem recursos para lidar com incompatibilidades entre o esquema da tabela e os dados reais nos arquivos. Você também pode usar parâmetros de tratamento de dados definidos pelo usuário nas tabelas do Apache Iceberg.
- Perguntas sobre viagens no tempo: no momento, as consultas de viagem no tempo não são compatível com tabelas do Apache Iceberg.
- Preços: quando você acessa as tabelas do Iceberg de um cluster RG ou de um grupo de trabalho do Redshift sem servidor, as consultas do data lake são executadas nos recursos computacionais do próprio cluster ou grupo de trabalho, portanto, não há cobrança separada pelas consultas ao data lake. Ao acessar tabelas do Iceberg por meio de um cluster DC2 ou RA3, você recebe a cobrança segundo os preços do Redshift Spectrum. Para obter informações sobre preços, consulte Preços do Amazon Redshift.
- Armazenamento em cache de metadados: esse armazenamento pressupõe que os arquivos de metadados são imutáveis com base na especificação do Iceberg. A imutabilidade do arquivo de metadados é um requisito para a integridade dos dados no Amazon Redshift.
- Identidade federada: a identidade federada não é aceita ao gravar em tabelas do Apache Iceberg. Isso inclui o uso da palavra-chave SESSION para o parâmetro IAM_ROLE ao criar esquemas externos. Para acessar mais informações sobre os parâmetros IAM_ROLE, consulte CREATE EXTERNAL SCHEMA.

Tipos de dados compatíveis com tabelas do Apache Iceberg

Este tópico descreve os tipos de dados compatíveis que o Redshift Spectrum pode ler em tabelas no formato Apache Iceberg.

O Amazon Redshift pode consultar tabelas do Iceberg que contêm os seguintes tipos de dados:

```
binary
boolean
date
decimal
double
float
int
```

```
list
long
map
string
struct
timestamp without time zone
```

Ao criar e definir uma tabela do Iceberg, use os nomes dos tipos de dados do Amazon Redshift na instrução SQL. O Redshift os associa automaticamente aos tipos do iceberg correspondentes. Para obter mais informações sobre tipos de dados do Iceberg, consulte Esquemas para o Iceberg na documentação do Apache Iceberg.

Ao ler as tabelas do Iceberg, os tipos de dados do Iceberg são associados nos tipos de dados do Redshift, conforme mostrado na tabela abaixo:

Tipo do Iceberg	Tipo do Amazon Redshift	Observações
`boolean`	`boolean`	-
-	`tinyint`	Não é aceito em tabelas do Iceberg.
-	`smallint`	Não é aceito em tabelas do Iceberg.
`int`	`int`	-
`long`	`bigint`	-
`double`	`double precision`	-
`float`	`real`	-
`decimal(P, S)`	`decimal(P, S)`	P é precisão, S é escala.
-	`char`	Não é aceito em tabelas do Iceberg.
`string`	`varchar(16384)`	Strings maiores do que 16384 estão truncadas para 16384.

Tipo do Iceberg	Tipo do Amazon Redshift	Observações
`binary`	`varbyte(64000)`	-
`date`	`date`	-
`time`	-	-
`timestamp`	`timestamp`	-
`timestamptz`	`timestampz`	-
`list<E>`	`SUPER`	-
`map<K,V>`	`SUPER`	-
`struct<...>`	`SUPER`	-
`fixed(L)`	-	No momento, o tipo `fixed(L)` não é aceito no Redshift Spectrum.
`uuid`	-	No momento, o tipo `uuid` não é aceito no Redshift Spectrum.
`variant`	-	O Amazon Redshift não comporta o Iceberg V3.
`geometry`	-	O Amazon Redshift não comporta o Iceberg V3.
`geography`	-	O Amazon Redshift não comporta o Iceberg V3.
`timestamp_ns`	-	O Amazon Redshift não comporta o Iceberg V3.
`timestamptz_ns`	-	O Amazon Redshift não comporta o Iceberg V3.

Tipo do Iceberg	Tipo do Amazon Redshift	Observações
Unknown	-	O Amazon Redshift não comporta o Iceberg V3.

Os tipos de dados a seguir são aceitos ao criar tabelas do Iceberg a partir do Redshift. Os tipos de dados do Redshift são associados em tipos de dados do Iceberg, conforme mostrado na tabela a seguir.

Tipo do Amazon Redshift	Alias do Amazon Redshift	Tipo do Iceberg	Observações
integer	int, int4	int	-
bigint	int8	long	-
decimal	numeric	decimal(p,S)	-
real	float4	float	-
double precision	float8, float	double	-
varchar	charactte r varying,n varchar, text	string	O tipo de dados varchar(n) não é aceito ao criar uma tabela do Iceberg.
date	-	date	-
timestamp	-	timestamp	-
timestamptz	-	timestamptz	-
boolean	-	boolean	-

Ao gravar em tabelas do Iceberg, além dos tipos de dados mencionados na tabela anterior, alguns tipos de dados de origem são promovidos para seus tipos do Iceberg aceitos, conforme mostrado na tabela a seguir.

Tipo do Amazon Redshift	Tipo do Iceberg
`tinyint`	`int`
`smallint`	`int`
`varchar(n)`	`string`

A tentativa de usar tipos de dados não aceitos vai gerar erros de sintaxe. Ao criar uma tabela do Iceberg com uma cláusula `CREATE TABLE AS SELECT`, você pode adicionar conversão explícita para contornar a diferença de tipo.

Por exemplo, digamos que você tenha uma tabela do Redshift RMS com o seguinte esquema:

```
CREATE TABLE rms_t (c1 int, c2 char(20));
```

Se você quiser criar uma tabela do Iceberg usando `rms_t` como fonte, precisará de uma conversão explícita para a coluna `c2`, porque o tipo `varchar(n)` não é aceito:

```
CREATE TABLE ext_schema.iceberg_t AS SELECT c1, c2::varchar FROM rms_t;
```

Para obter mais informações sobre tipos de dados no Amazon Redshift, consulte Tipos de dados.

Fazer referência a tabelas do Iceberg no Amazon Redshift

O Amazon Redshift fornece várias maneiras de fazer referência a tabelas do Apache Iceberg armazenadas em seu data lake. Você pode usar esquemas externos para criar referências aos bancos de dados do Catálogo de Dados contendo tabelas do Iceberg ou usar a notação em três partes para acesso direto aos catálogos montados automaticamente.

Usar esquemas externos para fazer referência a tabelas do Iceberg

Os esquemas externos oferecem uma forma de fazer referência a tabelas em seu Catálogo de Dados a partir do Amazon Redshift. Ao criar um esquema externo, você deve estabelecer conexão

entre o banco de dados do Amazon Redshift e um banco de dados específico do Catálogo de Dados que inclua as tabelas do Iceberg.

Para criar um esquema externo para tabelas do Iceberg:

```
CREATE EXTERNAL SCHEMA schema_name
FROM DATA CATALOG
DATABASE 'glue_database_name'
IAM_ROLE 'arn:aws:iam::account-id:role/role-name';
```

Depois de criar o esquema externo, você pode consultar as tabelas do Iceberg usando a notação em duas partes:

```
SELECT * FROM schema_name.iceberg_table_name;
```

Você também pode unir tabelas do Iceberg com tabelas locais do Amazon Redshift:

```
SELECT r.customer_id, i.order_date, r.customer_name
FROM local_customers r
JOIN schema_name.iceberg_orders i
ON r.customer_id = i.customer_id;
```

Usando notação em três partes com catálogos montados automaticamente.

Com a notação de três partes, é possível fazer referência direta a tabelas em catálogos montados automaticamente, sem criar esquemas externos. Esse método é particularmente útil quando você trabalha com buckets de tabelas do Amazon S3 federados com o AWS Lake Formation. Para acessar informações sobre como configurar a montagem automática do Catálogo de Dados, consulte Simplificar o acesso a objetos externos no Amazon Redshift usando a montagem automática do AWS Glue Data Catalog.

A sintaxe da notação em três partes é:

```
"catalog_name".database_name.table_name
```

Por exemplo, para consultar uma tabela do Iceberg em um catálogo de tabelas do Amazon S3 montado automaticamente:

```
SELECT * FROM "my_table_bucket@s3tablescatalog".my_database.my_iceberg_table;
```

Para acessar mais informações sobre a integração de buckets de tabelas do Amazon S3 com o Amazon Redshift, consulte Integrar tabelas do S3 com o Amazon Redshift no Guia do usuário do Amazon S3.

Você também pode referenciar tabelas no catálogo raiz montado automaticamente `awsdatacatalog`, que fornece acesso direto aos bancos de dados e tabelas registrados no AWS Glue Data Catalog:

```
SELECT * FROM awsdatacatalog.my_database.my_iceberg_table;
```

Para obter mais informações sobre o uso do catálogo `awsdatacatalog` raiz, consulte Consulta do AWS Glue Data Catalog no Guia de gerenciamento do Amazon Redshift e Gerenciamento de namespaces do catálogo de dados no Guia do desenvolvedor do AWS Lake Formation.

Você também pode usar a instrução USE para definir um catálogo e um banco de dados padrão para buckets de tabelas do Amazon S3:

```
USE "my_table_bucket@s3tablescatalog".my_database;
SELECT * FROM my_iceberg_table;
```

Para definir um caminho de pesquisa para resolução de esquema com buckets de tabelas do Amazon S3:

```
USE "my_table_bucket@s3tablescatalog";
SET search_path TO my_database;
SELECT * FROM my_iceberg_table;
```

> **Note**
>
> As instruções USE e `search_path` não são compatíveis com `s3tablescatalog`. Não podem ser usadas com `awsdatacatalog`. Para referenciar tabelas em `awsdatacatalog`, use a notação completa de três partes.

Práticas recomendadas para fazer referência a tabelas do Iceberg

Uma tabela Apache Iceberg é uma única entidade lógica composta por vários arquivos: um arquivo de metadados raiz (metadata.json), listas de manifestos, arquivos de manifesto e arquivos de dados (normalmente .parquet). O arquivo de metadados raiz serve como ponto de entrada e contém

referências a todos os outros arquivos que compõem a tabela. Quando você concede ao Amazon Redshift acesso a uma tabela do Iceberg, o Amazon Redshift usa o arquivo de metadados raiz para descobrir e ler todos os arquivos de dados referenciados. Se o Amazon Redshift tiver acesso ao arquivo de metadados raiz, ele também vai pressupor e exigir acesso a todos os arquivos de dados subjacentes. Isso condiz com o design do Iceberg, em que o acesso no nível da tabela é a unidade de autorização pretendida.

Para melhorar o desempenho da consulta, o Amazon Redshift armazena em cache os arquivos de metadados do Iceberg (incluindo o arquivo de metadados raiz, as listas de manifestos e os arquivos de manifesto) na memória. O arquivo de metadados raiz (metadata.json) é revalidado em relação ao Amazon S3 em um intervalo configurável (TTL). Depois que o TTL expira, o Amazon Redshift executa uma solicitação HEAD do Amazon S3 no arquivo de metadados raiz para verificar se o perfil do IAM ainda tem acesso e se o arquivo não foi modificado. Se a verificação de permissão falhar ou o arquivo tiver sido alterado, a entrada em cache será removida e os metadados serão recuperados do Amazon S3. Como o arquivo de metadados raiz é o ponto de entrada para todo o acesso à tabela, essa revalidação serve como porta de permissão para toda a tabela. As listas de manifestos e os arquivos de manifesto são armazenados em cache sem revalidação independente de TTL; a validade de acesso é derivada da verificação de permissão de metadados raiz. Isso significa que, se você revogar as permissões do Amazon S3 em uma tabela do Iceberg, as consultas poderão continuar sendo bem-sucedidas por no máximo 2 minutos usando metadados em cache.

> **Important**
>
> O Amazon S3 permite que você defina permissões no nível do objeto individual, o que significa que é tecnicamente possível conceder acesso aos metadados de uma tabela do Iceberg e, ao mesmo tempo, restringir o acesso a alguns arquivos de dados subjacentes. Isso cria uma inconsistência de permissão que pode levar a falhas de consulta ou erros de acesso inesperados no Amazon Redshift.

O Amazon Redshift valida periodicamente o acesso ao arquivo de metadados raiz em cache, mas não valida nem impõe a consistência entre as permissões no nível dos metadados e no nível do arquivo de dados em seu bucket do Amazon S3. É responsabilidade do cliente verificar se as permissões são aplicadas de forma consistente a todos os arquivos que constituem uma tabela do Iceberg.

Para evitar isso, pense nas seguintes práticas recomendadas ao fazer referência a tabelas do Iceberg no Amazon Redshift:

- Use nomes de esquemas descritivos: ao criar esquemas externos, use nomes que indiquem claramente a origem e a finalidade dos dados, como `sales_data_lake` ou `customer_analytics`.
- Utilize as estatísticas da tabela: verifique se as estatísticas de colunas são geradas para suas tabelas do Iceberg usando o AWS Glue para otimizar o desempenho das consultas. O Amazon Redshift usa essas estatísticas para planejamento e otimização de consultas.
- Pense na atualização dos dados: as tabelas do Iceberg podem ser atualizadas por outros serviços enquanto você as consulta. O Amazon Redshift oferece consistência transacional, garantindo que você veja um snapshot consistente dos dados durante a execução da consulta.
- Use as permissões apropriadas do IAM: verifique se o cluster ou grupo de trabalho do Amazon Redshift tem as permissões necessárias do IAM para acessar os locais do Amazon S3 onde as tabelas do Iceberg estão armazenadas, bem como os metadados do Catálogo de Dados.
- Permissões no nível da tabela: conceda permissões no nível da tabela, não no nível do arquivo individual.
- Permissões uniformes: mantenha acesso uniforme em todo o caminho do Amazon S3 da tabela do Iceberg, incluindo todos os arquivos de metadados, manifesto e dados.
- Evite políticas restritivas no nível do objeto: não defina políticas restritivas no nível do objeto em arquivos Parquet individuais dentro do prefixo de uma tabela do Iceberg.
- Entenda o armazenamento em cache do TTL para alterações de permissão: quando você revoga as permissões do Amazon S3 em uma tabela do Iceberg, as consultas podem continuar sendo bem-sucedidas usando metadados raiz em cache por até a duração do TTL configurada (padrão: 2 minutos).
- Monitore a performance das consultas: use os recursos de monitoramento de consultas do Amazon Redshift para acompanhar a performance das consultas em relação às tabelas do Iceberg e otimizá-las conforme necessário.

Padrões de referência comuns

Os exemplos a seguir demonstram os padrões comuns para fazer referência a tabelas do Iceberg:

Agregar dados em várias tabelas do Iceberg:

```
SELECT
    region,
    SUM(sales_amount) as total_sales,
```

```
    COUNT(*) as transaction_count
FROM data_lake.sales_transactions
WHERE transaction_date >= '2024-01-01'
GROUP BY region
ORDER BY total_sales DESC;
```

Unir tabelas do Iceberg com tabelas locais do Amazon Redshift:

```
SELECT
    c.customer_name,
    c.customer_tier,
    SUM(o.order_amount) as total_orders
FROM customers c
JOIN data_lake.order_history o ON c.customer_id = o.customer_id
WHERE o.order_date >= CURRENT_DATE - INTERVAL '30 days'
GROUP BY c.customer_name, c.customer_tier;
```

Usar a notação em três partes com consultas complexas:

```
WITH recent_orders AS (
    SELECT customer_id, order_date, order_amount
    FROM "analytics_bucket@s3tablescatalog".ecommerce.orders
    WHERE order_date >= CURRENT_DATE - INTERVAL '7 days'
)
SELECT
    customer_id,
    COUNT(*) as order_count,
    AVG(order_amount) as avg_order_value
FROM recent_orders
GROUP BY customer_id
HAVING COUNT(*) > 1;
```

Gravar em tabelas do Apache Iceberg

Com o Amazon Redshift, é possível criar e gravar em tabelas do Apache Iceberg armazenadas no Amazon S3 e nos buckets de tabelas do Amazon S3. Gravar dados do Iceberg diretamente do Amazon Redshift simplifica seu gerenciamento de dados ao eliminar ferramentas extras. As tabelas do Iceberg devem ser registradas no AWS Glue Data Catalog.

É possível usar a arquitetura de lakehouse com tabelas do Apache Iceberg e, ao mesmo tempo, aproveitar a poderosa analytics SQL do Amazon Redshift em warehouses e lakes. Você também

recebe acesso imediato aos recursos avançados do Amazon Redshift, como visões materializadas em suas tabelas do Iceberg, aprimorando significativamente seus recursos analíticos sem aumentar a complexidade.

As gravações do Iceberg são aceitas em clusters provisionados do Amazon Redshift e em instâncias do Amazon Redshift sem servidor.

Tópicos

- Comandos SQL
- Alteração de definições de tabela
- Semântica de transação
- Práticas recomendadas

Comandos SQL

As tabelas do Apache Iceberg no Amazon Redshift oferecem uma forma poderosa de gerenciar grandes conjuntos de dados analíticos em seu data lake. Essas tabelas aceitam transações ACID, evolução de esquemas e recursos de viagem no tempo, mantendo a alta performance das workloads de analytics. Usando tabelas do Apache Iceberg, você pode organizar e particionar seus dados com eficiência, controlar formatos de arquivo e compactação e integrar-se perfeitamente a outros serviços da AWS.

É possível criar tabelas do Iceberg particionadas e não particionadas usando comandos `CREATE TABLE ... USING ICEBERG` e `CREATE TABLE ... USING ICEBERG AS SELECT`. Você pode fazer referência a tabelas do Iceberg usando a notação de esquema externo (`external_schema.table_name`) ou a notação em três partes (`"catalog_name".database_name.table_name`). Os exemplos nesta seção demonstram os dois métodos.

Depois de criar uma tabela, você pode adicionar dados usando os comandos `INSERT` padrão. Lembre-se de que, embora o Amazon Redshift funcione com muitos tipos de dados do Iceberg, talvez seja necessário converter alguns formatos de dados ao inserir informações.

Você pode visualizar as tabelas do Iceberg usando o comando `SHOW TABLES`. Se quiser remover uma tabela do AWS Glue Data Catalog, você poderá usar o comando `DROP TABLE`. Observe que isso só remove o registro da tabela. Os dados reais permanecerão armazenados até que você os exclua separadamente.

Também é possível modificar dados existentes usando os comandos DELETE, UPDATE e MERGE. Para alterar as definições da tabela, como esquema, especificações de partição e propriedades, consulte Alteração de definições de tabela. Outras instruções DDL não documentadas ali não são suportadas para tabelas do Iceberg.

É possível gravar em uma tabela Iceberg que não foi criada pelo Amazon Redshift. No entanto, existem algumas limitações:

- A tabela deve ser uma tabela do Iceberg v2 ou v3.
- A tabela deve usar o Parquet como formato de dados padrão.
- A tabela não deve ter a compactação de metadados definida como True.
- A tabela não deve habilitar Write-Audit-Publish (WAP).

As seções a seguir demonstram a sintaxe SQL para criar, inserir, modificar e gerenciar tabelas do Iceberg no Amazon Redshift.

Sumário

- CRIAR TABELA
- CRIAR TABELA COMO SELEÇÃO
- SHOW TABLE
- INSERT INTO
- DELETE
- UPDATE
- MERGE
- DESCARTAR TABELA

CRIAR TABELA

```
CREATE TABLE [IF NOT EXISTS] <external_schema>.<table_name> (
  column_name data_type [DEFAULT literal_value] [, ...]
)
USING ICEBERG
[LOCATION 's3://your-bucket-name/prefix/']
[PARTITIONED BY [[column_name | transform_function]], ...]
[TABLE PROPERTIES (['format-version'='<version>'] [,
 'compression_type'='<compression_value>'])]
```

Você também pode usar a notação em três partes para buckets de tabelas do S3:

```
CREATE TABLE "<table_bucket_name>@s3tablescatalog".<database_name>.<table_name> (
  column_name data_type [DEFAULT literal_value] [, ...]
)
USING ICEBERG
[PARTITIONED BY [[column_name | transform_function]], ...]
[TABLE PROPERTIES (['format-version'='<version>'] [,
 'compression_type'='<compression_value>'])]
```

Para o catálogo `awsdatacatalog` raiz montado automaticamente:

```
CREATE TABLE awsdatacatalog.<database_name>.<table_name> (
  column_name data_type [DEFAULT literal_value] [, ...]
)
USING ICEBERG
LOCATION 's3://your-bucket-name/prefix/'
[PARTITIONED BY [[column_name | transform_function]], ...]
[TABLE PROPERTIES (['format-version'='<version>'] [,
 'compression_type'='<compression_value>'])]
```

Ao usar a sintaxe de esquema externo, *<external_schema>* deve ser um nome de esquema externo existente no qual a tabela externa será criada. Para acessar mais informações sobre como criar e gerenciar esquemas externos, consulte CREATE EXTERNAL SCHEMA na documentação do Amazon Redshift.

A cláusula `LOCATION` define a localização desta tabela do Iceberg recém-criada. `LOCATION` é obrigatório para tabelas criadas usando esquemas externos ou o catálogo `awsdatacatalog` raiz. Deve ser um local vazio, o que significa que não há objetos existentes do Amazon S3 compartilhando esse mesmo bucket e prefixo. A região do bucket do Amazon S3 deve ser a mesma região do cluster do Amazon Redshift. Para buckets de tabela do Amazon S3, `LOCATION` não pode ser especificado, pois a localização da tabela é determinada pelo catálogo de tabelas do Amazon S3 (`s3tablescatalog`).

No entanto, a AWS fornece um método para replicar dados de tabelas do Iceberg armazenadas em um AWS Glue Data Catalog em uma Região da AWS para uma Região da AWS diferente, permitindo replicar a gravação para uma região diferente. Para acessar mais informações, consulte Replicar seus dados entre Regiões da AWS.

`PARTITIONED BY` define a partição de tabelas do Iceberg. O Amazon Redshift comporta todas as transformações de partição do Iceberg v2, exceto `void`. Veja a lista de transformações aceitas:

- identity
- bucket[N]
- truncate[W]
- ano
- mês
- dia
- hora

Para acessar as definições completas dessas transformações e os tipos de dados compatíveis, consulte Transformações de partição na documentação do Apache Iceberg.

`PARTITIONED BY` aceita particionamento em vários níveis. Por exemplo, é possível executar o seguinte comando:

```
CREATE TABLE ...
USING ICEBERG
LOCATION ...
PARTITIONED BY (bucket(16, id), year(ship_date));
```

No entanto, o Amazon Redshift não aceita o uso de uma única coluna em mais de uma transformação. Por exemplo, a seguinte sintaxe não é aceita:

```
CREATE TABLE ...
USING ICEBERG
LOCATION ...
PARTITIONED BY (bucket(16, ship_date), year(ship_date));
```

A cláusula `TABLE PROPERTIES` define as propriedades extras da tabela do Iceberg. As propriedades de tabela compatíveis são `format-version` e `compression_type`. A propriedade `format-version` especifica a versão do formato da tabela do Iceberg. Os valores possíveis são`'2'` [TAG] (padrão) e `'3'`. A propriedade `compression_type` define a compactação padrão dos arquivos de dados Parquet. Se não for especificada, `zstd` será usado como codec de compactação. Os valores possíveis para `compression_type` são `zstd`, `brotli`, `gzip`, `snappy` e `uncompressed`.

Para tabelas do Iceberg v3, é possível especificar valores padrão para colunas usando a palavra-chave DEFAULT. Os valores padrão de colunas são compatíveis somente com tabelas do Iceberg

v3. O Amazon Redshift retorna um erro se você especificar um valor padrão em uma tabela do Iceberg v2. Somente valores literais são aceitos como padrões.

```
CREATE TABLE <external_schema>.<table_name> (
  column_name data_type [DEFAULT literal_value] [, ...]
)
USING ICEBERG
LOCATION 's3://your-bucket-name/prefix/'
[PARTITIONED BY [[column_name | transform_function]], ...]
[TABLE PROPERTIES ('format-version'='3' [, 'compression_type'='<compression_value>'])];
```

Exemplos com valores padrão:

```
-- External schema notation
CREATE TABLE my_external_schema.orders (
  order_id INT,
  status VARCHAR DEFAULT 'pending',
  region VARCHAR DEFAULT 'us-east-1'
)
USING ICEBERG
LOCATION 's3://amzn-s3-demo-bucket/orders/'
TABLE PROPERTIES ('format-version'='3');

-- Three-part notation (awsdatacatalog / Glue)
CREATE TABLE awsdatacatalog.my_glue_db.orders (
  order_id INT,
  status VARCHAR DEFAULT 'pending',
  region VARCHAR DEFAULT 'us-east-1'
)
USING ICEBERG
LOCATION 's3://amzn-s3-demo-bucket/orders/'
TABLE PROPERTIES ('format-version'='3');

-- Three-part notation (S3 Table Buckets)
CREATE TABLE "amzn-s3-demo-bucket@s3tablescatalog".my_namespace.orders (
  order_id INT,
  status VARCHAR DEFAULT 'pending',
  region VARCHAR DEFAULT 'us-east-1'
)
USING ICEBERG
TABLE PROPERTIES ('format-version'='3');
```

> **Note**
>
> `CREATE TABLE ... LIKE ...` não é aceito em tabelas do Iceberg. As tabelas do Iceberg também não aceitam restrições de coluna e atributos de coluna, como a tabela RMS.

Como alternativa, você pode criar e preencher uma tabela do Iceberg em uma única operação usando `CREATE TABLE AS SELECT`:

CRIAR TABELA COMO SELEÇÃO

```
CREATE TABLE <external_schema>.<table_name> [(
  column_name[, ...]
)]
USING ICEBERG
[LOCATION 's3://your-bucket-name/prefix/']
[PARTITIONED BY [[column_name | transform_function]], ...]
[TABLE PROPERTIES ('compression_type'='<compression-value>')]
AS
SELECT query
```

Você também pode usar a notação em três partes para criar tabelas em catálogos montados automaticamente:

```
CREATE TABLE "<catalog_name>".<database_name>.<table_name> [(
  column_name[, ...]
)]
USING ICEBERG
[LOCATION 's3://your-bucket-name/prefix/']
[PARTITIONED BY [[column_name | transform_function]], ...]
[TABLE PROPERTIES ('compression_type'='<compression-value>')]
AS
SELECT query
```

Isso é semelhante à instrução `CREATE TABLE`, exceto que `CREATE` é seguida por uma instrução `SELECT` para preencher a tabela com os resultados da consulta `SELECT`.

Não é mais possível especificar os tipos de dados com a cláusula `CREATE TABLE` aqui, porque a consulta `SELECT` determina os tipos de dados das colunas.

Se a consulta SELECT falhar por algum motivo, essa consulta falhará, e a tabela do Iceberg não será criada.

Para tabelas do Iceberg v3, especifique `'format-version'='3'` na cláusula TABLE PROPERTIES. Os valores padrão das colunas não são herdados da tabela de origem.

```
-- External schema notation
CREATE TABLE my_external_schema.orders_backup (
  order_id, status, region
)
USING ICEBERG
LOCATION 's3://amzn-s3-demo-bucket/orders-backup/'
TABLE PROPERTIES ('format-version'='3')
AS
SELECT order_id, status, region
FROM my_external_schema.orders;

-- Three-part notation (awsdatacatalog / Glue)
CREATE TABLE awsdatacatalog.my_glue_db.orders_backup (
  order_id, status, region
)
USING ICEBERG
LOCATION 's3://amzn-s3-demo-bucket/orders-backup/'
TABLE PROPERTIES ('format-version'='3')
AS
SELECT order_id, status, region
FROM awsdatacatalog.my_glue_db.orders;

-- Three-part notation (S3 Table Buckets)
CREATE TABLE "amzn-s3-demo-bucket@s3tablescatalog".my_namespace.orders_backup (
  order_id, status, region
)
USING ICEBERG
TABLE PROPERTIES ('format-version'='3')
AS
SELECT order_id, status, region
FROM "amzn-s3-demo-bucket@s3tablescatalog".my_namespace.orders;
```

Você pode visualizar a estrutura de suas tabelas do Iceberg usando `SHOW TABLE`:

SHOW TABLE

```
SHOW TABLE <external_schema>.<table_name>
```

Também é possível usar notação em três partes com catálogos montados automaticamente:

```
SHOW TABLE "<catalog_name>".<database_name>.<table_name>
```

`SHOW TABLE` exibe a instrução `CREATE TABLE` para a tabela do Iceberg. O comando mostrará os resultados apropriados com base no tipo da tabela. Veja um exemplo da saída `SHOW TABLE` da tabela do Iceberg:

```
CREATE TABLE my_schema.items (id int, price decimal(5, 2))
USING ICEBERG
LOCATION 's3://my_s3_bucket/items/'
PARTITIONED BY (bucket(16, id))
TABLE PROPERTIES ('compression_type'='snappy')
```

> **Note**
>
> Para tabelas do Amazon S3, como a localização da tabela é gerenciada pelo catálogo de tabelas do Amazon S3, a cláusula `LOCATION` será omitida nos resultados de `SHOW TABLE`.

Para tabelas do Iceberg v3, `SHOW TABLE` exibe os valores padrão das colunas em sua saída.

```
SHOW TABLE my_external_schema.orders;

-- Output includes the default values in the column definitions:
CREATE TABLE my_external_schema.orders (
  id int,
  status varchar DEFAULT 'active',
  priority int DEFAULT 0
)
USING ICEBERG
LOCATION 's3://amzn-s3-demo-bucket/orders/'
TABLE PROPERTIES ('format-version'='3');
```

Depois de criar tabelas, é possível adicionar dados usando `INSERT INTO`:

INSERT INTO

```
INSERT INTO <external_schema>.<table_name> [(column_name [, ...])] VALUES (...)
INSERT INTO <external_schema>.<table_name> [(column_name [, ...])] (SELECT query)

-- Using three-part notation for S3 table buckets:
INSERT INTO "<table_bucket_name>@s3tablescatalog".<database_name>.<table_name>
 [(column_name [, ...])] VALUES (...)
INSERT INTO "<table_bucket_name>@s3tablescatalog".<database_name>.<table_name>
 [(column_name [, ...])] (SELECT query)

-- Using three-part notation for the awsdatacatalog root catalog:
INSERT INTO awsdatacatalog.<database_name>.<table_name> [(column_name [, ...])] VALUES
 (...)
INSERT INTO awsdatacatalog.<database_name>.<table_name> [(column_name [, ...])] (SELECT
 query)
```

Você pode `INSERT INTO` a tabela do Iceberg existente usando a sintaxe acima. Se a cláusula `VALUES` for usada, você fornecerá os valores das colunas listadas por `column_name`, ou de todas as colunas, se a parte `column_name` for omitida.

Quando os dados são inseridos na tabela particionada, novas linhas são distribuídas de acordo com a especificação de partição predefinida. Se, por algum motivo, a consulta `SELECT` falhar, a consulta falhará, e nenhum dado será inserido na tabela do Iceberg.

Para tabelas do Iceberg v3 com valores padrão de colunas, se uma coluna com um valor padrão for omitida da instrução INSERT ou se DEFAULT for especificado como o valor, o valor padrão será gravado no arquivo de dados.

DELETE

A consulta `DELETE` para a tabela Iceberg usa a sintaxe `DELETE` existente na tabela do RMS.

```
[ WITH [RECURSIVE] common_table_expression [, common_table_expression , ...] ]
DELETE [ FROM ] iceberg_table
[ { USING } table_name, ... ] [ WHERE condition ]
```

Você também pode usar a notação em três partes para buckets de tabelas do S3:

```
[ WITH [RECURSIVE] common_table_expression [, common_table_expression , ...] ]
DELETE [ FROM ] "<table_bucket_name>@s3tablescatalog".<database_name>.<table_name>
```

```
[ { USING } table_name, ... ] [ WHERE condition ]
```

Para o catálogo `awsdatacatalog` raiz montado automaticamente:

```
[ WITH [RECURSIVE] common_table_expression [, common_table_expression , ...] ]
DELETE [ FROM ] awsdatacatalog.<database_name>.<table_name>
[ { USING } table_name, ... ] [ WHERE condition ]
```

O *iceberg_table* pode ser referenciado usando o formato *<external_schema>*.*<external_table_name>*, ou utilizando a notação de três partes para catálogo montado automaticamente. Consulte Referencing Iceberg tables in Amazon Redshift.

O *table_name* na cláusula `USING` será usado para realizar junção com a tabela de destino para excluir linhas que atendam à condição `WHERE`. O pode *table_name* ser uma tabela Iceberg ou uma tabela RMS do Amazon Redshift.

Como o Iceberg usa um esquema de particionamento oculto, é possível usar a consulta `DELETE` para remover partições, obtendo o mesmo efeito de `ALTER TABLE ... DROP PARTITION ...` para tabelas Hive.

Por exemplo, considere uma tabela do Iceberg particionada como a seguinte:

```
CREATE TABLE my_external_schema.lineitem
(l_item_id int,
 l_ship_date varchar,
 ...
)
USING ICEBERG
LOCATION ...
PARTITIONED BY l_ship_date;
```

Em seguida, é possível remover uma partição usando uma consulta como a seguinte:

```
DELETE FROM my_external_schema.lineitem WHERE l_ship_date = '20251231';
```

Para uma consulta como esta, o Amazon Redshift otimiza a execução para realizar apenas uma operação de metadados e encerrar a execução antecipadamente. Assim, diferentemente de uma consulta `DELETE` normal, a consulta de exclusão apenas de metadados não exibe etapas de execução em `EXPLAIN`:

```
explain DELETE FROM my_external_schema.lineitem WHERE l_ship_date = '20251231';

 QUERY PLAN
------------
"XN Seq Scan Metadata of my_external_schema.lineitem location: "s3://s3-path//table-
location" format:ICEBERG (cost=0.00..0.01 rows=0 width=0)"
(0 rows)
```

UPDATE

A sintaxe da consulta UPDATE para tabela Iceberg é muito semelhante à sintaxe existente de UPDATE para tabela RMS:

```
[ WITH [RECURSIVE] common_table_expression [, common_table_expression , ...] ]
UPDATE iceberg_table [ [ AS ] alias ] SET column = { expression } [,...]
[ FROM fromlist ]
[ WHERE condition ]
```

Você também pode usar a notação em três partes para buckets de tabelas do S3:

```
[ WITH [RECURSIVE] common_table_expression [, common_table_expression , ...] ]
UPDATE "<table_bucket_name>@s3tablescatalog".<database_name>.<table_name> [ [ AS ]
 alias ] SET column = { expression } [,...]
[ FROM fromlist ]
[ WHERE condition ]
```

Para o catálogo `awsdatacatalog` raiz montado automaticamente:

```
[ WITH [RECURSIVE] common_table_expression [, common_table_expression , ...] ]
UPDATE awsdatacatalog.<database_name>.<table_name> [ [ AS ] alias ] SET column =
 { expression } [,...]
[ FROM fromlist ]
[ WHERE condition ]
```

O *iceberg_table* pode ser referenciado usando o formato *<external_schema>*.*<external_table_name>*, ou utilizando a notação de três partes para catálogo montado automaticamente. Consulte Referencing Iceberg tables in Amazon Redshift.

Você pode atualizar uma tabela fazendo referência às informações em outras tabelas. Liste essas outras tabelas na cláusula FROM ou use uma subconsulta como parte da condição WHERE. As tabelas de origem podem ser tabelas Iceberg ou tabelas RMS do Amazon Redshift.

O UPDATE também pode ser executado em tabela particionada. Quando UPDATE altera valores de colunas que pertencem à especificação de partição atual, a nova linha atualizada será inserida na nova partição com base no valor atualizado.

Por exemplo, considere uma tabela do Iceberg particionada como a seguinte:

```
CREATE TABLE my_external_schema.lineitem
(l_item_id int,
 l_ship_date varchar,
 ...
)
USING ICEBERG
LOCATION ...
PARTITIONED BY l_ship_date;

INSERT INTO my_external_schema.lineitem VALUES (10099, '20251231', ...);
```

E, quando você executar a seguinte consulta de atualização:

```
UPDATE my_external_schema.lineitem SET l_ship_date = '20260101'
WHERE l_item_id = 10099;
```

a linha com `l_item_id` 10099 será movida da partição `20251231` para a nova partição `20260101`.

Também é importante observar que é possível que UPDATE tenha múltiplos valores candidatos. Considere a consulta abaixo:

```
CREATE TABLE my_ext_schema.t1(x1 int, y1 int) USING ICEBERG LOCATION ...;
CREATE TABLE my_ext_schema.t2(x2 int, y2 int) USING ICEBERG LOCATION ...;
INSERT INTO my_ext_schema.t1 VALUES (1,10), (2,20), (3,30);
INSERT INTO my_ext_schema.t2 VALUES (2,40), (2,50);
UPDATE my_ext_schema.t1 SET y1=y2 FROM my_ext_schema.t2 WHERE x1=x2;
```

Nesse caso, y1 pode ser 40 ou 50. O resultado é não determinístico. É possível definir o parâmetro de configuração `error_on_nondeterministic_update` como true para forçar erro na consulta quando esse caso ocorrer. Isso é consistente com o comportamento UPDATE existente de em tabelas RMS. Consulte mais informações em error_on_nondeterministic_update.

MERGE

A consulta MERGE mescla condicionalmente linhas de uma tabela de origem em uma tabela de destino. Ela compartilha a mesma sintaxe de consulta MERGE que a tabela RMS existente:

```
MERGE INTO target_iceberg_table USING source_table [ [ AS ] alias ]
ON match_condition
[ WHEN MATCHED THEN { UPDATE SET col_name = { expr } [,...] | DELETE }
  WHEN NOT MATCHED THEN INSERT [ ( col_name [,...] ) ]
  VALUES ( { expr } [, ...] )
| REMOVE DUPLICATES ]
```

Você também pode usar a notação em três partes para buckets de tabelas do S3:

```
MERGE INTO "<table_bucket_name>@s3tablescatalog".<database_name>.<table_name>
 USING source_table [ [ AS ] alias ]
ON match_condition
[ WHEN MATCHED THEN { UPDATE SET col_name = { expr } [,...] | DELETE }
  WHEN NOT MATCHED THEN INSERT [ ( col_name [,...] ) ]
  VALUES ( { expr } [, ...] )
| REMOVE DUPLICATES ]
```

Para o catálogo awsdatacatalog raiz montado automaticamente:

```
MERGE INTO awsdatacatalog.<database_name>.<table_name> USING source_table
 [ [ AS ] alias ]
ON match_condition
[ WHEN MATCHED THEN { UPDATE SET col_name = { expr } [,...] | DELETE }
  WHEN NOT MATCHED THEN INSERT [ ( col_name [,...] ) ]
  VALUES ( { expr } [, ...] )
| REMOVE DUPLICATES ]
```

O *target_iceberg_table* pode ser referenciado usando o formato *<external_schema>.<external_table_name>*, ou utilizando a notação de três partes para catálogo montado automaticamente. Consulte Referencing Iceberg tables in Amazon Redshift.

O *source_table* pode ser uma tabela Iceberg ou uma tabela RMS do Amazon Redshift.

Quando REMOVE DUPLICATES é utilizado, o comando MERGE usa o modo simplificado. Consulte mais detalhes sobre o modo simplificado na documentação original do comando MERGE.

Ao executar a consulta MERGE, o Amazon Redshift gera e armazena arquivos de dados intermediários no local da tabela de destino. Esses arquivos serão coletados ao final da consulta. Por esse motivo, a consulta MERGE requer a permissão DELETE no bucket do Amazon S3 para funcionar corretamente. Um erro de permissão insuficiente será gerado caso a operação de coleta de resíduos não seja concluída com sucesso. Para tabelas do Amazon S3, a coleta de resíduos é gerenciada pelo serviço Tabelas do Amazon S3. Portanto, a permissão DELETE não é necessária para executar a consulta MERGE.

DESCARTAR TABELA

Para remover uma tabela do Iceberg do catálogo, use o comando DROP TABLE:

```
DROP TABLE <external_schema>.<table_name>
```

Também é possível usar notação em três partes com catálogos montados automaticamente:

```
DROP TABLE "<catalog_name>".<database_name>.<table_name>
```

Eliminar uma tabela do Iceberg é uma operação somente para metadados. É removida a entrada da tabela do AWS Glue Data Catalog e um catálogo de tabelas do Amazon S3, se for uma tabela do Amazon S3. O Amazon Redshift não limpa nem exclui nenhum arquivo de dados ou arquivos de metadados existentes na localização da tabela. Você pode usar recursos no AWS Glue e em tabelas do Amazon S3 para remover arquivos órfãos. Para AWS Glue, consulte Excluir arquivos órfãos. Para tabelas do Amazon S3, consulte Manutenção de tabelas.

Alteração de definições de tabela

As instruções ALTER TABLE são usadas para alterar as definições de tabela, incluindo esquema, especificações de partição e propriedades. Você pode usar essas instruções DDL para gerenciar suas tabelas do Iceberg diretamente por meio do Amazon Redshift.

Todas as instruções ALTER TABLE oferecem suporte ao uso de referências externas de esquema ou referências de notação em três partes para tabelas do Iceberg. Nos exemplos de sintaxe abaixo, *iceberg_table* pode ser qualquer um dos seguintes:

```
-- External schema notation:
<external_schema>.<table_name>

-- Three-part notation for S3 table buckets:
```

```
"<table_bucket_name>@s3tablescatalog".<database_name>.<table_name>

-- Three-part notation for the awsdatacatalog root catalog:
awsdatacatalog.<database_name>.<table_name>
```

Para obter mais informações, consulte Fazer referência a tabelas do Iceberg no Amazon Redshift.

Considere o seguinte ao usar instruções `ALTER TABLE`:

- As instruções `ALTER TABLE` atualmente são compatíveis somente com tabelas do Iceberg v2 ou v3.
- Todas as instruções `ALTER TABLE` são operações somente de metadados.
- As instruções `ALTER TABLE` não oferecem suporte a tabelas com colunas de tipo complexo.

Sumário

- Atualizar para o Iceberg v3
- ALTER TABLE RENAME COLUMN
- ALTER TABLE ADD/DROP COLUMN
- ALTER TABLE ALTER COLUMN
- ALTER TABLE SET TABLE PROPERTIES
- ALTER TABLE ADD, DROP e REPLACE PARTITION FIELD

Atualizar para o Iceberg v3

É possível atualizar uma tabela do Iceberg v2 para a v3 usando `ALTER TABLE SET TABLE PROPERTIES`:

```
ALTER TABLE iceberg_table
SET TABLE PROPERTIES ('format-version' = '3');
```

Para obter detalhes adicionais, consulte the section called “Atualização de v2 para v3”.

ALTER TABLE RENAME COLUMN

```
ALTER TABLE iceberg_table
RENAME [COLUMN] col_name TO new_name;
```

`ALTER TABLE RENAME COLUMN` renomeia uma coluna existente. *`col_name`* pode ser uma coluna de partição ou uma coluna não particionada. Após a renomeação, o tipo de dados da coluna e a especificação da partição não mudam.

> **Note**
>
> `ALTER TABLE RENAME COLUMN` não é compatível com tabelas do AWS Lake Formation registradas.

ALTER TABLE ADD/DROP COLUMN

```
ALTER TABLE iceberg_table
ADD [COLUMN] col_name col_type;
```

```
ALTER TABLE iceberg_table
DROP [COLUMN] col_name;
```

`ADD COLUMN` adiciona uma coluna a uma tabela do Iceberg existente. Você pode usar qualquer tipo de dados compatível com as gravações do Iceberg do Amazon Redshift. Para obter mais informações, consulte Tipos de dados compatíveis com tabelas do Apache Iceberg.

`ADD COLUMN` é uma operação somente de metadados. Os valores das colunas recém-adicionadas nas linhas existentes são `NULL`.

`DROP COLUMN` descarta uma coluna de uma tabela do Iceberg existente. Para tabelas particionadas, você não pode descartar uma coluna que pertença à especificação de partição atual. Você deve primeiro remover o campo de partição que envolve a coluna antes de descartá-la. Para obter mais informações, consulte ALTER TABLE ADD, DROP e REPLACE PARTITION FIELD.

Para tabelas do Iceberg v3, você pode especificar um valor padrão ao adicionar uma coluna. Os arquivos de dados existentes retornam o valor padrão especificado, em vez de NULL, para a coluna recém-adicionada, sem exigir uma regravação dos dados. Os valores padrão de colunas são compatíveis somente com tabelas do Iceberg v3. O Amazon Redshift retorna um erro se você especificar um valor padrão em uma tabela do Iceberg v2.

```
-- External schema notation
ALTER TABLE my_external_schema.orders
ADD COLUMN priority INT DEFAULT 0;
```

```
-- Three-part notation (awsdatacatalog / Glue)
ALTER TABLE awsdatacatalog.my_glue_db.orders
ADD COLUMN priority INT DEFAULT 0;

-- Three-part notation (S3 Table Buckets)
ALTER TABLE "amzn-s3-demo-bucket@s3tablescatalog".my_namespace.orders
ADD COLUMN priority INT DEFAULT 0;
```

Para as linhas existentes, o valor padrão é preenchido na coluna recém-adicionada:

```
-- Existing table with rows
SELECT id FROM my_external_schema.orders;
--  1
--  2

-- Add a column with a default
ALTER TABLE my_external_schema.orders
ADD COLUMN status varchar DEFAULT 'active';

-- Existing rows return the default without a data rewrite
SELECT id, status FROM my_external_schema.orders;
--  1 | active
--  2 | active
```

ALTER TABLE ALTER COLUMN

```
ALTER TABLE iceberg_table
    ALTER COLUMN column_name TYPE updated_data_type;
```

`ALTER TABLE ALTER COLUMN` altera o tipo de dados de uma coluna existente. Somente a ampliação de tipo é permitida, não a redução. Como os dados não são regravados após `ALTER`, reduzir os tipos de dados pode causar estouro ao consumir os dados de tabela existentes. As seguintes ampliações de tipo são permitidas de acordo com a especificação do Iceberg:

```
int # bigint
float # double
decimal(P, S) # decimal(P2, S) where P2 > P (scale cannot be changed)
```

No Amazon Redshift, os tipos de dados podem ter aliases. Por exemplo, números inteiros de 4 bytes usam o nome do tipo `int` ou `int4`. Desde que sejam mapeados para inteiros de 4 bytes nos tipos Iceberg, a ampliação do tipo para inteiros de 8 bytes é permitida.

Por exemplo, uma tabela do Iceberg é criada usando tipos inteiros de 4 bytes:

```
CREATE TABLE iceberg_table (cint int) USING ICEBERG LOCATION 's3://your-bucket-name/
prefix/';
```

Ela pode ser ampliada por esta instrução ALTER:

```
ALTER TABLE iceberg_table ALTER COLUMN cint TYPE int8;
```

Para obter a lista completa de mapeamentos de tipos de dados entre os tipos do Amazon Redshift e os tipos Iceberg, consulte Tipos de dados compatíveis com tabelas do Apache Iceberg.

Não há suporte para ampliar o tipo de coluna que pertence à especificação de partição existente.

Para tabelas do Iceberg v3, você pode definir ou remover um valor padrão em uma coluna existente:

```
ALTER TABLE iceberg_table
ALTER COLUMN column_name SET DEFAULT literal_value;

ALTER TABLE iceberg_table
ALTER COLUMN column_name DROP DEFAULT;
```

SET DEFAULT altera o valor padrão de uma coluna existente. O novo valor padrão se aplica aos arquivos de dados gravados após a alteração. Os arquivos de dados existentes continuam usando o valor padrão anterior.

DROP DEFAULT remove o valor padrão aplicado a novas gravações, o que equivale a definir o padrão como NULL. Os arquivos de dados que não contêm a coluna continuam retornando o valor padrão inicial da coluna.

```
-- External schema notation
ALTER TABLE my_external_schema.orders
ALTER COLUMN status SET DEFAULT 'active';
ALTER TABLE my_external_schema.orders
ALTER COLUMN status DROP DEFAULT;

-- Three-part notation (awsdatacatalog / Glue)
ALTER TABLE awsdatacatalog.my_glue_db.orders
ALTER COLUMN status SET DEFAULT 'active';
ALTER TABLE awsdatacatalog.my_glue_db.orders
ALTER COLUMN status DROP DEFAULT;
```

```
-- Three-part notation (S3 Table Buckets)
ALTER TABLE "amzn-s3-demo-bucket@s3tablescatalog".my_namespace.orders
ALTER COLUMN status SET DEFAULT 'active';
ALTER TABLE "amzn-s3-demo-bucket@s3tablescatalog".my_namespace.orders
ALTER COLUMN status DROP DEFAULT;
```

ALTER TABLE SET TABLE PROPERTIES

```
ALTER TABLE iceberg_table
SET TABLE PROPERTIES (
 'compression_type' = 'compression_value');
```

Essa instrução permite que você substitua a propriedade padrão da tabela. Atualmente, a única propriedade de tabela permitida para essa instrução é `compression_type`. Você pode substituí-la para usar um tipo de compactação diferente para os arquivos Parquet da tabela do Iceberg. Os dados inseridos após `ALTER` usam o novo tipo de compactação.

Os valores possíveis para `compression_type` são `brotli`, `gzip`, `snappy`, `uncompressed` e `zstd`.

Você também pode atualizar uma tabela do Iceberg v2 existente para a v3 definindo a versão do formato:

```
ALTER TABLE iceberg_table
SET TABLE PROPERTIES ('format-version' = '3');
```

Para obter detalhes adicionais, consulte the section called “Atualização de v2 para v3”.

ALTER TABLE ADD, DROP e REPLACE PARTITION FIELD

```
ALTER TABLE iceberg_table
    ADD PARTITION FIELD column_name | transform_function;
```

```
ALTER TABLE iceberg_table
    DROP PARTITION FIELD column_name | transform_function;
```

```
ALTER TABLE iceberg_table
    REPLACE PARTITION FIELD column_name | transform_function
```

```
        WITH column_name | transform_function;
```

As instruções `[ADD | DROP | REPLACE] PARTITION FIELD` permitem que você altere a especificação existente da partição da tabela, dando suporte à evolução da partição por meio do Amazon Redshift.

As instruções `ALTER` para especificação de partição alteram apenas os metadados da tabela e não reparticionam os dados de tabela existentes. Depois de `ALTER`, os novos dados inseridos na tabela seguem a especificação de partição recém-definida.

Considere as seguintes limitações para essas instruções:

- Ao definir um novo campo de partição por meio de `ADD` ou `REPLACE ... WITH ...`, o novo campo não pode incluir uma coluna que já faça parte de outros campos de partição. Essa é a mesma limitação de quando você define a especificação de partição inicial em `CREATE TABLE`. Para obter mais informações, consulte CRIAR TABELA.

  Por exemplo, quando você tem uma tabela:

  ```
  CREATE TABLE iceberg_table ... PARTITIONED BY year(ship_date) USING ICEBERG ...;
  ```

  O seguinte falha porque `ship_date` já faz parte de um campo de partição existente:

  ```
  ALTER TABLE iceberg_table ADD PARTITION FIELD bucket(128, ship_date);
  ```

- Ao adicionar um novo campo de partição, o campo recém-adicionado sempre é tratado como o último nível de partição da tabela. Por exemplo:

  ```
  CREATE TABLE iceberg_table ... PARTITIONED BY year(ship_date) USING ICEBERG ...;
  ALTER TABLE iceberg_table ADD PARTITION FIELD bucket(256, item_id);
  ```

  A especificação da partição de tabela é a mesma que:

  ```
  CREATE TABLE iceberg_table ... PARTITIONED BY (year(ship_date), bucket(256, item_id))
      USING ICEBERG ...;
  ```

- Ao descartar um campo de partição, ele não se limita ao último nível de partição. Você pode eliminar qualquer campo de partição existente. Por exemplo:

  ```
  CREATE TABLE iceberg_table ... PARTITIONED BY (year(ship_date), bucket(256, item_id))
  ```

```
    USING ICEBERG ...;
ALTER TABLE iceberg_table DROP PARTITION FIELD year(ship_date);
```

Depois dessa instrução `ALTER`, a tabela é particionada somente por `bucket(256, item_id)`.

- Ao chamar `REPLACE PARTITION FIELD ... WITH ...`, o campo de partição a ser substituído pode ser qualquer campo na especificação e não está limitado ao último campo. Por exemplo:

```
CREATE TABLE iceberg_table ... PARTITIONED BY (year(ship_date), bucket(256, item_id))
    USING ICEBERG ...;
ALTER TABLE iceberg_table REPLACE PARTITION FIELD year(ship_date) WITH
 month(ship_date);
```

Depois dessa instrução `ALTER`, a partição da tabela se torna `(month(ship_date), bucket(256, item_id))`.

- A transformação `void` não é suportada em instruções de partição `ALTER`.

Semântica de transação

As consultas de gravação do Redshift Iceberg comportam ACID e isolamento de snapshot. As transações de gravação têm atomicidade garantida e não produzem atualizações parciais quando uma consulta falha inesperadamente.

Várias transações do Iceberg podem ser executadas simultaneamente e, se duas transações tentarem modificar a mesma tabela ou partição simultaneamente, a confirmação da transação falhará. Isso garante a integridade dos dados. Quando isso acontece, você deve resolver os conflitos manualmente e executar novamente as consultas com falha. O Amazon Redshift não tenta resolver automaticamente os conflitos novamente.

Uma única consulta de gravação do Iceberg é sempre tratada como uma única transação de confirmação automática. Quando uma consulta de gravação do Iceberg, como a consulta CREATE ou INSERT, é incluída em um bloco de transação explícito, nenhuma outra consulta pode ser executada no mesmo bloco de transação. A transação falhará.

Veja a seguir alguns exemplos: O primeiro exemplo demonstra que uma única consulta de instrução sempre é confirmada automaticamente após o término da consulta. Neste cenário, você está criando uma tabela de pedidos de venda:

```
CREATE TABLE sales_schema.orders (
    order_id int,
```

```
    customer_id int,
    order_date date,
    total_amount decimal(10,2)
) USING ICEBERG LOCATION 's3://my-data-lake/sales/orders/';
```

Este exemplo é um bloco de transação explícito para inserir um pedido de cliente usando a notação em três partes para um bucket de tabelas do S3. A transação não é confirmada automaticamente após a consulta INSERT, mas confirma e insere os dados do pedido com o comando COMMIT:

```
BEGIN;
INSERT INTO "analytics_bucket@s3tablescatalog".sales_db.orders VALUES (12345, 9876,
 '2024-10-30', 299.99);
COMMIT;
```

Este exemplo é um cenário explícito de reversão do bloco de transações em que você está testando a inserção de um pedido, mas decide cancelá-la. A transação não é confirmada automaticamente após a consulta INSERT, mas é revertida com o comando ROLLBACK sem inserir o pedido de teste.

```
BEGIN;
INSERT INTO sales_schema.orders VALUES (12346, 5432, '2024-10-30', 150.75);
ROLLBACK;
```

Este exemplo final demonstra como, quando você tenta executar outra instrução dentro do mesmo bloco de transação da consulta INSERT, a transação falha sem inserir os dados do pedido. Neste cenário, você está tentando inserir um pedido e consultar imediatamente a tabela:

```
BEGIN;
INSERT INTO sales_schema.orders VALUES (12347, 7890, '2024-10-30', 425.50);
SELECT * FROM sales_schema.orders WHERE order_id = 12347;
```

A única exceção a isso é a instrução `DROP TABLE`, que sempre se comporta como uma instrução de confirmação automática e não pode ser executada em um bloco de transação explícito. O objetivo disso é manter o mesmo comportamento que `DROP TABLE` em uma tabela externa. Consulte mais informações em DROP TABLE.

> **Note**
>
> Os SQLs de gravação do Iceberg não podem ser executados a partir do procedimento armazenado.

Práticas recomendadas

Pense nas seguintes práticas recomendadas ao gravar em uma tabela do Apache Iceberg:

- Para workloads de gravação ou streaming pequenas e frequentes, pense em usar os recursos de compactação fornecidos pelo AWS Glue Data Catalog ou pelas tabelas do Amazon S3 para otimizar o tamanho dos arquivos para leituras.

- O comando `DROP TABLE` cancela o registro da tabela do AWS Glue Data Catalog ou do catálogo de tabelas do Amazon S3, mas seus arquivos ainda permanecem. Você pode usar recursos no AWS Glue e em tabelas do Amazon S3 para remover arquivos órfãos. Para AWS Glue, consulte Excluir arquivos órfãos. Para tabelas do Amazon S3, consulte Manutenção de tabelas.

Recursos do Apache Iceberg v3 no Amazon Redshift

O Amazon Redshift é compatível com tabelas do Apache Iceberg v3. O Iceberg v3 introduz valores padrão de colunas, rastreamento da linhagem de linhas e vetores de exclusão. É possível criar tabelas da v3 ou atualizar tabelas da v2 existentes para a v3.

As tabelas do Iceberg v3 usam a mesma sintaxe SQL para consultas e operações DML (INSERT, DELETE, UPDATE, MERGE) que as tabelas da v2. Os recursos descritos nesta página são específicos das tabelas do Iceberg v3.

Criar uma tabela do Iceberg v3

Para criar uma tabela do Iceberg v3, especifique `'format-version'='3'` na cláusula TABLE PROPERTIES:

```
CREATE TABLE external_schema.table_name (
  column_name data_type [DEFAULT literal_value] [, ...]
)
USING ICEBERG
LOCATION 's3://your-bucket-name/prefix/'
[PARTITIONED BY [[column_name | transform_function]], ...]
TABLE PROPERTIES ('format-version'='3'
    [, 'compression_type'='compression_value']);
```

Exemplo:

```
CREATE TABLE my_schema.orders (
  id int,
  status varchar DEFAULT 'pending',
  priority int DEFAULT 0
)
USING ICEBERG
LOCATION 's3://amzn-s3-demo-bucket/orders/'
TABLE PROPERTIES ('format-version'='3');
```

Se você não especificar format-version, o Amazon Redshift criará a tabela como Iceberg v2.

Atualização de v2 para v3

É possível atualizar uma tabela do Iceberg v2 existente para a v3:

```
ALTER TABLE iceberg_table
SET TABLE PROPERTIES ('format-version' = '3');
```

A atualização da versão do formato é uma operação somente para os metadados. Os arquivos de dados existentes não são regravados. Os arquivos de exclusão posicional da v2 existentes permanecem válidos e são aplicados durante as leituras. A primeira operação de gravação após uma atualização gera valores de linhagem de linhas para toda a tabela. Nas operações subsequentes de DELETE, UPDATE ou MERGE, o Amazon Redshift combina as exclusões posicionais da v2 em vetores de exclusão para os arquivos de dados afetados pela operação.

> **Important**
>
> Antes de atualizar para o Iceberg v3, revise a seção the section called “Limitações”, pois alguns recursos ainda não têm suporte para tabelas da v3. Como não há suporte para fazer downgrade de v3 para v2, verifique se os recursos incompatíveis não afetarão as workloads antes de prosseguir com a atualização.

Limitações

- É possível usar o Iceberg v3 somente em clusters do Amazon Redshift sem servidor (exceto o de 4 RPU) e em clusters provisionados que usam tipos de instância RG.
- Não é possível ler nem gravar tipos complexos (struct, list, map, variant) em tabelas do Iceberg v3.

- Não é possível usar os seguintes tipos de dados em tabelas do Iceberg v3: struct, list, map, variant, geometry, geography, binary, uuid, time, timestamp_ns, timestamptz_ns e unknown.
- Não é possível usar tabelas do Iceberg v3 que contenham exclusões por igualdade.
- Não é possível criar visualizações materializadas em tabelas do Iceberg v3.
- Depois de atualizar uma tabela para a v3, o tipo `timestamptz` do Iceberg é mapeado para o tipo TIMESTAMPTZ do Amazon Redshift, em vez de TIMESTAMP (usado na v2). Como resultado, suas consultas retornam carimbos de data/hora que incluem informações de fuso horário.

Valores padrão das colunas

Os valores padrão de colunas são compatíveis somente com tabelas do Iceberg v3. O Amazon Redshift retorna um erro se você especificar um valor padrão em uma tabela do Iceberg v2.

Os valores padrão especificam um valor literal que uma coluna usa quando nenhum valor explícito é fornecido. Com isso, é possível adicionar novas colunas a uma tabela existente sem regravar os arquivos de dados. As leituras de arquivos de dados gravados anteriormente retornam automaticamente o valor padrão da nova coluna. O padrão também é gravado quando uma instrução DML omite a coluna ou especifica DEFAULT como o valor.

Somente valores literais são aceitos como padrões. Os tipos de dados aninhados não são compatíveis com valores padrão.

CREATE TABLE AS SELECT não herda os valores padrão das colunas da tabela de origem. Para definir padrões na nova tabela, use ALTER TABLE ALTER COLUMN SET DEFAULT após a criação.

Definição de padrões na criação da tabela

```
CREATE TABLE external_schema.table_name (
  column_name data_type DEFAULT literal_value [, ...]
)
USING ICEBERG
LOCATION '...'
TABLE PROPERTIES ('format-version'='3');
```

Adição de uma coluna com um padrão

```
ALTER TABLE iceberg_table
ADD COLUMN column_name data_type DEFAULT literal_value;
```

Os arquivos de dados existentes retornam o valor padrão da nova coluna adicionada sem exigir uma regravação dos dados.

Alteração ou remoção de um padrão

```
ALTER TABLE iceberg_table
ALTER COLUMN column_name SET DEFAULT literal_value;

ALTER TABLE iceberg_table
ALTER COLUMN column_name DROP DEFAULT;
```

SET DEFAULT altera o valor padrão de uma coluna existente. O novo valor padrão se aplica aos arquivos de dados gravados após a alteração. Os arquivos de dados existentes continuam usando o valor padrão anterior.

DROP DEFAULT remove o valor padrão de uma coluna. Depois dessa operação, novas gravações não aplicam mais um padrão à coluna. Os arquivos de dados que não contêm a coluna continuam retornando o valor padrão inicial da coluna.

Comportamento de INSERT

Ao inserir linhas sem especificar uma coluna que tenha um valor padrão, o valor padrão é gravado no arquivo de dados.

SHOW TABLE

SHOW TABLE exibe os valores padrão em sua saída.

Linhagem de linhas

A linhagem de linhas tem suporte somente para tabelas do Iceberg v3. Ela fornece duas pseudocolunas que rastreiam a identidade da linha e o histórico de modificações. Essas colunas são gerenciadas automaticamente pelo Amazon Redshift. Você não define seus valores.

Pseudocolunas

Coluna	Tipo de dados	Descrição
`_row_id`	BIGINT	Identifica exclusivamente cada linha da tabela. É atribuído automaticamente nas operações de gravação.

Coluna	Tipo de dados	Descrição
`_last_updated_sequence_number`	BIGINT	O número de sequência do snapshot da última operação de gravação que modificou a linha.

Consultar a linhagem de linhas

As colunas de linhagem de linhas devem ser nomeadas explicitamente na lista SELECT. Elas não são incluídas em SELECT *.

```
SELECT _row_id, _last_updated_sequence_number, *
FROM my_schema.my_iceberg_v3_table;
```

É possível usar colunas de linhagem de linhas nas cláusulas WHERE, ORDER BY, GROUP BY e JOIN:

```
SELECT *
FROM my_schema.my_iceberg_v3_table
WHERE _last_updated_sequence_number >= 3
ORDER BY _row_id;
```

Comportamento em tabelas que não são da v3

Ao consultar uma tabela que não seja do Iceberg v3, `_row_id` e `_last_updated_sequence_number` retornam NULL.

Comportamento após a atualização de v2 para v3

Para tabelas atualizadas do Iceberg v2 para v3, os valores de linhagem de linhas não ficam disponíveis imediatamente para os dados anteriores à atualização. Tanto `_row_id` quanto `_last_updated_sequence_number` retornam NULL até a primeira operação de gravação após a atualização, que gera valores de linhagem de linhas para toda a tabela. Essa é uma operação que altera somente os metadados e não regrava os arquivos de dados existentes.

Comportamento de gravação

A linhagem de linhas é preenchida automaticamente pelo Amazon Redshift em todas as operações de gravação (INSERT, CTAS, UPDATE, MERGE). Cada linha recebe um `_row_id` exclusivo, e

`_last_updated_sequence_number` reflete o número de sequência do snapshot do commit que gravou a linha.

Limitações

- As colunas de linhagem de linhas não são incluídas em SELECT *.
- A linhagem de linhas não é compatível com tabelas do Iceberg v2 ou anteriores.
- Os dados anteriores à atualização em tabelas atualizadas de v2 para v3 retornam NULL para ambas as pseudocolunas até a primeira operação de gravação após a atualização.

Vetores de exclusão

Os vetores de exclusão são o mecanismo usado pelo Iceberg v3 para rastrear exclusões em nível de linha. Eles substituem os arquivos de exclusão posicional usados pelo Iceberg v2.

Quando você executa DELETE, UPDATE ou MERGE em uma tabela do Iceberg v3, o Amazon Redshift registra as posições das linhas excluídas em vetores de exclusão, em vez de gravar arquivos de exclusão posicional separados. Isso é tratado automaticamente. Não são necessárias alterações na sintaxe SQL.

Como funcionam os vetores de exclusão

Um vetor de exclusão é um bitmap compactado que registra quais posições de linha em um arquivo de dados foram excluídas. Os vetores de exclusão são armazenados em arquivos Puffin no mesmo local do S3 que os dados da tabela. Cada arquivo de dados tem no máximo um vetor de exclusão.

Ao ler uma tabela do Iceberg v3, o Amazon Redshift aplica automaticamente os vetores de exclusão para excluir as linhas excluídas dos resultados da consulta.

Benefícios em relação aos arquivos de exclusão posicional

- Os vetores de exclusão são mais compactos do que os arquivos de exclusão posicional.
- Um único vetor de exclusão por arquivo de dados evita o acúmulo de muitos arquivos pequenos de exclusão.
- As exclusões subsequentes no mesmo arquivo de dados produzem um novo vetor de exclusão que combina as exclusões anteriores com as posições recém-excluídas, mantendo um único vetor de exclusão por arquivo de dados.

- Como os vetores de exclusão são mais compactos e evitam o acúmulo de muitos arquivos pequenos, as leituras e gravações são mais rápidas em comparação com os arquivos de exclusão posicional do Iceberg v2.

Comportamento após a atualização de v2 para v3

Depois de atualizar uma tabela do Iceberg de v2 para v3, os arquivos de exclusão posicional da v2 existentes permanecem válidos e são aplicados durante as leituras. Nas operações de gravação subsequentes (DELETE, UPDATE ou MERGE), o Amazon Redshift combina as exclusões posicionais v2 existentes em vetores de exclusão, conforme definido pela especificação do Iceberg.

Limitações

- Os vetores de exclusão não têm suporte para tabelas do Iceberg v2 ou anteriores.
- As tabelas do Iceberg v3 não podem voltar a usar arquivos de exclusão posicional para novas operações de gravação. Isso significa que as tabelas do Iceberg v3 podem ler arquivos de exclusão posicional existentes, mas nenhum novo arquivo de exclusão posicional pode ser adicionado. Em vez disso, as tabelas do Iceberg v3 só podem adicionar novas exclusões usando vetores de exclusão.

Performance de consulta do Amazon Redshift Spectrum

Este tópico descreve como melhorar a performance de consulta do Redshift Spectrum.

Verifique o plano da consulta para saber quais etapas foram enviadas para a camada do Amazon Redshift Spectrum.

As etapas a seguir são relacionadas com a consulta do Redshift Spectrum:

- S3 Seq Scan
- S3 HashAggregate
- S3 Query Scan
- Seq Scan PartitionInfo
- Partition Loop

O exemplo a seguir mostra o plano de uma consulta que une uma tabela externa a uma tabela local. Observe as etapas S3 Seq Scan e S3 HashAggregate que foram executadas com os dados do Amazon S3.

```
explain
select top 10 spectrum.sales.eventid, sum(spectrum.sales.pricepaid)
from spectrum.sales, event
where spectrum.sales.eventid = event.eventid
and spectrum.sales.pricepaid > 30
group by spectrum.sales.eventid
order by 2 desc;
```

```
QUERY PLAN


------------------------------------------------------------------------------
XN Limit  (cost=1001055770628.63..1001055770628.65 rows=10 width=31)


  ->  XN Merge  (cost=1001055770628.63..1001055770629.13 rows=200 width=31)


        Merge Key: sum(sales.derived_col2)


        ->  XN Network  (cost=1001055770628.63..1001055770629.13 rows=200 width=31)


              Send to leader


              ->  XN Sort  (cost=1001055770628.63..1001055770629.13 rows=200 width=31)


                    Sort Key: sum(sales.derived_col2)


                    ->  XN HashAggregate  (cost=1055770620.49..1055770620.99 rows=200
 width=31)
```

```
                          ->  XN Hash Join DS_BCAST_INNER  (cost=3119.97..1055769620.49
rows=200000 width=31)

                                Hash Cond: ("outer".derived_col1 = "inner".eventid)


                                ->  XN S3 Query Scan sales  (cost=3010.00..5010.50
rows=200000 width=31)

                                      ->  S3 HashAggregate  (cost=3010.00..3010.50
rows=200000 width=16)

                                            ->  S3 Seq Scan spectrum.sales
location:"s3://redshift-downloads/tickit/spectrum/sales" format:TEXT
(cost=0.00..2150.00 rows=172000 width=16)
                                                  Filter: (pricepaid > 30.00)


                                ->  XN Hash  (cost=87.98..87.98 rows=8798 width=4)


                                      ->  XN Seq Scan on event  (cost=0.00..87.98
rows=8798 width=4)
```

Observe os seguintes elementos no plano da consulta:

- O nó `S3 Seq Scan` mostra que o filtro `pricepaid > 30.00` foi processado na camada do Redshift Spectrum.

  Um nó de filtro sob o nó `XN S3 Query Scan` indica um processamento de predicado no Amazon Redshift, além dos dados obtidos da camada do Redshift Spectrum.
- O nó `S3 HashAggregate` indica uma agregação na camada do Redshift Spectrum para a cláusula de 'group by' (`group by spectrum.sales.eventid`).

As recomendações a seguir representam maneiras de melhorar a performance do Redshift Spectrum:

- Use arquivos de dados formatados pelo Apache Parquet. O Parquet armazena dados em um formato colunar, de maneira que o Redshift Spectrum pode eliminar colunas desnecessárias da varredura. Quando os dados estão em formato de arquivo de texto, o Redshift Spectrum precisa fazer a varredura do arquivo inteiro.

- Use vários arquivos para otimizar o processamento paralelo. Mantenha os tamanhos de seus arquivos maiores que 64 MB. Evite a distorção do tamanho dos dados mantendo os arquivos com aproximadamente o mesmo tamanho. Para ter informações sobre os arquivos e recomendações de configuração do Apache Parquet, consulte File Format: Configurations na documentação do Apache Parquet.
- Use o menor número possível de colunas nas consultas.
- Coloque as tabelas de fatos grandes no Amazon S3 e mantenha as tabelas menores utilizadas com maior frequência no banco de dados local do Amazon Redshift.
- Atualize as estatísticas das tabelas externas configurando o parâmetro numRows de TABLE PROPERTIES. Use CREATE EXTERNAL TABLE ou ALTER TABLE para definir o parâmetro TABLE PROPERTIES numRows para refletir o número de linhas na tabela. O Amazon Redshift não analisa as tabelas externas para gerar as estatísticas das tabelas que o otimizador de consultas utiliza para gerar um plano de consulta. Se as estatísticas da tabela não estiverem configuradas para uma tabela externa, o Amazon Redshift gerará um plano de execução de consulta. O Amazon Redshift gera esse plano baseado em uma suposição de que as tabelas externas são as maiores e as tabelas locais são as menores.
- O planejador de consultas do Amazon Redshift envia predicados e agregações para a camada de consulta do Redshift Spectrum sempre que possível. Quando grandes quantidades de dados são retornadas do Amazon S3, o processamento é limitado pelos recursos do seu cluster. O Redshift Spectrum é automaticamente dimensionado para processar grandes solicitações. Assim, a performance geral é aprimorado sempre que você pode enviar processamento para a camada do Redshift Spectrum.
- Ao escrever suas consultas, considere o uso de filtros e agregações que sejam elegíveis para o processamento na camada do Redshift Spectrum.

  Os exemplos a seguir mostram algumas operações que podem ser enviadas para a camada do Redshift Spectrum:

  - Cláusulas GROUP BY
  - Condições de comparação e de correspondência de padrões, tais como LIKE.
  - Funções agregadas, tais como COUNT, SUM, AVG, MIN e MAX.
  - Funções de string.

  As operações que não podem ser enviadas para a camada do Redshift Spectrum incluem DISTINCT e ORDER BY.

- Use partições para limitar os dados para a varredura. Divida seus dados em partições com base nos predicados de consulta mais comuns e, em seguida, elimine as partições filtrando pelas colunas de partição. Para obter mais informações, consulte Dividir as tabelas externas do Redshift Spectrum.

  Consulte a SVL_S3PARTITION para visualizar as partições totais e as partições qualificadas.
- Use o gerador de estatísticas do AWS Glue para calcular estatísticas no nível da coluna para tabelas do AWS Glue Data Catalog. Depois que o AWS Glue gera estatísticas para tabelas no Catálogo de Dados, o Amazon Redshift Spectrum as utiliza automaticamente para otimizar o plano de consulta. Para obter mais informações sobre como computar estatísticas no nível da coluna usando AWS Glue, consulte Working with column statistics no Guia do desenvolvedor do AWS Glue.

Opções de tratamento de dados

Este tópico descreve como configurar o Redshift Spectrum para tratar dados em formatos inesperados.

Você pode definir parâmetros de tabela ao criar tabelas externas para personalizar dados sendo consultados em tabelas externas. Do contrário, podem ocorrer erros de verificação. Para obter mais informações, consulte TABLE PROPERTIES (Propriedades de tabela) em CREATE EXTERNAL TABLE. Para obter exemplos, consulte Exemplos de tratamento de dados. Para obter uma lista de erros, consulte SVL_SPECTRUM_SCAN_ERROR.

Você pode definir as seguintes PROPRIEDADES DE TABELA ao criar tabelas externas para especificar o tratamento de entrada para dados que estão sendo consultados em tabelas externas.

- `column_count_mismatch_handling`, para identificar se o arquivo contém um número menor ou maior de valores para uma linha do que o de colunas especificado na definição da tabela externa.
- `invalid_char_handling` para especificar o tratamento de entrada para caracteres inválidos em colunas contendo VARCHAR, CHAR e dados de string. Quando você especifica REPLACE para `invalid_char_handling`, pode especificar o caractere de substituição a ser usado.
- `numeric_overflow_handling` para especificar o tratamento de transbordamento de conversão em colunas contendo dados inteiros e decimais.
- `surplus_bytes_handling` para especificar o tratamento de entradas para bytes excedentes em colunas que contêm dados VARBYTE.

- `surplus_char_handling` para especificar o tratamento de entrada para caracteres excedentes em colunas contendo VARCHAR, CHAR e dados de string.

Você pode definir uma opção de configuração para cancelar consultas que excedam um número máximo de erros. Para obter mais informações, consulte spectrum_query_maxerror.

Exemplo: Executar subconsultas correlacionadas no Redshift Spectrum

Este tópico descreve como executar subconsultas correlacionadas no Redshift Spectrum. Subconsulta correlacionada é uma consulta que usa valores da consulta externa.

Você pode executar subconsultas correlacionadas no Redshift Spectrum. A pseudocoluna `$spectrum_oid` possibilita realizar consultas correlacionadas com o Redshift Spectrum. Para executar uma subconsulta correlacionada, a pseudocoluna `$spectrum_oid` deve estar habilitada, mas não deve aparecer na instrução SQL. Para obter mais informações, consulte Pseudocolunas.

Para criar o esquema externo e as tabelas externas para esse exemplo, consulte Conceitos básicos do Amazon Redshift Spectrum.

Veja a seguir um exemplo de subconsulta correlacionada no Redshift Spectrum.

```
select *
from myspectrum_schema.sales s
where exists
( select *
from myspectrum_schema.listing l
where l.listid = s.listid )
order by salesid
limit 5;
```

```
salesid  listid   sellerid   buyerid   eventid   dateid  qtysold  pricepaid
 commission   saletime
1        1        36861      21191     7872      1875    4        728         109.2
    2008-02-18 02:36:48
2        4        8117       11498     4337      1983    2        76          11.4
    2008-06-06 05:00:16
3        5        1616       17433     8647      1983    2        350         52.5
    2008-06-06 08:26:17
```

```
4       5       1616    19715   8647    1986    1       175     26.25
    2008-06-09 08:38:52
5       6       47402   14115   8240    2069    2       154     23.1
    2008-08-31 09:17:02
```

Métricas no Amazon Redshift Spectrum

Este tópico descreve as visualizações do sistema que podem ser usadas para monitorar consultas de data lake.

Você pode monitorar consultas de data lake usando as seguintes visualizações do sistema:

- SVL_S3QUERY

  Use a exibição SVL_S3QUERY para obter detalhes sobre as consultas de data lake nos níveis de segmento e de fatia do nó.
- SVL_S3QUERY_SUMMARY

  Use a visualização SVL_S3QUERY_SUMMARY para obter um resumo de todas as consultas de data lake que foram executadas no sistema.

Veja a seguir alguns itens a serem observados na exibição SVL_S3QUERY_SUMMARY:

- O número de arquivos que foram processados pela consulta do Redshift Spectrum.
- O número de bytes processados na varredura do Amazon S3. O custo de uma consulta do Redshift Spectrum é refletido na quantidade de dados processados na varredura do Amazon S3.
- O número de bytes retornados da camada do Redshift Spectrum para o cluster. Uma grande quantidade de dados retornados pode afetar a performance do sistema.
- A duração máxima e a duração média das solicitações do Redshift Spectrum. As solicitações de longa duração podem indicar um gargalo.

Observação sobre clusters RG provisionados

Em clusters RG provisionados, as consultas do Redshift Spectrum são executadas na computação do próprio cluster, e não na frota dedicada do Spectrum usada pelos clusters RA3 e DC2 provisionados. SVL_S3QUERY e SVL_S3QUERY_SUMMARYcontinuam sendo preenchidos em clusters RG. As colunas a seguir têm semânticas diferentes ou estão obsoletas.

Colunas preenchidas com semânticas diferentes em clusters RG provisionados

Em clusters RA3 e DC2 provisionados, essas colunas descrevem linhas, bytes e unidades de trabalho que se movem entre o cluster e a frota do Spectrum. Em clusters RG provisionados, elas descrevem o trabalho equivalente realizado pelo leitor nativo do cluster:

- s3_scanned_rows: linhas lidas diretamente do Amazon S3 pelo leitor nativo do cluster (pré-filtro).
- s3_scanned_bytes: tamanho total do intervalo de varredura em bytes processado pelo leitor nativo do cluster.
- s3query_returned_rows: linhas produzidas após o envio do filtro pelo leitor nativo do cluster.
- s3query_returned_bytes: bytes produzidos após o envio do filtro pelo leitor nativo do cluster.
- splits: número de intervalos de varredura consumidos pelo leitor nativo do cluster.
- total_split_size: tamanho total de todos os intervalos de varredura consumidos, em bytes.
- max_split_size: tamanho do maior intervalo de varredura consumido, em bytes.

Colunas obsoletas em clusters RG provisionados

Essas colunas descrevem conceitos de frota do Spectrum que não existem no RG. Em clusters RG, elas são registradas como -1 em STL_S3QUERY e, portanto, aparecem como -1 ou não são significativas em SVL_S3QUERY e SVL_S3QUERY_SUMMARY:

- total_retries, max_retries: no RG, as novas tentativas ocorrem no nível do cliente do Amazon S3. Use STL_S3CLIENT e STL_S3CLIENT_ERROR para obter detalhes sobre novas tentativas.
- max_request_duration, avg_request_duration: o RG não usa o modelo de solicitação do Spectrum.
- max_request_parallelism, avg_request_parallelism: o RG não usa tokens de solicitação do Spectrum.
- slowdown_count, max_concurrent_slowdown_count: no RG, as lentidões do Amazon S3 são monitoradas em STL_S3CLIENT.

Para métricas agregadas por consulta em clusters RG (partições, arquivos digitalizados, linhas e bytes retornados, formato do arquivo, localização do arquivo, listagem e tempo de busca da partição), você também pode usar a visualização de monitoramento SYS_EXTERNAL_QUERY_DETAIL.

Solução de problemas de consulta no Amazon Redshift Spectrum

Este tópico é uma referência para problemas comuns que podem ser encontrados em consultas do Amazon Redshift Spectrum.

Para visualizar os erros gerados pelas consultas do Redshift Spectrum, consulte a tabela do sistema SVL_S3LOG.

Tópicos

- Número de tentativas excedido
- Acesso limitado
- Limite de recursos excedido
- Nenhuma linha foi retornada para uma tabela particionada
- Erro de falta de autorização
- Formatos de dados incompatíveis
- Erro de sintaxe ao usar a DDL do Hive no Amazon Redshift
- Permissão para criar tabelas temporárias
- Intervalo inválido
- Número da versão do Parquet inválida
- Campo obrigatório ausente nas informações da partição externa

Número de tentativas excedido

Se o tempo limite de uma solicitação do Amazon Redshift Spectrum expira, a solicitação é cancelada e reenviada. Depois de cinco novas tentativas com erro, a consulta falha com o seguinte erro.

```
error:  Spectrum Scan Error: Retries exceeded
```

As possíveis causas incluem:

- Arquivos grandes (maiores do que 1 GB). Verifique o tamanho dos arquivos no Amazon S3 e procure por arquivos grandes e distorções no tamanho de um arquivo. Quebre os arquivos grandes em arquivos menores, entre 100 MB e 1 GB. Tente manter os arquivos com aproximadamente o mesmo tamanho.
- A taxa de transferência na rede está lenta. Tente executar sua consulta mais tarde.

Acesso limitado

O Amazon Redshift Spectrum está sujeito às Service Quotas da AWS. Em alta utilização, poderá ser necessário desacelerar as solicitações do Redshift Spectrum, resultando no erro a seguir.

```
error:  Spectrum Scan Error: Access throttled
```

Pode ocorrer dois tipos de controle de utilização:

- Acesso limitado pelo Amazon S3.
- Acesso limitado pelo AWS KMS.

O contexto do erro fornece mais detalhes sobre o tipo de controle de utilização. Veja a seguir causas e possíveis soluções para esse controle de utilização.

Acesso limitado pelo Amazon S3

O Amazon S3 pode limitar uma solicitação do Redshift Spectrum se a taxa de solicitação de leitura em um prefixo é muito alta. Para obter informações sobre uma taxa de solicitação GET/HEAD que você pode obter no Amazon S3, consulte Otimizar a performance do Amazon S3 no Guia do usuário do Amazon Simple Storage Service. A taxa de solicitação GET/HEAD do Amazon S3 considera todas as solicitações GET/HEAD em um prefixo, portanto, aplicações diferentes que acessam o mesmo prefixo compartilham a taxa total de solicitações.

Se as solicitações do Redshift Spectrum forem limitadas pelo Amazon S3 frequentemente, reduza o número de solicitações GET/HEAD do Amazon S3 que o Redshift Spectrum faz para o Amazon S3. Para isso, tente mesclar arquivos pequenos em arquivos maiores. Recomendamos usar tamanhos de arquivo de 64 MB ou maior.

Considere também particionar as tabelas do Redshift Spectrum para se beneficiar da filtragem antecipada e reduzir o número de arquivos acessados no Amazon S3. Para obter mais informações, consulte Dividir as tabelas externas do Redshift Spectrum.

Acesso limitado pelo AWS KMS

Se você armazenar os dados no Amazon S3 usando criptografia no lado do servidor (SSE-S3 ou SSE-KMS), o Amazon S3 chamará uma operação de API ao AWS KMS para cada arquivo acessado pelo Redshift Spectrum. Essas solicitações são contabilizadas na sua cota de operações de criptografia. Para obter mais informações, consulte Cotas de solicitações do AWS KMS. Para

obter mais informações sobre SSE-S3 e SSE-KMS, consulte Proteger dados usando criptografia do lado do servidor e Proteger dados usando criptografia do lado do servidor com chaves KMS armazenadas no AWS KMS no Guia do usuário do Amazon Simple Storage Service.

A primeira etapa para reduzir o número de solicitações feitas pelo Redshift Spectrum ao AWS KMS é reduzir o número de arquivos acessados. Para isso, tente mesclar arquivos pequenos em arquivos maiores. Recomendamos usar tamanhos de arquivo de 64 MB ou maior.

Se as solicitações do Redshift Spectrum são frequentemente limitadas pelo AWS KMS, considere solicitar um aumento de cota para a taxa de solicitações do AWS KMS para operações criptográficas. Para solicitar um aumento de cota, consulte Limites de serviço da AWS na Referência geral da Amazon Web Services.

Limite de recursos excedido

O Redshift Spectrum impõe um limite superior sobre a quantidade de memória que uma solicitação pode usar. Uma solicitação do Redshift Spectrum que requer mais memória falha, resultando no erro a seguir.

```
error:  Spectrum Scan Error: Resource limit exceeded
```

Há dois motivos comuns que podem fazer com que uma solicitação do Redshift Spectrum exceda a provisão de memória:

- O Redshift Spectrum processa um grande bloco de dados que não pode ser dividido em blocos menores.
- Uma grande etapa de agregação é processada pelo Redshift Spectrum.

Recomendamos usar um formato de arquivo que ofereça suporte a leituras paralelas com tamanhos divididos de 128 MB ou menos. Consulte Arquivos de dados para consultas no Amazon Redshift Spectrum para obter os formatos de arquivo compatíveis e as diretrizes gerais para a criação de arquivos de dados. Ao usar formatos de arquivo ou algoritmos de compactação que não ofereçam suporte a leituras paralelas, recomendamos manter os tamanhos de arquivo entre 64 MB e 128 MB.

Nenhuma linha foi retornada para uma tabela particionada

Se a consulta não retornar nenhuma linha de uma tabela externa particionada, verifique se foi adicionada uma partição para essa tabela externa. O Redshift Spectrum só faz a varredura dos arquivos em um local do Amazon S3 que tenha sido explicitamente adicionado usando `ALTER`

`TABLE … ADD PARTITION`. Consulte a exibição SVV_EXTERNAL_PARTITIONS para encontrar partições existentes. Execute `ALTER TABLE … ADD PARTITION` para cada partição ausente.

Erro de falta de autorização

Verifique se a função do IAM para o cluster permite o acesso aos objetos de arquivo do Amazon S3. Se o banco de dados externo estiver localizado no Amazon Athena, verifique se a função do IAM permite o acesso aos recursos do Athena. Para obter mais informações, consulte Políticas do IAM do Amazon Redshift Spectrum.

Formatos de dados incompatíveis

Para um formato de arquivo colunar, como o Parquet, o tipo de coluna é incorporado nos dados. O tipo de coluna da definição CREATE EXTERNAL TABLE deve corresponder ao tipo de coluna do arquivo de dados. Se não houver correspondência, você receberá um erro semelhante ao seguinte:

```
File 'https://s3bucket/location/file has an incompatible Parquet schema
for column ‘s3://s3bucket/location.col1'. Column type: VARCHAR, Par
```

A mensagem de erro pode estar truncada devido ao limite de tamanho de mensagens. Para recuperar a mensagem de erro completa, incluindo o nome e o tipo da coluna, consulte de exibição do sistema SVL_S3LOG.

O exemplo a seguir consulta a SVL_S3LOG para a última consulta concluída.

```
select message
from svl_s3log
where query = pg_last_query_id()
order by query,segment,slice;
```

O exemplo a seguir mostra uma mensagem de erro completa.

```
                            message
-----------------------------------------------------------------------------------------
Spectrum Scan Error. File 'https://s3bucket/location/file has an incompatible
Parquet schema for column ' s3bucket/location.col1'.
Column type: VARCHAR, Parquet schema:\noptional int64 l_orderkey [i:0 d:1 r:0]\n
```

Para corrigir o erro, modifique a tabela externa para usar o mesmo tipo de coluna do arquivo Parquet.

Erro de sintaxe ao usar a DDL do Hive no Amazon Redshift

O Amazon Redshift é compatível com linguagem de definição de dados (DDL) para CREATE EXTERNAL TABLE, que é semelhante à DDL do Hive. Contudo, os dois tipos de DDL não são sempre exatamente iguais. Se você copiar a DDL do Hive para criar ou alterar as tabelas externas do Amazon Redshift, poderá encontrar erros de sintaxe. Os seguintes exemplos apresentam as diferenças entre as DDLs do Amazon Redshift e do Hive:

- O Amazon Redshift exige aspas simples ('), enquanto a DDL do Hive é compatível com aspas duplas (").
- O Amazon Redshift não é compatível com o tipo de dado STRING. Use o tipo VARCHAR em seu lugar.

Permissão para criar tabelas temporárias

Para executar as consultas do Redshift Spectrum, o usuário do banco de dados precisa ter permissão para criar tabelas temporárias no banco de dados. O exemplo a seguir concede permissão temporária no banco de dados `spectrumdb` para o grupo de usuários `spectrumusers`.

```
grant temp on database spectrumdb  to group spectrumusers;
```

Para obter mais informações, consulte GRANT.

Intervalo inválido

O Redshift Spectrum espera que os arquivos no Amazon S3 pertencentes a uma tabela externa não sejam substituídos durante uma consulta. Se isso acontecer, o seguinte erro poderá ocorrer.

```
Error: HTTP response error code: 416 Message: InvalidRange The requested range is not
 satisfiable
```

Para evitar o erro, certifique-se de que os arquivos do Amazon S3 não sejam substituídos enquanto eles são consultados com o Redshift Spectrum.

Número da versão do Parquet inválida

O Redshift Spectrum verifica os metadados de cada arquivo do Apache Parquet que ele acessa. Se a verificação falhar, isso pode resultar em um erro similar ao seguinte:

```
File 'https://s3.region.amazonaws.com/s3bucket/location/file has an invalid version
 number
```

Há dois motivos comuns que podem fazer com que a verificação falhe:

- O arquivo Parquet foi substituído durante a consulta (consulte Intervalo inválido).
- O arquivo Parquet está corrompido.

Campo obrigatório ausente nas informações da partição externa

Quando você tenta adicionar uma partição a uma tabela externa em um catálogo externo, você pode receber o seguinte erro:

```
Error: The required field (<field_name>) is missing from the external partition
 information. Add missing field in partition and retry. Partition location:
 <partition_path>
```

Esse erro significa que uma das partições na tabela externa usada em sua consulta tem informações de metadados de partição ausentes. Isso ocorre nos casos a seguir:

- Você adicionou uma partição a uma tabela externa em um catálogo externo, como o AWS Glue Data Catalog, com informações parciais.
- Você consultou uma tabela particionada no Amazon Redshift enquanto adicionava ou atualizava partições para a tabela correspondente em um catálogo externo, como o AWS Glue Data Catalog.

A seguir estão os campos que devem ser preenchidos ao recuperar uma partição do AWS Glue Data Catalog:

- StorageDescriptor
  - InputFormat
  - OutputFormat
  - SerDeInfo
- Valores

Você pode consultar SVV_EXTERNAL_PARTITIONS para encontrar partições existentes e ver detalhes em seus campos.

Para obter uma lista completa dos campos de partição AWS Glue Data Catalog, consulte Partição na Referência da API Web do AWS Glue.

Tutorial: Consultar dados aninhados com o Amazon Redshift Spectrum

Este tutorial demonstra como consultar dados aninhados com o Redshift Spectrum. Dados aninhados são dados que contêm campos aninhados. Campos aninhados são campos unidos como uma única entidade (por exemple, matrizes, structs ou objetos).

Tópicos

- Visão geral
- Etapa 1: Crie uma tabela externa que contém dados aninhados
- Etapa 2: Consultar os dados aninhados no Amazon S3 com extensões SQL
- Casos de uso de dados aninhados
- Limitações de dados aninhados (visualização)
- Serializar JSON aninhado complexo

Visão geral

O Amazon Redshift Spectrum oferece suporte a consulta de dados aninhados em formatos de arquivo Parquet, ORC, JSON e Ion. O Redshift Spectrum acessa dados usando tabelas externas. Você pode criar tabelas externas usando os tipos de dados complexos `struct`, `array` e `map`.

Por exemplo, suponha que o arquivo de dados contém os seguintes dados no Amazon S3 em uma pasta nomeada `customers`. Embora não haja um elemento raiz único, cada objeto JSON nestes dados de exemplo representa uma linha em uma tabela.

```
{"id": 1,
 "name": {"given": "John", "family": "Smith"},
 "phones": ["123-457789"],
 "orders": [{"shipdate": "2018-03-01T11:59:59.000Z", "price": 100.50},
            {"shipdate": "2018-03-01T09:10:00.000Z", "price": 99.12}]
}
{"id": 2,
 "name": {"given": "Jenny", "family": "Doe"},
 "phones": ["858-8675309", "415-9876543"],
```

```
 "orders": []
}
{"id": 3,
 "name": {"given": "Andy", "family": "Jones"},
 "phones": [],
 "orders": [{"shipdate": "2018-03-02T08:02:15.000Z", "price": 13.50}]
}
```

Você pode usar o Amazon Redshift Spectrum para consultar dados aninhados em arquivos. O tutorial a seguir mostra como fazer isso com dados do Apache Parquet.

Pré-requisitos

Se você ainda não estiver usando o Redshift Spectrum, acompanhe as etapas no Conceitos básicos do Amazon Redshift Spectrum antes de continuar.

Para criar um esquema externo, substitua o ARN do perfil do IAM no comando a seguir pelo ARN do perfil que você criou em Criar um perfil do IAM. Execute o comando no cliente SQL.

```
create external schema spectrum
from data catalog
database 'myspectrum_db'
iam_role 'arn:aws:iam::123456789012:role/myspectrum_role'
create external database if not exists;
```

Etapa 1: Crie uma tabela externa que contém dados aninhados

Você pode visualizar a fonte de dados baixando-a do Amazon S3.

Para criar a tabela externa para este tutorial, execute o seguinte comando.

```
CREATE EXTERNAL TABLE spectrum.customers (
  id     int,
  name   struct<given:varchar(20), family:varchar(20)>,
  phones array<varchar(20)>,
  orders array<struct<shipdate:timestamp, price:double precision>>
)
STORED AS PARQUET
LOCATION 's3://redshift-downloads/tickit/spectrum/customers/';
```

No exemplo que precede, a tabela externa `spectrum.customers` usa os tipos de dados `struct` e `array` para definir colunas com dados aninhados. O Amazon Redshift Spectrum oferece suporte

a consulta de dados aninhados em formatos de arquivo Parquet, ORC, JSON e Ion. O parâmetro `STORED AS` é `PARQUET` para arquivos do Apache Parquet. O parâmetro `LOCATION` deve se referir à pasta do Amazon S3 que contém os dados ou arquivos aninhados. Para obter mais informações, consulte CREATE EXTERNAL TABLE.

Você pode aninhar os tipos `array` e `struct` em qualquer nível. Por exemplo, você pode definir uma coluna nomeada `toparray` conforme exibido no seguinte exemplo.

```
toparray array<struct<nestedarray:
         array<struct<morenestedarray:
         array<string>>>>>
```

Você também pode aninhar os tipos `struct` como exibido na coluna `x` no exemplo a seguir.

```
x struct<a: string,
         b: struct<c: integer,
                   d: struct<e: string>
                  >
        >
```

Etapa 2: Consultar os dados aninhados no Amazon S3 com extensões SQL

O Redshift Spectrum oferece suporte a tipos complexos de `array`, `map` e `struct` por meio das extensões de sintaxe SQL do Amazon Redshift SQL.

Extensão 1: Acesso a colunas de estruturas

Você pode extrair dados das colunas `struct` usando uma notação de ponto que concatene nomes de campo em caminhos. Por exemplo, a consulta a seguir retorna nomes e nomes de família de clientes. O nome é acessado pelo caminho longo `c.name.given`. O nome de família é acessado pelo caminho longo `c.name.family`.

```
SELECT c.id, c.name.given, c.name.family
FROM   spectrum.customers c;
```

A consulta anterior retorna os dados a seguir.

```
id | given | family
---|-------|-------
1  | John  | Smith
2  | Jenny | Doe
```

```
3  | Andy  | Jones
(3 rows)
```

Uma `struct` pode ser uma coluna de outro `struct`, que pode ser uma coluna de outro `struct`, em qualquer nível. Os caminhos que acessa as colunas em `struct` aninhados de forma profunda podem ser arbitrariamente longos. Por exemplo, consulte a definição da coluna `x` no seguinte exemplo.

```
x struct<a: string,
         b: struct<c: integer,
                   d: struct<e: string>
                  >
        >
```

Você pode acessar os dados em `e` como `x.b.d.e`.

Extensão 2: Variação sobre matriz em uma cláusula FROM

Você pode extrair dados das colunas `array` (e, por extensão, colunas `map`) especificando as colunas `array` em uma cláusula `FROM` no lugar de nomes de tabela. A extensão aplica-se à cláusula `FROM` da consulta principal, e também as cláusulas `FROM` das subconsultas.

Você pode referenciar elementos `array` por posição, como `c.orders[0]`. (visualização)

Ao combinar a variação do `arrays` com junções, você pode alcançar vários tipos de não-aninhamento, como explicado os seguintes casos de uso.

Não aninhamento usando junções internas

A consulta a seguir selecione IDs de cliente e datas de envio de pedido de clientes que têm pedidos. A extensão do SQL na cláusula FROM `c.orders o` depende do alias `c`.

```
SELECT c.id, o.shipdate
FROM   spectrum.customers c, c.orders o
```

Para cada cliente `c` que possui pedidos, a cláusula `FROM` retorna uma linha para cada pedido `o` do cliente `c`. Essa linha combina a linha do cliente `c` e a linha de pedidos `o`. Em seguida, a cláusula `SELECT` mantém somente `c.id` e `o.shipdate`. O resultado é o seguinte.

```
id|      shipdate
--|----------------------
```

```
1 |2018-03-01  11:59:59
1 |2018-03-01  09:10:00
3 |2018-03-02  08:02:15
(3 rows)
```

O alias `c` fornece acesso aos campos de clientes, e o alias o fornece acesso aos campos de pedidos.

A semântica é semelhante ao SQL padrão. Você pode pensar na cláusula `FROM` como executando o loop aninhado a seguir, que é acompanhado por `SELECT` escolhendo os campos para saída.

```
for each customer c in spectrum.customers
  for each order o in c.orders
     output c.id and o.shipdate
```

Portanto, se um cliente não tiver um pedido, o cliente não aparece no resultado.

Você também pode considerá-lo como cláusula `FROM` da execução `JOIN` com a tabela `customers` e o array `orders`. Na verdade, você também pode escrever a consulta, conforme mostrado no exemplo a seguir.

```
SELECT c.id, o.shipdate
FROM   spectrum.customers c INNER JOIN c.orders o ON true
```

> **Note**
>
> Se um esquema nomeado `c` existe com uma tabela nomeada `orders`, então `c.orders` refere-se a tabela `orders`, e não a coluna de matriz de `customers`.

Não aninhamento usando junções à esquerda

A consulta a seguir exibe todos os nomes de clientes e seus pedidos. Se um cliente não fez um pedido, o nome do cliente ainda é retornado. Contudo, nesse caso, as colunas de pedido são NULL, conforme mostrado no exemplo a seguir para Jenny Doe.

```
SELECT c.id, c.name.given, c.name.family, o.shipdate, o.price
FROM   spectrum.customers c LEFT JOIN c.orders o ON true
```

A consulta anterior retorna os dados a seguir.

```
id  |  given  | family  |    shipdate          | price
----|---------|---------|----------------------|--------
 1  |  John   | Smith   | 2018-03-01  11:59:59 | 100.5
 1  |  John   | Smith   | 2018-03-01  09:10:00 |  99.12
 2  |  Jenny  | Doe     |                      |
 3  |  Andy   | Jones   | 2018-03-02  08:02:15 |  13.5
 (4 rows)
```

Extensão 3: Acessar uma matriz de escalares usando diretamente um alias

Quando um alias p em intervalos de uma cláusula FROM percorre uma matriz de escalares, a consulta se refere aos valores de p como p. Por exemplo, a consulta a seguir produz pares de nomes de clientes e de números de telefone.

```
SELECT c.name.given, c.name.family, p AS phone
FROM   spectrum.customers c LEFT JOIN c.phones p ON true
```

A consulta anterior retorna os dados a seguir.

```
given  |  family  |  phone
-------|----------|-----------
John   | Smith    | 123-4577891
Jenny  | Doe      | 858-8675309
Jenny  | Doe      | 415-9876543
Andy   | Jones    |
(4 rows)
```

Extensão 4: Acessar elementos de mapas

Redshift Spectrum trata o tipo de dados `map` como um tipo `array` que contém tipos `struct` com uma coluna `key` e uma coluna `value`. O `key` deve ser um `scalar`; o valor pode ser qualquer tipo de dados.

Por exemplo, o seguinte código criar uma tabela externa com um `map` para armazenar números de telefone.

```
CREATE EXTERNAL TABLE spectrum.customers2 (
  id     int,
  name   struct<given:varchar(20), family:varchar(20)>,
  phones map<varchar(20), varchar(20)>,
```

```
  orders array<struct<shipdate:timestamp, price:double precision>>
)
STORED AS PARQUET
LOCATION 's3://redshift-downloads/tickit/spectrum/customers/';
```

Como um tipo `map` se comporta como um tipo `array` com colunas `key` e `value`, você pode entender os esquemas anteriores como os seguintes.

```
CREATE EXTERNAL TABLE spectrum.customers3 (
  id     int,
  name   struct<given:varchar(20), family:varchar(20)>,
  phones array<struct<key:varchar(20), value:varchar(20)>>,
  orders array<struct<shipdate:timestamp, price:double precision>>
)
STORED AS PARQUET
LOCATION 's3://redshift-downloads/tickit/spectrum/customers/';
```

A consulta a seguir retorna os nomes de clientes com um número de celular e retorna a quantidade para cada nome. A consulta de mapeamento é tratada como o equivalente de consulta dos tipos `array` de `struct` aninhados. A consulta a seguir retorna somente dados se você tiver criado a tabela externa como descrito previamente.

```
SELECT c.name.given, c.name.family, p.value
FROM   spectrum.customers c, c.phones p
WHERE  p.key = 'mobile';
```

> **Note**
>
> O `key` de um `map` é um `string` para tipos de arquivo Ion e JSON.

Casos de uso de dados aninhados

Este tópico descreve casos de uso para dados aninhados. Dados aninhados são dados que contêm campos aninhados. Campos aninhados são campos unidos como uma única entidade (por exemple, matrizes, structs ou objetos).

Você pode combinar as extensões descritas anteriormente com as características usuais SQL. Os casos de uso a seguir ilustram algumas combinações comuns. Estes exemplos ajudam a demonstrar como você pode usar dados aninhados. Eles não fazem parte do tutorial.

Tópicos

- Ingerir dados aninhados
- Agregar dados aninhados com subconsultas
- Unir o Amazon Redshift e dados aninhados

Ingerir dados aninhados

Você pode usar uma declaração `CREATE TABLE AS` para ingerir dados de uma tabela externa que contenha tipos de dados complexos. A consulta a seguir extrai todos os clientes e seus números de telefone da tabela externa, usando `LEFT JOIN` e os armazena na tabela `CustomerPhones`. do Amazon Redshift.

```
CREATE TABLE CustomerPhones AS
SELECT  c.name.given, c.name.family, p AS phone
FROM    spectrum.customers c LEFT JOIN c.phones p ON true;
```

Agregar dados aninhados com subconsultas

Você pode usar uma subconsulta para agregar dados aninhados. O exemplo a seguir ilustra essa abordagem.

```
SELECT c.name.given, c.name.family, (SELECT COUNT(*) FROM c.orders o) AS ordercount
FROM   spectrum.customers c;
```

Os dados a seguir são retornados.

```
given   |  family  |  ordercount
--------|----------|-------------
 Jenny  |  Doe     |       0
 John   |  Smith   |       2
 Andy   |  Jones   |       1
 (3 rows)
```

> **Note**
>
> Quando você agrega dados aninhados agrupando os pela linha pai, a forma mais eficiente é a exibida no exemplo anterior. Nesse caso, as linhas aninhadas de `c.orders` são

agrupadas em sua linha pai `c`. Ou, se você sabe que `id` é exclusivo para cada `customer` e o `.shipdate` e nunca nulo, você pode agregar conforme exibido no seguinte exemplo. Contudo, esta abordagem não é geralmente mais eficiente que o exemplo anterior.

```
SELECT    c.name.given, c.name.family, COUNT(o.shipdate) AS ordercount
FROM      spectrum.customers c LEFT JOIN c.orders o ON true
GROUP BY  c.id, c.name.given, c.name.family;
```

Você também pode gravar a consulta usando uma subconsulta na cláusula `FROM` que refere-se a um alias (`c`) da consulta antecessora e extrai dados da matriz. O exemplo a seguir demonstra essa abordagem.

```
SELECT c.name.given, c.name.family, s.count AS ordercount
FROM   spectrum.customers c, (SELECT count(*) AS count FROM c.orders o) s;
```

Unir o Amazon Redshift e dados aninhados

Você também pode adicionar dados do Amazon Redshift com dados aninhados em uma tabela externa. Por exemplo, digamos que você tenha os dados a seguir aninhados no Amazon S3.

```
CREATE EXTERNAL TABLE spectrum.customers2 (
  id      int,
  name    struct<given:varchar(20), family:varchar(20)>,
  phones  array<varchar(20)>,
  orders  array<struct<shipdate:timestamp, item:int>>
);
```

Digamos também que possui a tabela a seguir no Amazon Redshift.

```
CREATE TABLE prices (
  id int,
  price double precision
);
```

A consulta a seguir encontra o número total e a quantidade de compras de cada cliente com base em precedência. O exemplo a seguir é apenas uma ilustração. Só retorna dados se você tiver criado as tabelas descritas anteriormente.

```
SELECT   c.name.given, c.name.family, COUNT(o.date) AS ordercount, SUM(p.price) AS
 ordersum
FROM     spectrum.customers2 c, c.orders o, prices p ON o.item = p.id
GROUP BY c.id, c.name.given, c.name.family;
```

Limitações de dados aninhados (visualização)

Este tópico descreve as limitações referentes à leitura de dados aninhados com o Redshift Spectrum. Dados aninhados são dados que contêm campos aninhados. Campos aninhados são campos unidos como uma única entidade (por exemple, matrizes, structs ou objetos).

> **Note**
>
> As limitações marcadas (visualização prévia) na lista a seguir só se aplicam a clusters de visualização prévia criados nas regiões a seguir.
>
> - Leste dos EUA (Ohio) (us-east-2)
> - Leste dos EUA (Norte da Virgínia) (us-east-1)
> - Oeste dos EUA (Norte da Califórnia) (us-west-1)
> - Ásia Pacific (Tóquio) (ap-northeast-1)
> - Europa (Irlanda) (eu-west-1)
> - UE (Estocolmo) (eu-north-1)
>
> Para obter informações sobre como configurar clusters de visualização, consulte Creating a preview cluster no Guia de gerenciamento do Amazon Redshift.

As limitações a seguir se aplicam à dados aninhados:

- Um tipo `map` ou `array` pode conter outros tipos `map` ou `array`, desde que as consultas em `arrays` ou `maps` aninhados não retornem valores `scalar`. (visualização)
- O Amazon Redshift Spectrum só dá suporte a tipos de dados complexos como tabelas externas.
- As colunas de resultado da subconsulta devem estar no nível superior. (visualização)
- Se uma expressão `OUTER JOIN` refere-se a uma tabela aninhada, pode consultar somente a tabela e seus arrays aninhados (e mapas). Se uma expressão `OUTER JOIN` não refere-se a uma tabela aninhada, pode consultar qualquer número de tabelas não aninhadas.

- Se uma cláusula FROM em uma subconsulta refere-se a uma tabela aninhada, não pode consultar de outra tabela.
- Se uma subconsulta depende de uma tabela aninhada referente a uma tabela pai, você só pode usar a tabela pai na cláusula FROM. Você não pode usar o pai em nenhuma outra cláusula, como a cláusula SELECT ou WHERE. Por exemplo, a consulta a seguir não é executada porque a cláusula SELECT da subconsulta se refere à tabela pai c.

```
SELECT c.name.given
FROM   spectrum.customers c
WHERE (SELECT COUNT(c.id) FROM c.phones p WHERE p LIKE '858%') > 1;
```

A consulta a seguir funciona porque o pai c é usado apenas na cláusula FROM de uma subconsulta.

```
SELECT c.name.given
FROM   spectrum.customers c
WHERE (SELECT COUNT(*) FROM c.phones p WHERE p LIKE '858%') > 1;
```

- Uma subconsulta que acessa dados aninhados em qualquer lugar, diferente da cláusula FROM deve retornar um valor único. As únicas exceções são os operadores (NOT) EXISTS em uma cláusula WHERE.
- (NOT) INNão há suporte ao .
- A profundidade máxima de assentamento para todos os tipos aninhados é de 100. Essa restrição aplica-se a todos os formatos de arquivo (Parquet, ORC, Ion e JSON).
- Subconsultas de agregação que acessam dados aninhadas podem se referir somente a arrays e maps em sua cláusula FROM, e não a uma tabela externa.
- Não há suporte para consultar as pseudocolunas de dados aninhados em uma tabela do Redshift Spectrum. Para obter mais informações, consulte Pseudocolunas.
- Ao extrair dados das colunas de matriz ou mapa especificando-os em uma cláusula FROM, você só poderá selecionar valores dessas colunas se os valores forem scalar. Por exemplo, ambas as consultas a seguir tentam elementos SELECT por dentro de uma matriz. A consulta que seleciona arr.a funciona porque arr.a é um valor scalar. A segunda consulta não funciona porque array é uma matriz extraída de s3.nested table na cláusula FROM. (visualização)

```
SELECT array_column FROM s3.nested_table;

array_column
```

```
-----------------
[{"a":1},{"b":2}]

SELECT arr.a FROM s3.nested_table t, t.array_column arr;

arr.a
-----
1

--This query fails to run.
SELECT array FROM s3.nested_table tab, tab.array_column array;
```

Você não pode usar uma matriz ou um mapa na cláusula `FROM` que acompanha outra matriz ou mapa. Para selecionar matrizes ou outras estruturas complexas que estejam aninhadas dentro de outras matrizes, considere usar índices na instrução `SELECT`.

Serializar JSON aninhado complexo

Este tópico demonstra como serializar dados aninhados no formato JSON. Dados aninhados são dados que contêm campos aninhados. Campos aninhados são campos unidos como uma única entidade (por exemple, matrizes, structs ou objetos).

Uma alternativa aos métodos demonstrados neste tutorial é consultar colunas de coleção aninhadas de nível superior como JSON serializado. Você pode usar a serialização para inspecionar, converter e ingerir dados aninhados como JSON com Redshift Spectrum. Este método é compatível com os formatos ORC, JSON, Ion e Parquet. Usar o parâmetro de configuração da sessão `json_serialization_enable` para configurar o comportamento de serialização. Quando definido, tipos de dados JSON complexos são serializados para VARCHAR (65535). O JSON aninhado pode ser acessado com Funções JSON. Para obter mais informações, consulte json_serialization_enable.

Por exemplo, sem definir `json_serialization_enable`, as consultas a seguir que acessam colunas aninhadas falham diretamente.

```
SELECT * FROM spectrum.customers LIMIT 1;

=> ERROR:  Nested tables do not support '*' in the SELECT clause.

SELECT name FROM spectrum.customers LIMIT 1;
```

```
=> ERROR:  column "name" does not exist in customers
```

Configurar `json_serialization_enable` permite consultar coleções de nível superior diretamente.

```
SET json_serialization_enable TO true;

SELECT * FROM spectrum.customers order by id LIMIT 1;

id | name                                  | phones         | orders
---+---------------------------------------+----------------
+-----------------------------------------------------------------------------------------------------------------
1  | {"given": "John", "family": "Smith"} | ["123-457789"] | [{"shipdate":
 "2018-03-01T11:59:59.000Z", "price": 100.50}, {"shipdate": "2018-03-01T09:10:00.000Z",
 "price": 99.12}]

SELECT name FROM spectrum.customers order by id LIMIT 1;

name
---------
{"given": "John", "family": "Smith"}
```

Considere os seguintes itens ao serializar JSON aninhado.

- Quando colunas de coleção são serializadas como VARCHAR (65535), seus subcampos aninhados não podem ser acessados diretamente como parte da sintaxe de consulta (por exemplo, na cláusula de filtro). No entanto, as funções JSON podem ser usadas para acessar JSON aninhado.
- As seguintes representações especializadas não são compatíveis:
  - Junções ORC
  - Mapas ORC com chaves de tipo complexas
  - Datagramas Ion
  - SEXP Ion
- Os timestamps são retornados como strings serializadas ISO.
- Chaves de mapa primitivas são promovidas para string (por exemplo, 1 para `"1"`).
- Valores nulos de nível superior são serializados como NULLs.
- Se a serialização ultrapassa o tamanho máximo VARCHAR de 65535, a célula é definida como NULL.

Serializar tipos complexos contendo strings JSON

Por padrão, os valores de string contidos em coleções aninhadas são serializados como strings JSON escapadas. Escapar pode ser indesejável quando as strings são JSON válidos. Em vez disso, você pode querer gravar subelementos aninhados ou campos que são VARCHAR diretamente como JSON. Habilite esse comportamento com a configuração no nível da sessão `json_serialization_parse_nested_strings`. Quando ambos `json_serialization_enable` e `json_serialization_parse_nested_strings` são definidos, valores JSON válidos são serializados inline sem caracteres de escape. Quando o valor não é um JSON válido, ele é escapado como se a propriedade `json_serialization_parse_nested_strings` não foi definido. Para obter mais informações, consulte json_serialization_parse_nested_strings.

Por exemplo, suponha que os dados do exemplo anterior continha JSON como um tipo complexo `structs` no campo `name`VARCHAR(20):

```
name
---------
{"given": "{\"first\":\"John\",\"middle\":\"James\"}", "family": "Smith"}
```

Quando `json_serialization_parse_nested_strings` estiver definido, a propriedade `name` é serializada da seguinte forma:

```
SET json_serialization_enable TO true;
SET json_serialization_parse_nested_strings TO true;
SELECT name FROM spectrum.customers order by id LIMIT 1;

name
---------
{"given": {"first":"John","middle":"James"}, "family": "Smith"}
```

Em vez de ser escapado assim:

```
SET json_serialization_enable TO true;
SELECT name FROM spectrum.customers order by id LIMIT 1;

name
---------
{"given": "{\"first\":\"John\",\"middle\":\"James\"}", "family": "Smith"}
```

Esboços do HyperLogLog

Este tópico descreve como utilizar esboços do HyperLogLog no Amazon Redshift. O HyperLogLog é um algoritmo destinado ao problema de contagem distinta que aproxima o número de elementos distintos em um conjunto de dados. Os esboços do HyperLogLog são matrizes de dados exclusivos sobre um conjunto de dados.

HyperLogLog é um algoritmo usado para estimar a cardinalidade de um multiconjunto. Cardinalidade refere-se ao número de valores distintos em um multiconjunto. Por exemplo, no conjunto de {4,3,6,2,2,6,4,3,6,2,2,3}, a cardinalidade é 4 com valores distintos de 4, 3, 6 e 2.

A precisão do algoritmo HyperLogLog (também conhecido como valor m) pode afetar a precisão da cardinalidade estimada. Durante a estimativa de cardinalidade, o Amazon Redshift usa um valor de precisão padrão de 15. Esse valor pode ser de até 26 para conjuntos de dados menores. Assim, o erro relativo médio varia entre 0,01 a 0,6%.

Ao calcular a cardinalidade de um multiconjunto, o algoritmo HyperLogLog gera uma construção chamada esboço HLL. Um esboço HLL encapsula informações sobre os valores distintos em um multiconjunto. O tipo de dados HLLSKETCH do Amazon Redshift representa esses valores de esboço. Esse tipo de dados pode ser usado para armazenar esboços em uma tabela do Amazon Redshift. Além disso, o Amazon Redshift oferece suporte a operações que podem ser aplicadas a valores HLLSKETCH como funções agregadas e escalares. Você pode usar essas funções para extrair a cardinalidade de um HLLSKETCH e combinar vários valores HLLSKETCH.

O tipo de dados HLLSKETCH oferece benefícios significativos de performance de consulta ao extrair a cardinalidade de grandes conjuntos de dados. Você pode pré-agregar esses conjuntos de dados usando valores HLLSKETCH e armazená-los em tabelas. O Amazon Redshift pode extrair a cardinalidade diretamente dos valores HLLSKETCH armazenados sem acessar os conjuntos de dados subjacentes.

Ao processar esboços HLL, o Amazon Redshift executa otimizações que minimizam o espaço de memória do esboço e maximizam a precisão da cardinalidade extraída. O Amazon Redshift usa duas representações para esboços HLL, esparsos e densos. Um HLLSKETCH é iniciado em formato esparso. À medida que novos valores são inseridos nele, seu tamanho aumenta. Depois que seu tamanho atingir o tamanho da representação densa, o Amazon Redshift converte automaticamente o esboço de esparso para denso.

O Amazon Redshift importa, exporta e imprime um HLLSKETCH como JSON quando o esboço está em um formato esparso. O Amazon Redshift importa, exporta e imprime um HLLSKETCH como

uma string Base64 quando o esboço está em um formato denso. Para obter mais informações sobre UNLOAD, consulte Descarregar o tipo de dados HLLSKETCH. Para importar dados de valores separados por vírgula (CSV) para o Amazon Redshift, use o comando COPY. Para obter mais informações, consulte Carregando o tipo de dados HLLSKETCH.

Para obter informações sobre funções usadas com HyperLogLog, consulte Funções HyperLogLog.

Tópicos

- Considerações
- Limitações
- Exemplos

Considerações

Este tópico descreve os detalhes de uso do HyperLogLog no Amazon Redshift.

Veja as seguintes considerações sobre o uso do HyperLogLog no Amazon Redshift:

- As seguintes funções não Hyperloglog podem aceitar uma entrada do tipo HLLSKETCH ou colunas do tipo HLLSKETCH:
  - A função agregada COUNT
  - As expressões condicionais COALESCE e NVL
  - Expressões de CASOS
- A codificação compatível é RAW.
- Você pode executar uma operação UNLOAD na tabela do com colunas HLLSKETCH em texto ou CSV. Você pode usar as colunas UNLOAD HLLSKETCH para gravar dados HLLSKETCH. O Amazon Redshift mostra os dados em um formato JSON para uma representação esparsa ou um formato Base64 para uma representação densa. Para obter mais informações sobre UNLOAD, consulte Descarregar o tipo de dados HLLSKETCH.

  A seguir está o formato usado para um esboço do HyperLogLog esparso representado em um formato JSON.

```
{"version":1,"logm":15,"sparse":{"indices":
[15099259,33107846,37891580,50065963],"values":[2,3,2,1]}}
```

- Você pode importar texto ou dados CSV para o Amazon Redshift usando o comando COPY. Para obter mais informações, consulte Carregando o tipo de dados HLLSKETCH.
- A codificação padrão para HLLSKETCH é RAW. Para obter mais informações, consulte Codificações de compactação.

Limitações

Este tópico descreve as limitações do HyperLogLog no Amazon Redshift.

Veja as seguintes limitações para usar o HyperLogLog no Amazon Redshift:

- As tabelas do Amazon Redshift não são compatíveis com uma coluna HLLSKETCH como uma chave de classificação ou como uma chave de distribuição para uma tabela do Amazon Redshift.
- O Amazon Redshift não é compatível com colunas HLLSKETCH em cláusulas ORDER BY, GROUP BY ou DISTINCT.
- Você só pode UNLOAD colunas HLLSKETCH para o formato de texto ou CSV. Em seguida, o Amazon Redshift grava os dados HLLSKETCH em um formato JSON ou Base64. Para obter mais informações sobre UNLOAD, consulte UNLOAD.
- O Amazon Redshift suporta apenas esboços HyperLogLog com uma precisão (valor de logm) de 15.
- Drivers JDBC e ODBC não suportam o tipo de dados HLLSKETCH. Portanto, o conjunto de resultados usa VARCHAR para representar os valores HLLSKETCH.
- O Amazon Redshift Spectrum não oferece suporte nativo aos dados HLLSKETCH. Portanto, não é possível criar ou alterar uma tabela externa com uma coluna de HLLSKETCH.
- Os tipos de dados para funções definidas pelo usuário (UDFs) do Python não são compatíveis com o tipo de dados HLLSKETCH. Para obter mais informações sobre UDFs do Python, consulte UDFs escalares do Python.

> **Note**
>
> A partir de 1.º de novembro de 2025, o Amazon Redshift não permitirá mais a criação de UDFs do Python. As UDFs do Python existentes continuarão a funcionar normalmente até 30 de junho de 2026. A partir de 1.º de julho de 2026, o Amazon Redshift não permitirá mais a criação de UDFs do Python. Recomendamos enfaticamente que você migre as UDFs existentes do Python para as UDFs do Lambda antes de 1.º de novembro de 2025. Para ter

mais informações sobre como criar e usar UDFs do Lambda, consulte UDFs escalares do Lambda. Para ter informações sobre a conversão de UDFs do Python existentes em UDFs do Lambda, consulte a publicação de blog .

Exemplos

Esta seção contém exemplos de uso do HyperLogLog com o Amazon Redshift.

Tópicos

- Exemplo: retornar cardinalidade em uma subconsulta
- Exemplo: retorna um tipo HLLSKETCH de esboços combinados em uma subconsulta
- Exemplo: retorna um esboço do HyperLogLog da combinação de vários esboços
- Exemplo: gerar esboços do HyperLogLog sobre dados do S3 usando tabelas externas

Exemplo: retornar cardinalidade em uma subconsulta

O exemplo a seguir retorna a cardinalidade para cada esboço em uma subconsulta para uma tabela chamada Vendas.

```
CREATE TABLE Sales (customer VARCHAR, country VARCHAR, amount BIGINT);
INSERT INTO Sales VALUES ('David Joe', 'Greece', 14.5),  ('David Joe', 'Greece',
 19.95), ('John Doe', 'USA', 29.95), ('John Doe', 'USA', 19.95), ('George Spanos',
 'Greece', 9.95), ('George Spanos', 'Greece', 2.95);
```

A consulta a seguir gera um esboço HLL para os clientes de cada país e extrai a cardinalidade. Isso mostra clientes únicos de cada país.

```
SELECT hll_cardinality(sketch), country
FROM (SELECT hll_create_sketch(customer) AS sketch, country
        FROM Sales
        GROUP BY country) AS hll_subquery;

hll_cardinality | country
----------------+---------
            1   | USA
            2   | Greece
 ...
```

Exemplo: retorna um tipo HLLSKETCH de esboços combinados em uma subconsulta

O exemplo a seguir retorna um único tipo HLLSKETCH que representa a combinação de esboços individuais de uma subconsulta. Os esboços são combinados usando a função agregada HLL_COMBINE.

```
SELECT hll_combine(sketch)
FROM (SELECT hll_create_sketch(customer) AS sketch
        FROM Sales
        GROUP BY country) AS hll_subquery

                                        hll_combine
-----------------------------------------------------------------------------------------------
 {"version":1,"logm":15,"sparse":{"indices":[29808639,35021072,47612452],"values":
[1,1,1]}}
(1 row)
```

Exemplo: retorna um esboço do HyperLogLog da combinação de vários esboços

Para o exemplo a seguir, suponha que a tabela `page-users` armazena esboços pré-agregados para cada página que os usuários visitaram em um determinado site. Cada linha nesta tabela contém um esboço HyperLogLog que representa todos os IDs de usuário que mostram as páginas visitadas.

```
page_users
-- +----------------+-------------+--------------+
-- | _PARTITIONTIME | page        | sketch |
-- +----------------+-------------+--------------+
-- | 2019-07-28     | homepage    | CHAQkAQYA... |
-- | 2019-07-28     | Product A   | CHAQxPnYB... |
-- +----------------+-------------+--------------+
```

O exemplo a seguir une os vários esboços pré-agregados e gera um único esboço. Este esboço encapsula a cardinalidade coletiva que cada esboço encapsula.

```
SELECT hll_combine(sketch) as sketch
FROM page_users
```

A saída será semelhante à seguinte.

```
-- +----------------------------------------+
-- | sketch |
-- +----------------------------------------+
-- | CHAQ3sGoCxgCIAuCB4iAIBgTIBgqgIAgAwY.... |
-- +----------------------------------------+
```

Quando um novo esboço é criado, você pode usar a função HLL_CARDINALITY para obter os valores distintos coletivos, como mostrado a seguir.

```
SELECT hll_cardinality(sketch)
FROM (
  SELECT
  hll_combine(sketch) as sketch
  FROM page_users
) AS hll_subquery
```

A saída será semelhante à seguinte.

```
-- +-------+
-- | count |
-- +-------+
-- | 54356 |
-- +-------+
```

Exemplo: gerar esboços do HyperLogLog sobre dados do S3 usando tabelas externas

Os exemplos a seguir armazenam em cache os esboços do HyperLogLog para evitar acessar diretamente o Amazon S3 para estimativa de cardinalidade.

Você pode pré-agregar e armazenar em cache esboços do HyperLogLog em tabelas externas definidas para armazenar dados do Amazon S3. Ao fazer isso, você pode extrair estimativas de cardinalidade sem acessar os dados básicos subjacentes.

Por exemplo, suponha que você tenha descarregado um conjunto de arquivos de texto delimitados por tabulação no Amazon S3. Execute a consulta a seguir para definir uma tabela externa chamada

`sales` no esquema externo do Amazon Redshift chamado`spectrum`. O bucket do Amazon S3 para este exemplo está na do Leste dos EUA (Norte da Virgínia) Região da AWS.

```
create external table spectrum.sales(
salesid integer,
listid integer,
sellerid smallint,
buyerid smallint,
eventid integer,
dateid integer,
qtysold integer,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp)
row format delimited
fields terminated by '\t' stored as textfile
location 's3://redshift-downloads/tickit/spectrum/sales/';
```

Suponha que você queira computar os compradores distintos que compraram um item em datas arbitrárias. Para fazer isso, o exemplo a seguir gera esboços para os IDs do comprador para cada dia do ano e armazena o resultado na tabela do Amazon Redshift `hll_sales`.

```
CREATE TABLE hll_sales AS
SELECT saletime, hll_create_sketch(buyerid) AS sketch
FROM spectrum.sales
GROUP BY saletime;

SELECT TOP 5 * FROM hll_sales;
```

A saída será semelhante à seguinte.

```
-- hll_sales

-- | saletime        | sketch
    |
-- +-----------------
+-------------------------------------------------------------------+
-- | 7/22/2008 8:30  | {"version":1,"logm":15,"sparse":{"indices":[9281416],"values":
[1]}}
-- | 2/19/2008 0:38  | {"version":1,"logm":15,"sparse":{"indices":[48735497],"values":
[3]}}
```

```
-- | 11/5/2008 4:49  | {"version":1,"logm":15,"sparse":{"indices":[27858661],"values":
[1]}}
-- | 10/27/2008 4:08 | {"version":1,"logm":15,"sparse":{"indices":[65295430],"values":
[2]}}
-- | 2/16/2008 9:37  | {"version":1,"logm":15,"sparse":{"indices":[56869618],"values":
[2]}}
-- +----------------
 +-----------------------------------------------------------------------+
```

A consulta a seguir mostra o número estimado de consumidores distintos que compraram um item durante a sexta-feira após a Ação de Graças em 2008.

```
SELECT hll_cardinality(hll_combine(sketch)) as distinct_buyers
FROM hll_sales
WHERE trunc(saletime) = '2008-11-28';
```

A saída será semelhante à seguinte.

```
distinct_buyers
---------------
386
```

Suponha que você queira o número de usuários distintos que compraram um item em um determinado intervalo de datas. Um exemplo pode ser da sexta-feira após o Dia de Ação de Graças até a segunda-feira seguinte. Para obter isso, a consulta a seguir usa a função agregada `hll_combine`. Com essa função, você pode evitar a contagem duplicada de compradores que compraram um item em mais de um dia do intervalo selecionado.

```
SELECT hll_cardinality(hll_combine(sketch)) as distinct_buyers
FROM hll_sales
WHERE saletime BETWEEN '2008-11-28' AND '2008-12-01';
```

A saída será semelhante à seguinte.

```
distinct_buyers
---------------
1166
```

Para manter a tabela `hll_sales` atualizada, execute a consulta a seguir no final de cada dia. Isso gera um esboço do HyperLogLog com base nos IDs dos compradores que compraram um item hoje e o adiciona à tabela `hll_sales`.

```
INSERT INTO hll_sales
SELECT saletime, hll_create_sketch(buyerid)
FROM spectrum.sales
WHERE TRUNC(saletime) = to_char(GETDATE(), 'YYYY-MM-DD')
GROUP BY saletime;
```

Consultas entre bancos de dados

Este tópico descreve consultas entre bancos de dados, que são consultas que operam em vários bancos de dados do Amazon Redshift em um único cluster do Amazon Redshift.

Ao usar consultas entre bancos de dados no Amazon Redshift, você pode consultar e gravar em bancos de dados em um cluster do Amazon Redshift. Com consultas entre bancos de dados, é possível consultar e gravar dados de qualquer banco de dados no cluster do Amazon Redshift, independentemente do banco de dados que está conectado no momento. As consultas entre bancos de dados eliminam cópias de dados e simplificam sua organização de dados para oferecer suporte a vários grupos de negócios do mesmo data warehouse.

Com consultas entre bancos de dados, você pode fazer o seguinte:

- Consultar dados entre bancos de dados em seu cluster do Amazon Redshift.

  Você não só pode consultar de bancos de dados aos quais você está conectado, como também pode ler de quaisquer outros bancos de dados para os quais você tenha permissões.

  Quando você consulta objetos de banco de dados em qualquer outro banco de dados não conectado, você tem acesso de leitura somente a esses objetos de banco de dados. Você pode usar consultas entre bancos de dados para acessar dados de qualquer um dos bancos de dados em seu cluster do Amazon Redshift sem precisar se conectar a esse banco de dados específico. Fazer isso pode ajudar você a consultar e unir dados distribuídos em vários bancos de dados em seu cluster do Amazon Redshift de forma rápida e fácil.

  Você também pode unir conjuntos de dados de vários bancos de dados em uma única consulta e analisar os dados usando Business Intelligence (BI) ou ferramentas de análise. Você pode usar comandos SQL padrão do Amazon Redshift para continuar a configurar controles detalhados de acesso de leitura em nível de tabela para os usuários. Dessa forma, você pode ajudar a garantir que os usuários vejam somente os subconjuntos relevantes dos dados aos quais têm permissão de acesso.
- Grave dados em bancos de dados no cluster do Amazon Redshift.

  Você pode gravar de bancos de dados que estão conectados no momento e escrever de qualquer outro banco de dados para o qual tenha permissão.

  Quando você tem permissões de gravação em objetos de banco de dados em outros bancos de dados não conectados, é possível usar consultas entre bancos de dados para gravar dados de

qualquer banco de dados no cluster do Amazon Redshift, sem precisar se conectar a esse banco de dados específico. Isso pode ajudar você em operações de gravação complexas e, ao mesmo tempo, unir dados espalhados por vários bancos de dados no cluster do Amazon Redshift de forma rápida e fácil.

Você também pode unir conjuntos de dados de vários bancos de dados em uma única consulta e gravar os dados usando diferentes ferramentas de extração, transformação e carregamento (ETL) ou analytics. Você pode usar comandos SQL padrão do Amazon Redshift para continuar a configurar controles de acesso detalhados em nível de tabela para os usuários. Isso garante que os usuários vejam apenas os subconjuntos relevantes dos dados para os quais eles têm permissões.

- Objetos de consulta.

  Você pode consultar outros objetos de banco de dados usando nomes de objeto totalmente qualificados expressos com a notação de três partes. O caminho completo para qualquer objeto de banco de dados consiste em três componentes: nome do banco de dados, esquema e nome do objeto. Você pode acessar qualquer objeto de qualquer outro banco de dados usando a notação de caminho completo ., *database_name*.*schema_name*.*object_name*. Para acessar uma coluna específica, use *database_name*.*schema_name*.*object_name*.*column_name*.

  Você também pode criar um alias para um esquema em outro banco de dados usando a notação de esquema externo. Este esquema externo faz referência a outro banco de dados e esquema par. Consulta pode acessar o outro objeto de banco de dados usando a notação de esquema externo ., *external_schema_name*.*object_name*.

  Na mesma consulta somente leitura, você pode consultar vários objetos de banco de dados, como tabelas de usuário, visualizações regulares, visões materializadas e visualizações de vinculação tardia de outros bancos de dados.

- Gerencie permissões.

  Usuários com privilégios de acesso a objetos em banco de dados no cluster do Amazon Redshift podem consultar e gravar dados nesses objetos. Você concede privilégios a usuários e grupos de usuários usando comando GRANT. Também é possível revogar privilégios usando o comando REVOKE quando um usuário não requer mais o acesso a objetos de banco de dados específicos.

- Trabalhe com metadados e ferramentas de BI.

Você pode criar um esquema externo para fazer referência a um esquema em outro banco de dados do Amazon Redshift dentro do mesmo cluster do Amazon Redshift. Para obter mais informações, consulte o comando CREATE EXTERNAL SCHEMA.

Depois que as referências de esquema externo são criadas, o Amazon Redshift mostra as tabelas sob o esquema do outro banco de dados em SVV_EXTERNAL_TABLES e SVV_EXTERNAL_COLUMNS para obter as ferramentas para explorar os metadados.

Para integrar consulta entre bancos de dados com ferramentas de BI, você pode usar as visualizações do sistema a seguir. Isso ajuda a visualizar informações sobre os metadados de objetos nos bancos de dados conectados e em outros bancos de dados no cluster do Amazon Redshift.

Veja a seguir visualizações do sistema que mostram todos os objetos do Amazon Redshift e objetos externos de todos os bancos de dados em seu cluster do Amazon Redshift:

- SVV_ALL_COLUMNS
- SVV_ALL_SCHEMAS
- SVV_ALL_TABLES

Veja a seguir visualizações do sistema que mostram todos os objetos do Amazon Redshift de todos os bancos de dados no cluster do Amazon Redshift:

- SVV_REDSHIFT_COLUMNS
- SVV_REDSHIFT_DATABASES
- SVV_REDSHIFT_FUNCTIONS
- SVV_REDSHIFT_SCHEMAS
- SVV_REDSHIFT_TABLES

Tópicos

- Considerações
- Limitações
- Exemplos de consulta entre bancos de dados
- Usar consultas entre bancos de dados com o editor de consultas

Considerações

Este tópico apresenta detalhes sobre consultas entre bancos de dados no Amazon Redshift.

Ao trabalhar com o recurso de consulta entre bancos de dados no Amazon Redshift, considere o seguinte:

- O Amazon Redshift permite consultas entre bancos de dados em todos os tipos de nós RG e RA3 e namespaces com tecnologia sem servidor.
- O Amazon Redshift oferece suporte à junção de dados de tabelas ou visualizações em um ou mais bancos de dados no mesmo cluster do Amazon Redshift.
- Todas as consultas em uma transação no banco de dados conectado ler dados no mesmo estado do outro banco de dados como os dados estavam no início da transação. Essa abordagem ajuda a fornecer consistência transacional de consulta entre bancos de dados. O Amazon Redshift oferece suporte à consistência transacional para consultas entre bancos de dados.
- Para obter metadados entre bancos de dados, use as visualizações de metadados SVV_ALL* e SVV_REDSHIFT*. Não é possível usar a notação de três partes ou esquemas externos para consultar tabelas ou exibições de metadados entre bancos de dados em information_schema e pg_catalog.

Limitações

Este tópico descreve as limitações referentes a consultas entre bancos de dados no Amazon Redshift.

Ao trabalhar com o recurso de consulta entre bancos de dados no Amazon Redshift, esteja ciente das seguintes limitações:

- Você não pode consultar visualizações que são criadas em outros bancos de dados que se referem a objetos de outro banco de dados.
- Não é possível gravar em esquemas externos para objetos de banco de dados em nenhum outro banco de dados não conectado.
- Você só pode criar visões materializadas e de vinculação tardia em objetos de outros bancos de dados no cluster. Você não pode criar visualizações regulares em objetos de outros bancos de dados no cluster.

- O Amazon Redshift não oferece suporte a tabelas com privilégios em nível de coluna para consultas entre bancos de dados.
- A realização de consultas entre bancos de dados em tabelas com chaves de classificação intercaladas não é aceita.

Exemplos de consulta entre bancos de dados

Este tópico contém exemplos sobre como usar consultas entre bancos de dados. As consultas entre bancos de dados são consultas que operam em vários bancos de dados em um único cluster do Amazon Redshift.

Use os exemplos a seguir para ajudar a saber como configurar uma consulta entre bancos de dados que faça referência a um banco de dados do Amazon Redshift.

Para iniciar, crie os bancos de dados `db1` e `db2` e também os usuários `user1` e `user2` no cluster do Amazon Redshift. Para obter mais informações, consulte CREATE DATABASE e CREATE USER.

```
--As user1 on db1
CREATE DATABASE db1;

CREATE DATABASE db2;

CREATE USER user1 PASSWORD 'Redshift01';

CREATE USER user2 PASSWORD 'Redshift01';
```

Como `user1` no `db1`, crie uma tabela, conceda privilégios de acesso ao `user2` e insira valores em `table1`. Para obter mais informações, consulte GRANT e INSERT.

```
--As user1 on db1
CREATE TABLE table1 (c1 int, c2 int, c3 int);

GRANT SELECT ON table1 TO user2;

INSERT INTO table1 VALUES (1,2,3),(4,5,6),(7,8,9);
```

Como `user2` no `db2`, execute uma consulta entre bancos de dados no `db2` usando a notação de três partes.

```
--As user2 on db2
SELECT * from db1.public.table1 ORDER BY c1;

c1 | c2  | c3
---+-----+----
1  |  2  | 3
4  |  5  | 6
7  |  8  | 9
(3 rows)
```

Agora conceda privilégios de gravação ao `user2` e insira valores na `table1` no db1 como `user2`.

```
--As user1 on db1
GRANT INSERT ON table1 TO user2;
```

Como `user2` no db2, execute uma consulta entre bancos de dados no db2 usando a notação de três partes para inserir dados na `table1` no db1.

```
--As user2 on db2
INSERT INTO db1.public.table1 VALUES (10,11,12);
SELECT * from db1.public.table1 ORDER BY c1;

c1  | c2   | c3
----+------+----
1   |  2   | 3
4   |  5   | 6
7   |  8   | 9
10  |  11  | 12
(4 rows)
```

Como `user2` no db2, crie um esquema externo e execute uma consulta entre bancos de dados no db2 usando a notação do esquema externo.

```
--As user2 on db2
CREATE EXTERNAL SCHEMA db1_public_sch
FROM REDSHIFT DATABASE 'db1' SCHEMA 'public';

SELECT * FROM db1_public_sch.table1 ORDER BY c1;

c1  | c2 | c3
----+----+----
```

```
1   | 2  | 3
4   | 5  | 6
7   | 8  | 9
10  | 11 | 12
(4 rows)
```

Para criar exibições diferentes e conceder permissões a essas exibições, como `user1` no `db1`, faça como a seguir.

```
--As user1 on db1
CREATE VIEW regular_view AS SELECT c1 FROM table1;

GRANT SELECT ON regular_view TO user2;


CREATE MATERIALIZED VIEW mat_view AS SELECT c2 FROM table1;

GRANT SELECT ON mat_view TO user2;


CREATE VIEW late_bind_view AS SELECT c3 FROM public.table1 WITH NO SCHEMA BINDING;

GRANT SELECT ON late_bind_view TO user2;
```

Como `user2` no `db2`, execute a consulta de banco de dados a seguir usando a notação de três partes para exibir o modo de exibição específico.

```
--As user2 on db2
SELECT * FROM db1.public.regular_view;

c1
----
1
4
7
10
(4 rows)

SELECT * FROM db1.public.mat_view;

c2
----
```

```
2
5
8
11
(4 rows)

SELECT * FROM db1.public.late_bind_view;

c3
----
3
6
9
12
(4 rows)
```

Como `user2` no `db2`, execute a seguinte consulta entre bancos de dados usando a notação de esquema externo para consultar a visualização de vinculação tardia.

```
--As user2 on db2
SELECT * FROM db1_public_sch.late_bind_view;

c3
----
3
6
9
12
(4 rows)
```

Como `user2` no `db2`, execute o comando a seguir usando tabelas conectadas em uma única consulta.

```
--As user2 on db2
CREATE TABLE table1 (a int, b int, c int);

INSERT INTO table1 VALUES (1,2,3), (4,5,6), (7,8,9);

SELECT a AS col_1, (db1.public.table1.c2 + b) AS sum_col2, (db1.public.table1.c3 + c)
 AS sum_col3 FROM db1.public.table1, table1 WHERE db1.public.table1.c1 = a;
col_1 | sum_col2 | sum_col3
------+----------+----------
```

```
1     | 4        | 6
4     | 10       | 12
7     | 16       | 18
(3 rows)
```

O exemplo a seguir lista todos os bancos de dados do no cluster.

```
select database_name, database_owner, database_type
from svv_redshift_databases
where database_name in ('db1', 'db2');

 database_name | database_owner | database_type
---------------+----------------+---------------
 db1           |            100 | local
 db2           |            100 | local
(2 rows)
```

O exemplo a seguir lista todos os esquemas do Amazon Redshift de todos os bancos de dados no cluster.

```
select database_name, schema_name, schema_owner, schema_type
from svv_redshift_schemas
where database_name in ('db1', 'db2');

 database_name |    schema_name     | schema_owner | schema_type
---------------+--------------------+--------------+-------------
 db1           | pg_catalog         |            1 | local
 db1           | public             |            1 | local
 db1           | information_schema |            1 | local
 db2           | pg_catalog         |            1 | local
 db2           | public             |            1 | local
 db2           | information_schema |            1 | local
(6 rows)
```

O exemplo a seguir lista todas as tabelas ou exibições do Amazon Redshift de todos os bancos de dados no cluster.

```
select database_name, schema_name, table_name, table_type
from svv_redshift_tables
where database_name in ('db1', 'db2') and schema_name in ('public');
```

```
 database_name | schema_name |     table_name      | table_type
---------------+-------------+---------------------+------------
 db1           | public      | late_bind_view      | VIEW
 db1           | public      | mat_view            | VIEW
 db1           | public      | mv_tbl__mat_view__0 | TABLE
 db1           | public      | regular_view        | VIEW
 db1           | public      | table1              | TABLE
 db2           | public      | table2              | TABLE
(6 rows)
```

O exemplo a seguir lista todos os esquemas do Amazon Redshift e externos de todos os bancos de dados no cluster.

```
select database_name, schema_name, schema_owner, schema_type
from svv_all_schemas where database_name in ('db1', 'db2') ;

 database_name |    schema_name     | schema_owner | schema_type
---------------+--------------------+--------------+-------------
 db1           | pg_catalog         |            1 | local
 db1           | public             |            1 | local
 db1           | information_schema |            1 | local
 db2           | pg_catalog         |            1 | local
 db2           | public             |            1 | local
 db2           | information_schema |            1 | local
 db2           | db1_public_sch     |            1 | external
(7 rows)
```

O exemplo a seguir lista todos os Amazon Redshift e tabelas externas de todos os bancos de dados no cluster.

```
select database_name, schema_name, table_name, table_type
from svv_all_tables
where database_name in ('db1', 'db2') and schema_name in ('public');

 database_name | schema_name |     table_name      | table_type
---------------+-------------+---------------------+------------
 db1           | public      | regular_view        | VIEW
 db1           | public      | mv_tbl__mat_view__0 | TABLE
 db1           | public      | mat_view            | VIEW
 db1           | public      | late_bind_view      | VIEW
 db1           | public      | table1              | TABLE
 db2           | public      | table2              | TABLE
```

```
(6 rows)
```

Usar consultas entre bancos de dados com o editor de consultas

Este tópico explica como usar consultas entre bancos de dados com o Editor de Consultas. As consultas entre bancos de dados são consultas que operam em vários bancos de dados em um único cluster do Amazon Redshift.

Você pode usar consultas entre bancos de dados para acessar dados de qualquer um dos bancos de dados em seu cluster do Amazon Redshift sem precisar se conectar a esse banco de dados específico. Ao executar consultas entre bancos de dados em outros bancos de dados não conectados, você tem acesso de leitura e gravação apenas aos objetos desses bancos de dados.

Você pode consultar e gravar em outros objetos de banco de dados utilizando nomes de objeto totalmente qualificados, expressos com a notação de três partes. O caminho completo para qualquer objeto de banco de dados consiste em três componentes: nome do banco de dados, esquema e nome do objeto. Um exemplo é *database_name.schema_name.object_name*.

Para usar consultas entre bancos de dados com o editor de consultas v2

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. Crie um cluster para usar consultas entre bancos de dados no editor de consultas do Amazon Redshift v2. Para obter mais informações, consulte “Criar um cluster” no Guia de gerenciamento de clusters do Amazon Redshift.
3. Habilite o acesso ao editor de consultas com as permissões apropriadas. Para obter mais informações, consulte Querying a database using the query editor v2 no Guia de gerenciamento do Amazon Redshift.
4. No menu de navegação, escolha Editor de consultas v2 e se conecte a um banco de dados no cluster.

   Quando você se conecta ao editor de consultas v2 pela primeira vez, o Amazon Redshift mostra os recursos do banco de dados conectado por padrão.
5. Escolha os outros bancos de dados que você tem acesso para exibir objetos de banco de dados para esses outros bancos de dados. Para visualizar objetos, verifique se você tem as permissões apropriadas. Depois de escolher um banco de dados, o Amazon Redshift mostra a lista de esquemas do banco de dados.

Selecione um esquema para ver a lista de objetos de banco de dados dentro desse esquema.

> **Note**
>
> O Amazon Redshift não oferece suporte diretamente a objetos de catálogo de consulta que fazem parte do AWS Glue ou bancos de dados federados. Para consultá-los, primeiro crie esquemas externos que se referem a essas fontes de dados externas em cada banco de dados.
>
> As consultas entre bancos de dados do Amazon Redshift com notação de três partes não oferecem suporte a tabelas de metadados nos esquemas `information_schema` e `pg_catalog` porque essas exibições de metadados são específicas para um banco de dados.

6. (Opcional) Filtre a lista de tabelas ou exibições para o esquema selecionado.

Compatibilidade com o Apache Iceberg para o Amazon Redshift

Você pode registrar clusters provisionados do Amazon Redshift ou namespaces sem servidor inteiros no AWS Glue Data Catalog para criar catálogos que compartilham dados ativos com segurança entre contas da AWS. Você pode acessar esses catálogos por meio de qualquer mecanismo de consulta SQL compatível com a API REST do Apache Iceberg. O AWS Lake Formation gerencia as permissões para os catálogos, consentindo que você gerencie uma única cópia de dados com um único conjunto de permissões, ao mesmo tempo em que aproveita os recursos do Amazon Redshift, como visões materializadas e Integrações ETL zero.

Todos os catálogos criados de clusters provisionados registrados e namespaces sem servidor do Amazon Redshift no AWS Glue Data Catalog são montados automaticamente como bancos de dados externos em todos os clusters provisionados e grupos de trabalho sem servidor na mesma Região da AWS e na mesma conta. Os catálogos criados no AWS Glue Data Catalog para armazenar dados no Redshift Managed Storage (RMS) são montados de forma semelhante como bancos de dados externos. Depois que os bancos de dados forem montados, você poderá se conectar diretamente a esses bancos de dados e consultar os objetos usando a notação de três partes `database@namespace-catalog.schema.table`.

Regiões onde a compatibilidade com o Apache Iceberg está disponível

A compatibilidade com o Apache Iceberg para o Amazon Redshift está disponível nas seguintes Regiões da AWS:

- Leste dos EUA (Norte da Virgínia)
- Leste dos EUA (Ohio)
- Oeste dos EUA (N. da Califórnia)
- Ásia-Pacífico (Hong Kong)
- Ásia-Pacífico (Seul)
- Ásia-Pacífico (Singapura)
- Ásia-Pacífico (Sydney)

- Ásia-Pacífico (Tóquio)
- Canadá (Central)
- Europa (Frankfurt)
- Europa (Irlanda)
- Europe (London)
- Europe (Stockholm)
- América do Sul (São Paulo)

Considerações e limitações ao usar catálogos do Amazon Redshift no AWS Glue Data Catalog

Ao usar catálogos do Amazon Redshift no AWS Glue Data Catalog, considere o seguinte:

- Os data warehouses registrados no AWS Glue Data Catalog seguem a sintaxe de três partes para acessar tabelas (`database@namespace-catalog.schema.table`). Por exemplo, se você registrou um namespace do Amazon Redshift chamado “a”, preenchido por um banco de dados chamado “b”, que continha um esquema chamado “c”, no qual havia uma tabela chamada “d”, você selecionaria “d” usando a seguinte instrução:

  ```
  SELECT * FROM b@a.c.d;
  ```

  Observe que o tamanho total da parte `database@namespace-catalog` da sintaxe deve ser de 127 caracteres ou menos.
- Quando você registra um cluster ou namespace no AWS Glue Data Catalog, o Amazon Redshift registra todos os bancos de dados e relações nesse cluster ou namespace.
- Você pode registrar vários clusters e namespaces do Redshift no AWS Glue Data Catalog.
- Ao registrar um cluster ou namespace, apenas os esquemas e as relações internas são registradas nesse cluster ou namespace. Os seguintes itens não são registrados:
  - Esquemas externos.
  - Tabelas externas. Observe que as visualizações de vinculação tardia criadas com base em tabelas externas serão registradas.
  - Funções criadas pelo usuário.
  - Procedimentos.

  - Tabelas com segurança em nível de linha ou políticas de mascaramento de dados dinâmicos anexadas.
  - Objetos de banco de dados com nomes em letras maiúsculas ou maiúsculas e minúsculas. Tabelas contendo colunas em letras maiúsculas ou em maiúsculas e minúsculas não são registradas. Isso se aplica mesmo quando enable_case_sensitive_identifier está desabilitado.
- As permissões do banco de dados do Amazon Redshift, como perfis concedidos pelo controle de acesso por perfil, não são transferidas para catálogos no AWS Glue Data Catalog. Use o AWS Lake Formation para configurar permissões para o AWS Glue Data Catalog. Para obter mais informações sobre como usar o Lake Formation para configurar permissões, consulte Configuração de permissões para unidades de compartilhamento de dados do Amazon Redshift no Guia do desenvolvedor do AWS Lake Formation.
- Ao criar um catálogo de um cluster registrado ou namespace sem servidor, o AWS Glue Data Catalog cria um grupo de trabalho gerenciado do Amazon Redshift usando os recursos de computação do Amazon Redshift para lidar com os requisitos de computação ao consultar esse catálogo. Você pode visualizar o grupo de trabalho gerenciado no console do Amazon Redshift sem servidor e gerenciá-lo no AWS Glue.
- Quando você registra um cluster pausado, o AWS Glue Data Catalog não monta esse cluster como um catálogo enquanto o cluster não for retomado.
- Quando você registra um namespace sem servidor que não está sendo usado ativamente, o AWS Glue Data Catalog não monta esse namespace como um catálogo enquanto o namespace não for usado novamente.
- Para acessar tabelas em data warehouses registrados no AWS Glue Data Catalog, o nível de isolamento do banco de dados deve ser SNAPSHOT. Tentar acessar tabelas com um nível de isolamento SERIALIZABLE resultará em um erro. Para obter mais informações sobre isolamento serializável, consulte Níveis de isolamento no Amazon Redshift. Para informações sobre como alterar o nível de isolamento de um banco de dados, consulte ALTER DATABASE.

  Observe que o nível de isolamento do banco de dados padrão `dev` não pode ser alterado. Consequentemente, isso significa que as tabelas no banco de dados `dev` não serão incluídas nos dados acessíveis ao registrar data warehouses no AWS Glue Data Catalog.

Requisitos de política do IAM para acessar clusters e namespaces registrados no Catálogo de Dados

Este tópico descreve as permissões do IAM necessárias para registrar clusters provisionados e namespaces sem servidor no Catálogo de Dados e acessá-los com o Amazon Redshift.

Depois de registrar um cluster provisionado ou namespace sem servidor no AWS Glue Data Catalog, você precisa das permissões a seguir para descobrir quando um catálogo é criado e as alterações subsequentes nesse catálogo.

- `glue:GetCatalog`
- `glue:GetCatalogs`
- `lakeformation:GetDataAccess`

Essas permissões estão incluídas na função vinculada ao serviço `AmazonRedshiftServiceLinkedRolePolicy`. Para obter mais informações sobre essa função, consulte Uso de funções vinculadas ao serviço para o Amazon Redshift no Guia de gerenciamento do Amazon Redshift.

Registrar clusters e namespaces do Amazon Redshift no AWS Glue Data Catalog

Você pode adicionar clusters provisionados e namespaces sem servidor do Amazon Redshift ao AWS Glue Data Catalog para acessá-los usando a API REST do Apache Iceberg. Para isso, registre o data warehouse do Amazon Redshift no AWS Glue Data Catalog usando o console do Amazon Redshift ou a AWS CLI e crie um catálogo federado do Amazon Redshift para o data warehouse usando oAWS Lake Formation.

Os data warehouses do Amazon Redshift registrados no Catálogo de Dados atuam como unidades de compartilhamento de dados de produtor. As alterações feitas em clusters ou namespaces sem servidor com base em catálogos são refletidas no cluster ou namespace no Redshift e vice-versa.

Registering using the Amazon Redshift console

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.

2. Acesse o namespace sem servidor ou o cluster provisionado que você deseja registrar e selecione-o.
3. Na página de detalhes do namespace ou cluster, selecione Registrar no AWS Glue Data Catalog no menu suspenso Ações.
4. Insira o ID da conta de destino na qual você deseja registrar o namespace ou o cluster e escolha Registrar.
5. Se estiver usando a sua conta para fazer o registro na mesma conta no AWS Glue, o console do AWS Lake Formation será aberto para você concluir o processo. Se estiver fazendo o registro em uma conta diferente, aparecerá um link que direcionará você ao Lake Formation.

Registering using the AWS CLI

Para registrar um cluster ou namespace no AWS Glue Data Catalog usando a AWS CLI, use o comando `register-namespace` com as seguintes opções:

- `namespace-identifier`: um objeto com o identificador único do cluster ou namespace que você está registrando. Esse objeto será diferente dependendo se você está registrando um cluster provisionado ou um namespace sem servidor. Considere o seguinte:
  - Para clusters provisionados, você fornece um objeto `ProvisionedIdentifier`, que contém um objeto `ClusterIdentifier` com o identificador exclusivo do cluster que você está registrando.
  - Para namespaces sem servidor, você fornece um objeto `ServerlessIdentifier`, que contém um objeto `NamespaceIdentifier` com o identificador único do namespace que você está registrando, bem como um objeto `WorkgroupIdentifier` com o identificador único do grupo de trabalho associado a esse namespace.
- `consumer-identifiers`: uma matriz com um único elemento, contendo o identificador exclusivo da conta na qual você está registrando o cluster ou namespace.

O exemplo a seguir registra o namespace sem servidor `mySampleNamespace` no ID `012345678910` da conta.

```
aws redshift register-namespace /
--namespace-identifier {ServerlessIdentifier: {NamespaceIdentifer:
 mySampleNamespace, WorkgroupIdentifier: mySampleWorkgroup}} /
--consumer-identifiers [012345678910]
```

Cancelar o registro de clusters e namespaces do Amazon Redshift no AWS Glue Data Catalog

Você pode cancelar o registro de um cluster provisionado ou namespace sem servidor no AWS Glue Data Catalog usando o console do Amazon Redshift ou a AWS CLI. Os data warehouses dos quais você cancela o registro no AWS Glue Data Catalog são removidos apenas do próprio AWS Glue Data Catalog. Os warehouses continuam fazendo parte da conta do Amazon Redshift. Além disso, o catálogo permanece no AWS Glue e deve ser removido manualmente. Quando você remove o catálogo do AWS Glue, o grupo de trabalho gerenciado associado a ele também é removido.

Deregistering using the Amazon Redshift console

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. Acesse o namespace sem servidor ou cluster provisionado cujo registro você deseja cancelar e selecione-o.
3. Na página de detalhes do namespace ou cluster, selecione Cancelar registro do AWS Glue Data Catalog no menu suspenso Ações. Essa opção só aparecerá se você tiver escolhido um data warehouse que já tenha sido registrado no AWS Glue Data Catalog.
4. Insira o ID da conta que tem o catálogo que contém o cluster ou namespace e escolha Cancelar registro.

Deregistering using the AWS CLI

Para cancelar o registro de um cluster ou namespace no AWS Glue Data Catalog usando a AWS CLI, use o comando `deregister-namespace` com as seguintes opções:

- `namespace-identifier`: um objeto com o identificador exclusivo do cluster ou namespace do qual você está cancelando o registro. Esse objeto será diferente dependendo se você está cancelando o registro de um cluster provisionado ou de um namespace sem servidor. Considere o seguinte:
  - Para clusters provisionados, você fornece um objeto `ProvisionedIdentifier`, que contém um objeto `ClusterIdentifier` com o identificador exclusivo do cluster do qual você está cancelando o registro.
  - Para namespaces sem servidor, você fornece um objeto `ServerlessIdentifier`, que contém um objeto `NamespaceIdentifier` com o identificador exclusivo do namespace do

qual você está cancelando o registro, bem como um objeto `WorkgroupIdentifier` com o identificador exclusivo do grupo de trabalho associado a esse namespace.

- `consumer-identifiers`: uma matriz com um único elemento, contendo o identificador exclusivo da conta da qual você está cancelando o registro do cluster ou namespace.

O exemplo a seguir cancela o registro do namespace sem servidor `mySampleNamespace` no ID de conta `012345678910`.

```
aws redshift deregister-namespace /
--namespace-identifier {ServerlessIdentifier: {NamespaceIdentifer:
 mySampleNamespace, WorkgroupIdentifier: mySampleWorkgroup}} /
--consumer-identifiers [012345678910]
```

Grupos de trabalho gerenciados

Quando você registra um cluster provisionado ou namespace sem servidor no AWS Glue Data Catalog e cria um catálogo com base nele, o AWS Glue cria um grupo de trabalho gerenciado para fornecer recursos de computação para qualquer mecanismo de consulta SQL que acesse esse catálogo. Por exemplo, um usuário do Amazon Athena que consulte uma tabela em um catálogo teria os requisitos de computação atendidos pelos recursos de computação do Amazon Redshift do grupo de trabalho gerenciado. Grupos de trabalho gerenciados usam recursos de computação do Amazon Redshift, mas são gerenciados no AWS Glue.

Grupos de trabalho gerenciados são contabilizados na cota de grupos de trabalho do Amazon Redshift sem servidor, que por padrão é 25. Para obter mais informações, consulte “Cotas e limites no Amazon Redshift” no Guia de gerenciamento de clusters do Amazon Redshift.

Você pode visualizar informações gerais, métricas de performance e informações de consulta para os grupos de trabalho gerenciados no console do Amazon Redshift sem servidor. Para obter mais informações, consulte Exibir as propriedades de um grupo de trabalho no Guia de gerenciamento do Amazon Redshift.

Consultar catálogos registrados no AWS Glue Data Catalog

Depois que você registra um data warehouse do Amazon Redshift no AWS Glue Data Catalog e define permissões para o catálogo resultante no AWS Lake Formation, o catálogo é montado automaticamente em todas as instâncias do Amazon Redshift com acesso ao data warehouse

de origem na mesma conta e Região da AWS. Você pode consultar esse catálogo como faria com um cluster ou grupo de trabalho local. Você também pode consultar catálogos registrados no AWS Glue Data Catalog usando qualquer mecanismo SQL compatível com a REST OpenAPI do Apache Iceberg. Para obter mais informações sobre como consultar catálogos no AWS Glue Data Catalog usando a API REST do Apache Iceberg, consulte Acessar o Catálogo de Dados no Guia do desenvolvedor do AWS Glue. Para obter informações sobre a API REST do Apache Iceberg, consulte a especificação da REST OpenAPI do Apache Iceberg.

Para consultar um catálogo, você deve primeiro definir as permissões para o catálogo usando o AWS Lake Formation. Para obter mais informações sobre como definir permissões para catálogos no AWS Lake Formation, consulte Configuração de permissões para unidades de compartilhamento de dados do Amazon Redshift no Guia do desenvolvedor do AWS Lake Formation. Você também precisa de um perfil do IAM com a política gerenciada `AmazonRedshiftServiceLinkedRolePolicy` anexada. Para obter informações sobre funções vinculadas ao serviço, consulte Uso de funções vinculadas ao serviço para o Amazon Redshift no Guia de gerenciamento do Amazon Redshift.

Observe que, para acessar tabelas, as consultas em catálogos devem seguir esta sintaxe de três partes:

```
database@namespace.schema.table
```

Para obter informações gerais sobre como consultar data warehouses do Amazon Redshift, consulte Consultar um banco de dados no Guia de gerenciamento do Amazon Redshift.

Querying using the query editor v2

Após a definição de permissões para que uma conta acesse um grupo de trabalho gerenciado, esse grupo de trabalho gerenciado aparece no painel de visualização em árvore na seção de bancos de dados externos do seu banco de dados sem servidor. Você pode consultar o grupo de trabalho gerenciado da mesma forma que consultaria um cluster interno provisionado ou um grupo de trabalho sem servidor do Amazon Redshift usando o formato de sintaxe de três partes `database@namespace/cluster.schema.table`. Veja o seguinte exemplo de declaração:

```
SELECT price FROM sales_db@mynamespace.sales_schema.inventory_table
```

Querying using the Data API

Você pode consultar grupos de trabalho gerenciados usando a API de dados do Amazon Redshift da mesma forma que consultaria um cluster provisionado interno ou um grupo de trabalho sem

servidor do Amazon Redshift passando o nome do recurso da Amazon (ARN) do catálogo para o atributo `database` relevante. Considere o exemplo a seguir, que cria uma tabela em um catálogo.

```
aws redshift-data execute-statement –sql 'CREATE TABLE IF NOT EXISTS "dev@test-
rms-catalog".public.t1 (c1 INT, c2 VARCHAR(10));' –database arn:aws:glue:us-
east-1:550022730026:catalog/test-rms-catalog
```

Consulta de catálogos federados do IRC do AWS Glue com o Amazon Redshift

Com o AWS Glue Iceberg REST Catalog (IRC), você pode federar catálogos de dados externos, como o Databricks Unity Catalog, no AWS Glue Data Catalog. Depois que o catálogo é federado, o Amazon Redshift pode consultar essas tabelas externas diretamente, sem exigir movimentação de dados. Este capítulo descreve como acessar as tabelas do Catálogo do Unity do Databricks usando a federação do Amazon Redshift.

Visão geral da arquitetura

A arquitetura envolve três componentes que podem residir em contas e regiões diferentes:

- Dados de origem (bucket do Amazon S3): o bucket do Amazon S3 gerenciado pela plataforma de dados externa (por exemplo, Databricks). É aqui que residem os arquivos reais de dados da tabela do Iceberg. Para ler tabelas Delta, verifique se os metadados do Iceberg estão disponíveis para essas tabelas usando o UniForm.
- Catálogo federado do IRC do AWS Glue: a entrada do catálogo de dados que federa o catálogo externo (por exemplo, Catálogo do Unity do Databricks) e expõe os metadados por meio do protocolo REST do Iceberg.
- Computação do Amazon Redshift: o cluster provisionado ou grupo de trabalho com tecnologia sem servidor que executa consultas nas tabelas federadas.

É possível acessar os catálogos federados do Amazon Redshift das seguintes maneiras:

- Montagem automática: o Amazon Redshift monta automaticamente o banco de dados `awsdatacatalog`, que fornece acesso a todos os bancos de dados registrados no catálogo de

dados da mesma conta e região. Nenhuma criação de esquema é necessária. As consultas usam notação de três partes: `"awsdatacatalog"."database_name"."table_name"`.

- Esquema externo: você cria explicitamente um esquema externo usando `CREATE EXTERNAL SCHEMA` que aponta para um banco de dados de links de recursos no catálogo de dados, especificando um perfil do IAM estático ou `SESSION` para autorização de identidade federada.

> **Important**
>
> O esquema externo funciona somente quando todos os recursos (catálogo, computação e bucket do Amazon S3) estão na mesma região.

Configuração e consulta de um catálogo federado na mesma conta

Configuração da federação de catálogos

Siga as instruções fornecidas em Federar no Catálogo do Unity do Databricks no Guia do desenvolvedor do AWS Lake Formation.

Criação de um link de recurso no catálogo de dados

Faça login como administrador de data lake. Crie um link de recurso de banco de dados no catálogo de dados que aponte para o banco de dados federado.

Concessão de permissões para acessar tabelas no Amazon Redshift

Como administrador do data lake, conceda as seguintes permissões para a entidade principal do IAM ou o perfil do IAM usado para criar o esquema externo:

- Acesso DESCRIBE no link do recurso do banco de dados
- Acesso DESCRIBE no banco de dados federado
- SELECT e DESCRIBE nas tabelas federadas

Consulta da tabela federada no Amazon Redshift

Conecte-se ao seu cluster ou grupo de trabalho do Amazon Redshift como usuário administrador e acesse a tabela usando uma das seguintes opções.

Etapa 1: criar um esquema externo usando um perfil do IAM

Crie um perfil do IAM com permissão para APIs AWS Glue e AWS Lake Formation e anexe o perfil ao cluster.

```
CREATE EXTERNAL SCHEMA external_schema_name
FROM DATA CATALOG DATABASE 'resource_link_database'
IAM_ROLE 'arn:aws:iam::account_id:role/spectrum_role';
```

Você pode conceder permissões nesse esquema externo para usuários e grupos locais do Amazon Redshift. As identidades locais do Amazon Redshift podem acessar os dados consultando o esquema usando permissões concedidas ao perfil do IAM anexado ao esquema externo.

```
SELECT * FROM "database"."external_schema_name"."table_name" LIMIT 10;
```

Opção 2: criar um esquema externo usando SESSION do IAM para uma entidade principal federada

```
CREATE EXTERNAL SCHEMA external_schema_name
FROM DATA CATALOG DATABASE 'resource_link_database'
IAM_ROLE 'SESSION' CATALOG_ID 'account_id';
```

Faça login como usuário federado ou com uma identidade do IAM e execute o comando a seguir para consultar a tabela:

```
SELECT * FROM "database"."external_schema_name"."table_name" LIMIT 10;
```

Opção 3: usar a montagem automática para consultar uma tabela federada

O usuário administrador deve conceder o uso do banco de dados montado automaticamente para as entidades principais federadas. A entidade principal pode então acessar a tabela federada executando o seguinte comando:

```
SELECT * FROM "awsdatacatalog"."resource_link_database"."table_name" LIMIT 10;
```

Configuração e consulta de um catálogo federado entre contas

Você pode compartilhar o catálogo federado entre contas usando AWS Lake Formation e acessar a tabela compartilhada de uma conta diferente usando o Amazon Redshift.

Conta do concedente (onde reside o catálogo federado)

Configuração da federação de catálogos

Siga as instruções fornecidas em Federar no Catálogo do Unity do Databricks no Guia do desenvolvedor do AWS Lake Formation.

Configuração da versão entre contas

Defina a versão entre contas como V4 nas configurações do AWS Lake Formation. Se já existirem permissões no nível do catálogo, verifique se tem o AWS RAM permissão para compartilhar recursos adicionando a seguinte política de recursos do Catálogo de Dados:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ShareResource"
      ],
      "Principal": {
        "Service": [
          "ram.amazonaws.com"
        ]
      },
      "Resource": [
        "arn:aws:glue:region:grantor_account_id:catalog",
        "arn:aws:glue:region:grantor_account_id:database/*",
        "arn:aws:glue:region:grantor_account_id:table/*"
      ]
    }
  ]
}
```

Compartilhamento de recursos federados entre contas

Essa ação cria automaticamente um convite de compartilhamento de AWS RAM para compartilhar o recurso com a conta do destinatário.

Opção 1: compartilhar na conta para que o administrador do data lake gerencie a delegação

Como administrador do data lake, conceda as seguintes permissões à conta do destinatário:

- Acesso DESCRIBE ao banco de dados federado com permissão concedida
- SELECT e DESCRIBE nas tabelas federadas com permissão concedida

Opção 2: compartilhar diretamente com as entidades principais do IAM entre contas

Como administrador do data lake, conceda as seguintes permissões à entidade principal do IAM na conta do destinatário:

- Acesso DESCRIBE no banco de dados federado
- SELECT e DESCRIBE nas tabelas federadas

Se desejar, você pode conceder as permissões acima à conta sem permitir que o administrador do data lake da conta do destinatário visualize os recursos compartilhados.

Conta do destinatário (onde reside o Amazon Redshift)

Aceitação do compartilhamento de AWS RAM

Faça login como administrador de data lake. Se a aceitação automática estiver definida para contas de membro, o compartilhamento de AWS RAM será aceito automaticamente. Caso contrário, aceite o convite de compartilhamento de AWS RAM. Depois de aceitos, os recursos compartilhados ficam visíveis no console de AWS RAM em “Compartilhado comigo”.

Criação de um link de recurso de banco de dados

Crie um link de recurso de banco de dados no catálogo de dados local que aponte para o banco de dados federado compartilhado. Esse link de recurso serve como referência local que o Amazon Redshift usa para acessar os dados compartilhados.

Concessão de permissões para acessar tabelas no Amazon Redshift

Opção 1: compartilhado no nível da conta

Como administrador do data lake, conceda as seguintes permissões do AWS Lake Formation para a entidade principal do IAM ou o perfil do IAM usado para criar o esquema externo:

- DESCRIBE no banco de dados do link do recurso
- DESCRIBE no banco de dados federado compartilhado
- SELECT e DESCRIBE nas tabelas federadas compartilhadas

Opção 2: compartilhado diretamente com as entidades principais do IAM entre contas

Como administrador do data lake, conceda as seguintes permissões do AWS Lake Formation para a entidade principal do IAM ou o perfil do IAM usado para criar o esquema externo:

- DESCRIBE no banco de dados do link do recurso

Consulta da tabela federada no Amazon Redshift

Conecte-se ao seu cluster ou grupo de trabalho do Amazon Redshift como usuário administrador e acesse a tabela usando uma das seguintes opções.

Etapa 1: criar um esquema externo usando um perfil do IAM

Crie um perfil do IAM com permissão para APIs AWS Glue e AWS Lake Formation e anexe o perfil ao cluster.

```
CREATE EXTERNAL SCHEMA external_schema_name
FROM DATA CATALOG DATABASE 'resource_link_database'
IAM_ROLE 'arn:aws:iam::account_id:role/spectrum_role';
```

Você pode conceder permissões nesse esquema externo para usuários e grupos locais do Amazon Redshift. As identidades locais do Amazon Redshift podem acessar os dados consultando o esquema usando permissões concedidas ao perfil do IAM anexado ao esquema externo.

```
SELECT * FROM "database"."external_schema_name"."table_name" LIMIT 10;
```

Opção 2: criar um esquema externo usando SESSION do IAM para uma entidade principal federada

```
CREATE EXTERNAL SCHEMA external_schema_name
FROM DATA CATALOG DATABASE 'resource_link_database'
IAM_ROLE 'SESSION' CATALOG_ID 'account_id';
```

Faça login como usuário federado ou com uma identidade do IAM e execute o comando a seguir para consultar a tabela:

```
SELECT * FROM "database"."external_schema_name"."table_name" LIMIT 10;
```

Opção 3: usar a montagem automática para consultar uma tabela federada

O usuário administrador deve conceder o uso do banco de dados montado automaticamente para as entidades principais federadas. A entidade principal pode então acessar a tabela federada executando o seguinte comando:

```
SELECT * FROM "awsdatacatalog"."resource_link_database"."table_name" LIMIT 10;
```

Práticas recomendadas e limitações

- Prefira a montagem automática para implantações entre regiões. A montagem automática lida com a resolução entre regiões de forma transparente e não requer sobrecarga de gerenciamento de esquemas.
- Use a implantação na mesma região quando esquemas externos forem necessários. Se o fluxo de trabalho exigir esquemas externos, por exemplo, para controlar permissões ou nomenclatura no nível do esquema, verifique se o catálogo federado do AWS Glue, a computação do Amazon Redshift e o bucket do Amazon S3 de origem estão todos na mesma região.

Compartilhamento de dados no Amazon Redshift

Com o Amazon Redshift, é possível compartilhar dados em tempo real entre clusters do Amazon Redshift ou com outros serviços da AWS. Com o compartilhamento de dados, é possível compartilhar dados em tempo real sem precisar criar uma cópia ou movê-los. Os administradores de banco de dados e engenheiros de dados podem usar o compartilhamento de dados para fornecer acesso seguro e somente leitura aos dados para fins de analytics, mantendo o controle sobre os dados. Os analistas de dados, profissionais de business intelligence e cientistas de dados podem utilizar dados compartilhados para obter insights sem duplicar ou mover dados. Casos de uso comuns incluem compartilhar dados com parceiros, permitir análises multifuncionais e facilitar a democratização de dados dentro de uma organização. As seções a seguir apresentam detalhes da configuração e do gerenciamento do compartilhamento de dados no Amazon Redshift.

Com o compartilhamento de dados do Amazon Redshift, você pode compartilhar com segurança o acesso de leitura a dados em tempo real em clusters do Amazon Redshift, grupos de trabalho, Contas da AWS e Regiões da AWS sem mover ou copiar manualmente os dados. Como os dados ainda estão ativos, todos os usuários podem ver as informações mais atualizadas e consistentes no Amazon Redshift assim que elas são atualizadas.

É possível compartilhar dados entre clusters provisionados, grupos de trabalho sem servidor, zonas de disponibilidade, Contas da AWS e Regiões da AWS. Você pode compartilhar entre tipos de cluster, bem como entre clusters provisionados e sem servidor.

Você pode compartilhar objetos de banco de dados de leituras e gravações em clusters do Amazon Redshift diferentes ou grupos de trabalho do Amazon Redshift Serverless dentro da mesma Conta da AWS ou de uma Conta da AWS para outra. Também é possível ler e gravar dados em todas as regiões. Você pode conceder permissões como SELECT, INSERT e UPDATE para tabelas diferentes e USAGE e CREATE para esquemas diferentes. Os dados permanecem ativos e disponíveis para todos os warehouses assim que uma transação de gravação é confirmada.

Casos de uso de compartilhamento de dados para o Amazon Redshift

O compartilhamento de dados do Amazon Redshift é especialmente útil para esses casos de uso:

- Suporte a diferentes tipos de workloads críticos aos negócios — Use um cluster central de extrair, transformar e carregar (ETL) que compartilhe dados com vários clusters de business intelligence

(BI) ou analíticos. Essa abordagem fornece isolamento da workload de leitura e estorno para workloads individuais. Você pode dimensionar e escalar sua computação de workload individual de acordo com os requisitos específicos do workload de preço e performance.

- Habilitar a colaboração entre grupos — Habilite a colaboração perfeita entre equipes e grupos de negócios para análise mais ampla, ciência de dados e análise de impacto entre produtos.
- Fornecimento de dados como serviço — Compartilhe dados como um serviço em toda a sua organização.
- Compartilhamento de dados entre ambientes — Compartilhe dados entre os ambientes de desenvolvimento, teste e produção. Você pode melhorar a agilidade da equipe compartilhando dados em diferentes níveis de granularidade.
- Licenciar acesso a dados no Amazon Redshift: liste os conjuntos de dados do Amazon Redshift no catálogo do AWS Data Exchange que os clientes podem encontrar, assinar e consultar em minutos.

Casos de uso de compartilhamento de dados com acesso de gravação

O compartilhamento de dados para gravações tem vários casos de uso importantes:

- Atualizar os dados da fonte comercial sobre o produtor: você pode compartilhar dados como um serviço em toda a organização, mas os consumidores também podem realizar ações nos dados de origem. Por exemplo, eles podem comunicar valores atualizados ou confirmar o recebimento de dados. Esses são apenas alguns possíveis casos de uso comercial.
- Inserir registros adicionais sobre o produtor: os consumidores podem adicionar registros aos dados da fonte original. Eles podem ser marcados como do consumidor, se necessário.

Para obter informações específicas sobre como executar operações de gravação em uma unidade de compartilhamento de dados, consulte Compartilhar acesso de gravação a dados.

Compartilhamento de dados em diferentes níveis no Amazon Redshift

Com o Amazon Redshift, você pode compartilhar dados em diferentes níveis. Esses níveis incluem bancos de dados, esquemas, tabelas, visualizações (incluindo visualizações regulares, de vinculação tardia e materializadas) e funções definidas pelo usuário (UDFs) do SQL. Você pode criar vários

datashares para um determinado banco de dados. Um datashare pode conter objetos de vários esquemas no banco de dados no qual o compartilhamento é criado.

Ao ter essa flexibilidade no compartilhamento de dados, você obtém um controle de acesso minucioso. Você pode personalizar esse controle para diferentes usuários e empresas que precisam de acesso aos dados do Amazon Redshift.

Gerenciamento de consistência para compartilhamento de dados no Amazon Redshift

O Amazon Redshift fornece consistência transacional em todos os clusters de produtores e consumidores e compartilha visualizações atualizadas e consistentes dos dados com todos os consumidores.

Você pode atualizar continuamente os dados no cluster de produtores. Todas as consultas em um cluster de consumidores dentro de uma transação ler o mesmo estado dos dados compartilhados. O Amazon Redshift não considera os dados que foram alterados por uma outra transação no cluster de produtores que foi confirmada após o início da transação no cluster de consumidores. Depois que a alteração de dados é confirmada no cluster de produtores, novas transações no cluster de consumidores podem consultar imediatamente os dados atualizados.

A forte consistência elimina os riscos de haver relatórios de negócios menos fidedignos que possam conter resultados inválidos durante o compartilhamento de dados. Esse fator é especialmente importante para a análise financeira ou onde os resultados podem ser usados para preparar conjuntos de dados que são usados para treinar modelos de Machine Learning.

Considerações sobre compartilhamento de dados no Amazon Redshift

Com o compartilhamento de dados do Amazon Redshift, você pode compartilhar com segurança o acesso de leitura a dados em tempo real em clusters do Amazon Redshift, grupos de trabalho, Contas da AWS e Regiões da AWS sem mover ou copiar manualmente os dados. Anteriormente, objetos em unidades de compartilhamento de dados eram somente leitura em todas as circunstâncias. A gravação em um objeto em uma unidade de compartilhamento de dados é um novo recurso. Os objetos em unidades de compartilhamento de dados só são habilitados para gravação quando um produtor concede especificamente privilégios de gravação, como INSERT ou CREATE,

em objetos na unidade de compartilhamento de dados. Além disso, para o compartilhamento entre contas, um produtor precisa autorizar a unidade de compartilhamento de dados para gravações e o consumidor deve associar clusters e grupos de trabalho específicos para gravações.

Esta seção aborda considerações quando se trabalha com compartilhamento de dados no Amazon Redshift.

Tópicos

- Considerações gerais sobre compartilhamento de dados no Amazon Redshift
- Considerações sobre leituras e gravações de compartilhamento de dados no Amazon Redshift
- Considerações sobre o compartilhamento de dados na restauração do Amazon Redshift sem servidor
- Considerações sobre compartilhamento de dados com tabelas de data lake no Amazon Redshift
- Considerações sobre compartilhamento de dados com o AWS Lake Formation no Amazon Redshift
- Considerações sobre compartilhamento de dados com o AWS Data Exchange no Amazon Redshift
- Permissões que você pode conceder a unidades de compartilhamento de dados
- Instruções SQL aceitas para gravações de compartilhamento de dados em consumidores
- Instruções SQL não aceitas para gravações de compartilhamento de dados em consumidores

Considerações gerais sobre compartilhamento de dados no Amazon Redshift

Veja as considerações gerais a seguir para trabalhar com unidades de compartilhamento de dados no Amazon Redshift.

- Banco de dados padrão: ao ler dados de uma unidade de compartilhamento de dados, sua conexão com o banco de dados do cluster local é mantida. Consulte mais informações sobre configuração e leitura de um banco de dados criado a partir de uma unidade de compartilhamento de dados em Consulta de objetos da unidade de compartilhamento de dados e Visões materializadas em tabelas externas de data lake no Amazon Redshift Spectrum.
- Conexões: você deve ter conexão direta com um banco de dados da unidade de compartilhamento de dados ou executar o comando USE para gravar em unidades de compartilhamento de dados. Você também pode usar a notação de três partes. O comando USE não é permitido em tabelas externas.

- Performance: a performance das consultas em dados compartilhados depende da capacidade computacional dos clusters de consumidor.
- Cobranças de transferência de dados: o compartilhamento de dados entre regiões inclui cobranças adicionais de transferência de dados entre regiões.
  - Essas cobranças de transferência de dados não se aplicam na mesma região, somente entre regiões. Para obter mais informações, consulte Gerenciar o controle de custos para compartilhamento de dados entre regiões.
  - O consumidor é cobrado por todas as taxas de computação e transferência de dados entre regiões necessárias para consultar os dados do produtor. O produtor é cobrado pelo armazenamento subjacente de dados em seu cluster provisionado ou namespace sem servidor.
- Compartilhamento de dados e entre clusters: você só precisa de unidades de compartilhamento de dados quando está compartilhando dados entre diferentes clusters provisionados ou grupos de trabalho sem servidor do Amazon Redshift. No mesmo cluster, você pode consultar outro banco de dados usando uma notação simples de três partes `database.schema.table`, desde que tenha as permissões necessárias nos objetos do outro banco de dados.
- Descoberta de metadados: quando você é um consumidor conectado diretamente a um banco de dados de unidade de compartilhamento de dados por meio dos drivers Redshift JDBC, ODBC ou Python, pode exibir dados de catálogo das seguintes formas:
  - Comandos SQL SHOW.
  - Consulta a tabelas e exibições information_schema.
  - Consulta a exibições de metadados SVV.
- Visibilidade das permissões: os consumidores podem ver as permissões concedidas às unidades de compartilhamento de dados por meio do comando SQL SHOW GRANTS.
- Gerenciamento de criptografia de cluster para compartilhamento de dados: para compartilhar dados em uma Conta da AWS, tanto o cluster produtor quanto o consumidor devem ser criptografados.
  - Se os clusters do produtor e do consumidor e os namespaces sem servidor estiverem na mesma conta, eles deverão ter o mesmo status de criptografia (descriptografado ou criptografado). Em todos os outros casos, incluindo as unidades de compartilhamento de dados gerenciadas pelo Lake Formation, tanto o consumidor quanto o produtor devem ser criptografados. Isso é para fins de segurança. No entanto, eles não precisam compartilhar a mesma chave de criptografia.
  - Para proteger dados em trânsito, todos os dados são criptografados em trânsito por meio do esquema de criptografia do cluster de produtor. O cluster de consumidor adota esse esquema de criptografia quando os dados são carregados. Em seguida, o cluster de consumidor opera

como um cluster criptografado normal. As comunicações entre o produtor e o consumidor também são criptografadas usando um esquema de chave compartilhada. Para obter mais informações sobre a criptografia em trânsito, consulte Criptografia em trânsito.

Considerações sobre leituras e gravações de compartilhamento de dados no Amazon Redshift

> **Note**
>
> As gravações em vários warehouses do Amazon Redshift usando compartilhamento de dados só são compatíveis com o patch 186 do Amazon Redshift para clusters provisionados na versão de acompanhamento atual 1.0.78881 ou posterior e para grupos de trabalho do Amazon Redshift sem servidor na versão 1.0.78890 ou posterior.

Veja as seguintes considerações ao trabalhar com leituras e gravações da unidade de compartilhamento de dados no Amazon Redshift:

- Você só pode compartilhar UDFs de SQL por meio de unidades de compartilhamento de dados. UDFs Python e Lambda não são compatíveis.
- Se o banco de dados de produtor tiver um agrupamento específico, use as mesmas configurações de agrupamento para o banco de dados de consumidor.
- O Amazon Redshift não oferece suporte a funções definidas pelo usuário SQL aninhadas em clusters de produtores.
- O Amazon Redshift não oferece suporte ao compartilhamento de tabelas com chaves de classificação intercaladas e exibições que se referem a tabelas com chaves de classificação intercaladas.
- O Amazon Redshift não é compatível com o acesso a objetos de unidade de compartilhamento de dados que tiveram um evento de DDL simultâneo entre a preparação e a execução do acesso.
- O Amazon Redshift não é compatível com o compartilhamento de procedimentos armazenados por meio de unidades de compartilhamento de dados.
- O Amazon Redshift não comporta o compartilhamento de visualizações do sistema de metadados e de tabelas do sistema.
- Tipo de computação: para usar esse recurso, é necessário usar grupos de trabalho sem servidor ou clusters provisionados com tipos de nó RG ou RA3.

- Nível de isolamento: o nível de isolamento do banco de dados deve ser isolamento de snapshot para permitir que outros grupos de trabalho sem servidor e clusters provisionados gravem nele.
- Consultas e transações com várias instruções: no momento, não é possível usar consultas com várias instruções fora de um bloco de transações. Por isso, se você estiver usando um editor de consultas como o dbeaver e tiver várias consultas de gravação, precisará agrupar suas consultas em uma instrução de transação BEGIN...END explícita.

  Quando instruções de vários comandos são usadas fora das transações, se o primeiro comando for uma gravação em um banco de dados de produtor, os comandos de gravação subsequentes na instrução só serão permitidos nesse banco de dados de produtor. Se o primeiro comando for de leitura, os comandos de gravação subsequentes só serão permitidos no banco de dados em uso, se configurado, caso contrário, no banco de dados local. Observe que as gravações em uma transação só são possíveis em um único banco de dados.
- Dimensionamento do consumidor: os clusters de consumidor devem ter pelo menos 64 fatias ou mais para realizar gravações usando compartilhamento de dados.
- Visualizações e visões materializadas: não é possível criar, atualizar ou alterar visualizações ou visões materializadas em um banco de dados de unidade de compartilhamento de dados.
- Segurança: não é possível anexar ou remover políticas de segurança, como nível de coluna (CLS), nível de linha (RLS) e mascaramento dinâmico de dados (DDM), para objetos de unidade de compartilhamento de dados.
- Gerenciabilidade: os warehouses de consumidor não podem adicionar objetos de unidade de compartilhamento de dados ou visualizações que façam referência a objetos de unidade de compartilhamento de dados em outra unidade de compartilhamento de dados. Os consumidores também não podem modificar ou remover uma unidade de compartilhamento de dados existente.
- Operações de truncamento: as gravações da unidade de compartilhamento de dados são compatíveis com truncamentos transacionais para tabelas remotas. Isso é diferente dos truncamentos que você executa localmente em um cluster, que são confirmados de forma automática. Consulte mais informações sobre o comando SQL em TRUNCATE.
- Clonagem: CREATE TABLE com instruções da cláusula LIKE permite a clonagem de uma única tabela principal quando você grava do warehouse de consumidor para o de produtor.

Considerações sobre o compartilhamento de dados na restauração do Amazon Redshift sem servidor

Considere o seguinte ao trabalhar com unidades de compartilhamentos de dados durante operações de restauração no Amazon Redshift sem servidor:

Restaure o snapshot em um namespace de produtor do Amazon Redshift sem servidor:

- A restauração de um snapshot para um namespace de produtor do Amazon Redshift sem servidor substitui as unidades de compartilhamento de dados atuais pelas unidades de compartilhamento de dados no snapshot. Entretanto, uma permissão de unidade de compartilhamento de dados somente será mantida se TODAS as seguintes condições forem atendidas:
  - O snapshot foi tirado no mesmo namespace para o qual o cliente está restaurando E
  - A permissão da unidade de compartilhamento de dados existe no namespace atual E
  - A permissão da unidade de compartilhamento de dados existia no namespace quando o snapshot foi tirado.
- Se uma unidade de compartilhamento de dados tiver sido concedida a um consumidor ao tirar o snapshot, mas tiver sido removida ou a concessão tiver sido revogada no namespace mais recente, ela será restaurada. No entanto, as permissões de unidade de compartilhamento de dados concedidas a outros clusters de consumidor não serão mais válidas após a restauração. Para restaurar o acesso à unidade de compartilhamento de dados, conceda novamente as permissões de uso de unidades de compartilhamento de dados aos clusters de consumidor desejados.
- Se uma unidade de compartilhamento de dados for concedida a um consumidor no namespace mais recente ao tirar o snapshot, o acesso à unidade de compartilhamento de dados não será mantido após a restauração.
- O escopo da autorização/associação da unidade de compartilhamento de dados não mudará no namespace após a restauração. Por exemplo, antes da restauração, se uma unidade de compartilhamento de dados estiver associada à Conta da AWS como um todo, o escopo da associação permanecerá no nível da conta após a restauração, independentemente do escopo da associação ao tirar o instantâneo.
- A restauração por meio de um snapshot reverterá a configuração acessível ao público para o respectivo estado no momento em que o snapshot foi criado, independentemente de quaisquer alterações subsequentes.

Restaure o snapshot em um namespace de consumidor do Amazon Redshift sem servidor:

- A restauração de um namespace de consumidor preserva o acesso à unidade de compartilhamento de dados sem exigir que o administrador do produtor conceda novamente o uso. No entanto, se o banco de dados criado na unidade de compartilhamento de dados não existir mais após a restauração, o consumidor deverá recriá-lo na unidade de compartilhamento de dados.

Considerações sobre compartilhamento de dados com tabelas de data lake no Amazon Redshift

Veja as seguintes considerações ao trabalhar com tabelas de data lake no Amazon Redshift:

- Não é possível usar o compartilhamento de dados de tabelas de data lake com AWS KMS keys gerenciadas pelo cliente para criptografia do bucket do Amazon S3. Você pode usar Chaves gerenciadas pela AWS para criptografia. Para obter mais informações, consulte Usar a criptografia do lado do servidor com chaves gerenciadas pelo Amazon S3 (SSE-S3).
- Para compartilhar dados de tabelas de data lake de um catálogo do AWS Glue criptografado, você deve delegar operações do AWS KMS a um perfil do IAM seguindo as instruções em Como criptografar seu Data Catalog.
- As tabelas externas que especificam explicitamente os arquivos de manifesto na cláusula `LOCATION` não são compatíveis com o compartilhamento de dados. Isso inclui as seguintes tabelas compatíveis com o Amazon Redshift Spectrum:
  - Delta Lake
  - Hudi

Considerações sobre compartilhamento de dados com o AWS Lake Formation no Amazon Redshift

Veja a seguir considerações e limitações sobre o compartilhamento de dados do Amazon Redshift com o Lake Formation. Para obter informações sobre considerações e limitações de compartilhamento de dados, consulte Considerações ao usar o compartilhamento de dados no Amazon Redshift. Para obter informações sobre as limitações do Lake Formation, consulte Observações sobre como trabalhar com unidades de compartilhamento de dados do Amazon Redshift no Lake Formation.

- No momento, não há suporte ao compartilhamento de uma unidade de compartilhamento de dados com o Lake Formation entre regiões.
- Se filtros de coluna forem definidos para um usuário em uma relação compartilhada, a execução de uma operação `SELECT *` retornará somente as colunas às quais o usuário tem acesso.
- Não há suporte para filtros de célula do Lake Formation.
- Se você criou e compartilhou uma visão e suas tabelas com o Lake Formation, pode configurar filtros para gerenciar o acesso às tabelas. O Amazon Redshift aplica políticas definidas pelo Lake Formation quando usuários do cluster consumidor acessam objetos compartilhados. Quando um usuário acessa uma visão compartilhada com o Lake Formation, o Redshift aplica somente as políticas do Lake Formation definidas na visão, ignorando as definidas nas tabelas contidas na visão. No entanto, quando os usuários acessam diretamente a tabela, o Redshift aplica as políticas do Lake Formation definidas na tabela.
- Você não poderá criar visões materializadas no consumidor com base em uma tabela compartilhada se a tabela tiver filtros do Lake Formation configurados.
- O administrador do Lake Formation deve ter as permissões de administrador do data lake e as permissões necessárias para aceitar uma unidade de compartilhamento de dados.
- O cluster produtor-consumidor deve ser um cluster RG ou RA3 com a versão mais recente do cluster do Amazon Redshift ou um grupo de trabalho sem servidor para compartilhar unidades de compartilhamento de dados por meio do Lake Formation.
- Os clusters de produtor e de consumidor devem ser criptografados.
- As políticas de controle de acesso no nível da linha e da coluna do Redshift implementadas no cluster produtor ou no grupo de trabalho são ignoradas quando a unidade de compartilhamento de dados é compartilhada com o Lake Formation. O administrador do Lake Formation deve configurar essas políticas no Lake Formation. O administrador do cluster produtor ou do grupo de trabalho pode desativar o RLS de uma tabela usando o comando ALTER TABLE.
- O compartilhamento de unidades de compartilhamento de dados por meio do Lake Formation está disponível somente para usuários que tenham acesso ao Redshift e ao Lake Formation.

Considerações sobre compartilhamento de dados com o AWS Data Exchange no Amazon Redshift

Ao usar o AWS Data Exchange for Amazon Redshift, considere o seguinte:

- Tanto os produtores como os consumidores devem usar os tipos de instância RG ou RA3 para usar as unidades de compartilhamento de dados do Amazon Redshift. Os produtores devem usar os tipos de instância RG ou RA3 com a versão mais recente do cluster do Amazon Redshift.
- Os clusters de produtor e de consumidor devem ser criptografados.
- Você deve estar registrado como provedor do AWS Data Exchange para listar produtos no AWS Data Exchange, inclusive produtos que contenham unidades de compartilhamento de dados do AWS Data Exchange. Para obter mais informações, consulte Primeiros passos como provedor.
- Você não precisa ser um provedor do AWS Data Exchange registrado para localizar, assinar e consultar dados do Amazon Redshift por meio do AWS Data Exchange.
- Para controlar o acesso a seus dados, crie unidades de compartilhamento de dados do AWS Data Exchange com a configuração publicamente acessível ativada. Para alterar uma unidade de compartilhamento de dados AWS Data Exchange de modo a desativar a configuração publicamente acessível, defina a variável de sessão para permitir ALTER DATASHARE SET PUBLICACCESSIBLE FALSE. Para obter mais informações, consulte Observações de uso do ALTER DATASHARE.
- Os produtores não podem adicionar ou remover consumidores de unidades de compartilhamento de dados do AWS Data Exchange manualmente porque o acesso às unidades de compartilhamento de dados é concedido com base no fato de haver uma assinatura ativa de um produto do AWS Data Exchange que contenha a unidade de compartilhamento de dados do AWS Data Exchange.
- Os produtores não conseguem visualizar as consultas SQL que os consumidores executam. Eles podem visualizar somente metadados, como o número de consultas ou os objetos que os consumidores consultam, por meio de tabelas do Amazon Redshift que somente o produtor pode acessar. Para obter mais informações, consulte Monitoramento e auditoria do compartilhamento de dados no Amazon Redshift.
- Recomendamos tornar suas unidades de compartilhamento de dados acessíveis publicamente. Se você não fizer isso, os assinantes do AWS Data Exchange com clusters de consumidor acessíveis publicamente não poderão usar sua unidade de compartilhamento de dados.
- Recomendamos não excluir uma unidade de compartilhamento de dados do AWS Data Exchange compartilhada com outras Contas da AWS usando a instrução DROP DATASHARE. Se você fizer isso, as Contas da AWS que têm acesso à unidade de compartilhamento de dados perderão o acesso. Essa ação é irreversível. Executar esse tipo de alteração pode violar os termos do produto de dados no AWS Data Exchange. Para excluir uma unidade de compartilhamento de dados do AWS Data Exchange, consulte Observações sobre o uso de DROP DATASHARE.

- Para compartilhamento de dados entre regiões, você pode criar unidades de compartilhamento de dados do AWS Data Exchange para compartilhar dados licenciados.
- Ao consumir dados de uma região diferente, o consumidor paga a taxa de transferência de dados entre regiões, da região produtora para a região consumidora.

Permissões que você pode conceder a unidades de compartilhamento de dados

Tipos de objeto diferentes e permissões variadas que você pode conceder a elas em um contexto de compartilhamento de dados.

Bancos de dados:

- CREATE
- USAGE
- ALTER
- DROP

Esquemas:

- CREATE
- USAGE
- ALTER
- DROP

Tabelas:

- SELECT
- INSERT
- UPDATE
- DELETE
- TRUNCATE
- DROP

- ALTER
- REFERENCES

Funções:

- EXECUTE

Instruções SQL aceitas para gravações de compartilhamento de dados em consumidores

As seguintes instruções de linguagem de definição de dados (DDL) são compatíveis com a unidade de compartilhamento de dados com gravações:

- ALTER TABLE RENAME TO
- ALTER TABLE RENAME COLUMN TO
- ALTER TABLE ADD/DROP COLUMN
- ALTER SCHEMA RENAME TO
- { CREATE | DROP } SCHEMA
- { CREATE | DROP | SHOW } TABLE
- CREATE TABLE table_name AS
- BEGIN | START TRANSACTION
- END | COMMIT | ROLLBACK
- TRUNCATE

As seguintes instruções de linguagem de manipulação de dados (DML) são compatíveis com a unidade de compartilhamento de dados com gravações:

- SELECT
- INSERT
- INSERT INTO SELECT
- UPDATE
- DELETE

- MERGE
- COPY sem COMPUPDATE

As seguintes instruções de análise são compatíveis com a unidade de compartilhamento de dados com gravações:

- ANALYZE. O consumidor executa USER ANALYZE e envia as estatísticas para o produtor.
- Análise ativada pelo CTAS/COPY executado no consumidor. Isso inclui multiplexação para pequenos consumidores.
- Análise automática executada no produtor depois de COPY.

As seguintes instruções de permissão são compatíveis com a unidade de compartilhamento de dados com gravações:

- { GRANT | REVOKE } privilege_name ON OBJECT_TYPE object_name TO consumer_user
- SHOW GRANTS. Para obter mais informações, consulte SHOW GRANTS.

Instruções SQL não aceitas para gravações de compartilhamento de dados em consumidores

As seguintes opções não são compatíveis:

- Consultas com várias instruções a warehouses consumidores durante a gravação em produtores.
- Consultas com várias instruções para warehouses de consumidor em um banco de dados diferente, se o comando anterior for uma instrução de leitura.
- Referências de objeto com notações de um ponto ou de dois pontos, e não com a notação de três pontos, se a conexão não for com um banco de dados compartilhado.
- Gravação de consultas em escala de simultaneidade de consumidores para produtores.
- Gravação de trabalhos de cópia automática de consumidores para produtores.
- Gravação de trabalhos de transmissão de consumidores para produtores.
- Criação de tabelas de integração ETL zero por consumidores em clusters produtores. Para obter informações sobre integrações ETL zero, consulte Working with zero-ETL integrations.
- Gravação em uma tabela com uma chave de classificação intercalada por meio de uma unidade de compartilhamento de dados.

- Gravação em um procedimento armazenado por meio de uma unidade de compartilhamento de dados.
- Gravação em funções definidas pelo usuário (UDFs) SQL por meio de uma unidade de compartilhamento de dados. Isso inclui UDFs aninhadas, do Python e do Lambda.
- Instruções UPDATE, INSERT ou COPY em colunas de identidade para warehouses de consumidor com mais fatias de computação do que o produtor.
- Instruções MERGE em tabelas externas não RMS para warehouses de consumidor, ao gravar nos produtores.
- Instruções CREATE TABLE com:
  - Expressão DEFAULT definida para o tipo de dados VARBYTE. O tipo de dados VARBYTE não pode ser convertido explicitamente em outro tipo de dados. Para obter mais informações, consulte Função CAST.
  - Instruções AS OF SELECT com parâmetro NULL para warehouses de consumidor, ao gravar em produtores.

Regiões da AWS em que o compartilhamento de dados está disponível

A tabela a seguir mostra a disponibilidade de recursos de leitura e gravação de compartilhamento de dados.

Região	Compartilhamento de dados na mesma região	Compartilhamento de dados entre regiões	AWS Lake Formation Unidades de compartilhamento de dados controladas pelo
Leste dos EUA (Norte da Virgínia) (us-east-1)	Sim	Sim	Sim
Leste dos EUA (Ohio) (us-east-2)	Sim	Sim	Sim
Oeste dos EUA (N. da Califórnia) (us-west-1)	Sim	Sim	Sim

Região	Compartilhamento de dados na mesma região	Compartilhamento de dados entre regiões	AWS Lake Formation Unidades de compartilhamento de dados controladas pelo
Oeste dos EUA (Oregon) (us-west-2)	Sim	Sim	Sim
África (Cidade do Cabo) (af-south-1)	Sim	Não	Sim
Ásia-Pacífico (Hong Kong) (ap-east-1)	Sim	Não	Sim
Ásia-Pacífico (Hyderabad) (ap-south-2)	Sim	Não	Sim
Ásia-Pacífico (Jacarta) (ap-southeast-3)	Sim	Não	Sim
Ásia-Pacífico (Malásia) (ap-southeast-5)	Sim	Não	Sim
Ásia-Pacífico (Melbourne) (ap-southeast-4)	Sim	Não	Sim
Ásia-Pacífico (Mumbai) (ap-south-1)	Sim	Sim	Sim
Ásia-Pacífico (Nova Zelândia) (ap-southeast-6)	Sim	Não	Sim
Ásia-Pacífico (Osaka) (ap-northeast-3)	Sim	Sim	Sim

Região	Compartilhamento de dados na mesma região	Compartilhamento de dados entre regiões	AWS Lake Formation Unidades de compartilhamento de dados controladas pelo
Ásia-Pacífico (Seul) (ap-northeast-2)	Sim	Sim	Sim
Ásia-Pacífico (Singapura) (ap-southeast-1)	Sim	Sim	Sim
Ásia-Pacífico (Sydney) (ap-southeast-2)	Sim	Sim	Sim
Ásia-Pacífico (Taipei) (ap-east-2)	Sim	Não	Sim
Ásia-Pacífico (Tailândia) (ap-southeast-7)	Sim	Não	Sim
Ásia-Pacífico (Tóquio) (ap-northeast-1)	Sim	Sim	Sim
Canadá (Central) (ca-central-1)	Sim	Sim	Sim
Oeste do Canadá (Calgary) ca-west-1	Sim	Não	Sim
China (Pequim) (cn-north-1)	Sim	Sim	Sim
China (Ningxia) (cn-northwest-1)	Sim	Sim	Sim
Europa (Frankfurt) (eu-central-1)	Sim	Sim	Sim

Região	Compartilhamento de dados na mesma região	Compartilhamento de dados entre regiões	AWS Lake Formation Unidades de compartilhamento de dados controladas pelo
Europa (Irlanda) (eu-west-1)	Sim	Sim	Sim
Europa (Londres) (eu-west-2)	Sim	Sim	Sim
UE (Milão) (eu-south-1)	Sim	Não	Sim
Europa (Paris) (eu-west-3)	Sim	Sim	Sim
Europa (Espanha) (eu-south-2)	Sim	Não	Sim
Europa (Estocolmo) (eu-north-1)	Sim	Sim	Sim
Europa (Zurique) (eu-central-2)	Sim	Não	Sim
Israel (Tel Aviv) (il-central-1)	Sim	Não	Sim
México (Centro) (mx-central-1)	Sim	Não	Sim
Oriente Médio (Bahrein) (me-south-1)	Sim	Não	Sim
Oriente Médio (EAU) (me-central-1)	Sim	Não	Sim

Região	Compartilhamento de dados na mesma região	Compartilhamento de dados entre regiões	AWS Lake Formation Unidades de compartilhamento de dados controladas pelo
América do Sul (São Paulo) (sa-east-1)	Sim	Sim	Sim
AWS GovCloud (Leste dos EUA) (us-gov-east-1)	Sim	Não	Sim
AWS GovCloud (Oeste dos EUA) (us-gov-wast-1)	Sim	Não	Sim

Conceitos básicos do compartilhamento de dados no Amazon Redshift

Você pode começar a compartilhar dados seguindo um dos guias desta seção.

Tópicos

- Conceitos básicos do compartilhamento de dados somente leitura no console
- Conceitos básicos do compartilhamento de dados somente leitura com a interface SQL
- Conceitos básicos das gravações em vários data warehouses usando o compartilhamento de dados no Amazon Redshift
- Conceitos básicos do compartilhamento de dados por meio do AWS CloudFormation no Amazon Redshift

Conceitos básicos do compartilhamento de dados somente leitura no console

Com o Amazon Redshift, você pode gerenciar o compartilhamento de dados com gravações usando o console para controlar o acesso e supervisionar os dados em clusters do Amazon Redshift e

contas da AWS. As seções a seguir apresentam instruções detalhadas sobre como configurar e gerenciar o compartilhamento de dados usando o console do Amazon Redshift.

Use o console do Amazon Redshift para gerenciar datashares criados em sua conta ou compartilhados de outras contas.

Você precisa de permissões para criar, editar ou excluir unidades de compartilhamento de dados. Para mais informações, consulte Gerenciar permissões para unidades de compartilhamento de dados no Amazon Redshift.

- Se você for um administrador de produtor, poderá criar unidades de compartilhamento de dados, adicionar consumidores de dados, acrescentar objetos de unidades de compartilhamento de dados, criar bancos de dados usando unidades de compartilhamento de dados, editar unidades de compartilhamento de dados ou excluir unidades de compartilhamento de dados na guia CLUSTERS.

  No menu de navegação, navegue pela guia Clusters e escolha um cluster da lista de clusters. Depois, siga um destes procedimentos:

  - Escolha a guia Datashares (Unidades de compartilhamento de dados) e selecione uma unidade de compartilhamento de dados na seção Datashares created in my namespace (Unidades de compartilhamento de dados criadas em meu namespace). Depois, siga um destes procedimentos:
    - Criar uma unidade de compartilhamento de dados

      Quando um datashare é criado, você pode adicionar objetos de datashare ou consumidores de dados. Para obter mais informações, consulte Adicionar objetos de unidade de compartilhamento de dados às unidades de compartilhamento de dados e Adicionar consumidores de dados às unidades de compartilhamento de dados.
    - Editar datashares criados em sua conta
    - Excluir unidades de compartilhamento de dados criadas na conta
  - Selecione Datashares e escolha um datashare na seção Datashares de outros clusters. Depois, siga um destes procedimentos:
    - Criar uma unidade de compartilhamento de dados
    - Criar bancos de dados a partir de datashares
  - Escolha Bancos de dados e escolha um banco de dados na seção Bancos de dados. Em seguida, escolha Criar datashare. Para obter mais informações, consulte Criar bancos de dados a partir de datashares.

> **Note**
>
> Para exibir bancos de dados e objetos em bancos de dados ou para exibir datashares no cluster, conecte-se a um banco de dados. Para obter mais informações, consulte Conectar-se a um banco de dados.

Conectar-se a um banco de dados

Com o Amazon Redshift, você pode estabelecer conexão com o cluster do data warehouse e executar consultas SQL, carregar dados ou realizar tarefas administrativas. A conexão com um banco de dados refere-se ao processo de criar um canal seguro entre uma ferramenta ou aplicação cliente e o cluster do Amazon Redshift. As seções a seguir apresentarão instruções detalhadas sobre como estabelecer conexão com um banco de dados do Amazon Redshift.

Conecte-se a um banco de dados para exibir bancos de dados e objetos dentro de bancos de dados neste cluster ou para exibir datashares.

As credenciais de usuário usadas para se conectar a um banco de dados especificado devem ter as permissões necessárias para exibir todas as unidades de compartilhamento de dados.

Se não houver conexão local, siga um destes procedimentos:

- Na página de detalhes do cluster, na guia Bancos de dados, na seção Bancos de dados ou Objetos de unidade de compartilhamento de dados, escolha Conectar-se ao banco de dados para visualizar objetos de banco de dados no cluster.
- Na página de detalhes do cluster, a partir da guia Datashares, siga um destes procedimentos:
  - Na seção Datashares de outros clusters, escolha Conectar ao banco de dados para exibir datashares de outros clusters.
  - Na seção Datashares criados no meu cluster, escolha Conectar ao banco de dados para exibir os datashares em seu cluster.
- Na janela Conectar ao banco de dados, siga um destes procedimentos:
  - Se escolher Criar uma nova conexão, escolha AWS Secrets Manager para usar um segredo armazenado para autenticar o acesso para a conexão.

    Ou escolha Credenciais temporárias para usar credenciais de banco de dados para autenticar o acesso para a conexão. Especifique valores para o Nome do banco de dados e o Usuário do banco de dados.

Selecione Conectar.

- Escolha Usar uma conexão recente para se conectar a outro banco de dados que você tem as permissões necessárias.

  O Amazon Redshift faz a conexão automaticamente.

Depois de estabelecer uma conexão com o banco de dados, você pode começar a criar unidades de compartilhamento de dados, consultar unidades de compartilhamento de dados ou criar bancos de dados usando unidades de compartilhamento de dados.

Criar datashares

Com o Amazon Redshift, é possível compartilhar dados em tempo real entre clusters do Amazon Redshift ou contas da AWS usando unidades de compartilhamento de dados. Uma unidade de compartilhamento de dados é um objeto consumidor-produtor usado para compartilhar dados em tempo real do cluster do Amazon Redshift com outros clusters ou contas da AWS. Ao criar unidades de compartilhamento de dados, é possível compartilhar dados de maneira segura e, ao mesmo tempo, manter o controle sobre o acesso e garantir que os dados permaneçam atualizados. As seções a seguir apresentam detalhes sobre como criar unidades de compartilhamento de dados e adicionar objetos de banco de dados, como esquemas, tabelas e visualizações, para compartilhar dados em tempo real com segurança.

Criar uma unidade de compartilhamento de dados

Uma unidade de compartilhamento de dados é um contêiner lógico de objetos, permissões e consumidores do banco de dados. Consumidores são clusters provisionados do Amazon Redshift ou namespaces do Amazon Redshift sem servidor em sua conta e em outras Contas da AWS. Cada unidade de compartilhamento de dados é associada ao banco de dados no qual foi criada e somente objetos desse banco de dados podem ser adicionados. Como administrador de produtor, você pode criar unidades de compartilhamento de dados no console e com SQL seguindo um dos procedimentos abaixo.

Console

No console, você pode criar unidades de compartilhamento de dados nas guias Unidades de compartilhamento de dados na página de detalhes do cluster ou namespace. Após a criação da unidade de compartilhamento de dados, você pode criar unidades usando unidades de compartilhamento de dados em um consumidor como administrador do consumidor.

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. No menu de navegação, escolha Clusters e o seu cluster. A página de detalhes do cluster é exibida.
3. Na página de detalhes do cluster ou namespace, na guia Unidades de compartilhamento de dados, na seção Unidades de compartilhamento de dados, conecte-se a um banco de dados se você não tiver uma conexão de banco de dados. Na seção Unidades de compartilhamento de dados criados na minha conta, escolha Criar unidade de compartilhamento de dados. A página Criar compartilhamento de dados é exibida.
4. Escolha Criar datashare. Você só pode criar um datashare a partir de um banco de dados local. Se você ainda não tiver se conectado ao banco de dados, a página Conectar-se ao banco de dados será aberta. Siga as etapas em Conectar-se a um banco de dados para se conectar a um banco de dados. Se houver uma conexão recente, a página Criar compartilhamento de dados será exibida.
5. Na seção Datashare information (Informações da unidade de compartilhamento de dados), escolha uma destas opções:
   - Selecione Unidade de compartilhamento de dados para criar unidades e compartilhar dados para fins de leitura ou gravação entre diferentes data warehouses do Amazon Redshift (clusters provisionados ou endpoints sem servidor) ou na mesma Conta da AWS ou em Contas da AWS diferentes.
   - Selecione Unidade de compartilhamento de dados do AWS Data Exchange para criar unidades de compartilhamento de dados e licenciar dados por meio do AWS Data Exchange.
6. Especifique valores para Datashare name (Nome da unidade de compartilhamento de dados), Database name (Nome do banco de dados) e Publicly accessible (Acessível publicamente). Quando você alterar o nome do banco de dados, faça uma nova conexão de banco de dados.
7. Adicione objetos à unidade de compartilhamento de dados usando as seções Permissões com escopo ou Permissões diretas. Para adicionar objetos a uma unidade de compartilhamento de dados, consulte Criar uma unidade de compartilhamento de dados no Amazon Redshift.
8. Na seção Consumidores de dados, você pode optar por publicar no Amazon Redshift ou no AWS Glue Data Catalog, o que inicia o processo de compartilhamento de dados com o Lake Formation. Publicar a unidade de compartilhamento de dados no Amazon Redshift significa compartilhar os dados com outro namespace ou conta do Amazon Redshift que atua como consumidor.

> **Note**
>
> Depois que a unidade de compartilhamento de dados é criada, não é possível editar a configuração para publicar na outra opção.

9. Escolha Criar datashare.

SQL

O seguinte comando cria uma unidade de compartilhamento de dados:

```
CREATE DATASHARE salesshare;
```

No momento da criação da unidade de compartilhamento de dados, cada unidade de compartilhamento de dados é associada a um banco de dados. Somente objetos desse banco de dados podem ser compartilhados nesse datashare. Vários datashares podem ser criados no mesmo banco de dados com a mesma granularidade ou diferente de objetos. Não há limite para o número de datashares que um cluster pode criar. Também é possível usar o console do Amazon Redshift para criar datashares. Para obter mais informações, consulte CREATE DATASHARE.

Você também pode controlar restrições de segurança à unidade de compartilhamento de dados durante a criação. O exemplo a seguir mostra que o consumidor com acesso IP público tem permissão para ler a unidade de compartilhamento de dados.

```
CREATE DATASHARE my_datashare [PUBLICACCESSIBLE = TRUE];
```

A configuração PUBLICACCESSIBLE = TRUE permite aos consumidores consultar a unidade de compartilhamento de dados em clusters acessíveis ao público e grupos de trabalho provisionados. Deixe isso de fora ou o defina explicitamente como falso caso você não queira permitir.

Você pode modificar propriedades sobre o tipo de consumidor após a criação da unidade de compartilhamento de dados. Por exemplo, você pode definir que os clusters que desejam consumir dados de um dado datashare não podem ser acessíveis publicamente. Consultas de clusters de consumidores que não atendem às restrições de segurança especificadas na unidade de compartilhamento de dados são rejeitadas no tempo de execução da consulta. Para obter mais informações, consulte ALTER DATASHARE.

Adicionar objetos de unidade de compartilhamento de dados às unidades de compartilhamento de dados

Você pode adicionar objetos de banco de dados de vários tipos no console e com SQL seguindo um dos procedimentos abaixo.

Console

Você pode adicionar objetos à unidade de compartilhamento de dados usando as seções Permissões com escopo ou Permissões diretas. Selecione Conceder permissões com escopo ou Conceder permissões diretas para adicionar objetos. Selecione o botão Adicionar para adicionar objetos. Uma caixa de diálogo é exibida. Siga estas etapas:

1. Se você selecionar Conceder permissões com escopo, a página Conceder permissões com escopo será exibida, na qual você poderá conceder permissões com escopo em nível de banco de dados ou de esquema. As unidades de compartilhamento de dados que têm permissões com escopo têm as permissões especificadas em todos os objetos atuais e futuros dentro do banco de dados ou esquema. Para mais detalhes, consulte ., Permissões em escopo.

    1. Em seguida, selecione Permissões com escopo definido do banco de dados para conceder permissões com escopo em nível de banco de dados. Quando você concede permissões com escopo, elas se aplicam ao banco de dados atual durante a criação da unidade de compartilhamento de dados. Essas permissões não podem ser concedidas a objetos individuais e são aplicáveis a objetos existentes e novos (esquemas, tabelas, visualizações e UDFs).
    2. Selecione as permissões com escopo definido para esquemas, tabelas, visualizações ou funções definidas pelo usuário. Isso significa que todos os objetos no banco de dados têm as permissões selecionadas concedidas aos consumidores. Selecione Conceder para concluir a concessão de permissões com escopo definido do banco de dados.
    3. Em seguida, selecione Permissões com escopo definido de esquema para conceder permissões com escopo em nível de esquema. Quando você concede permissões com escopo definido de esquema, todos os objetos adicionados ao esquema têm as permissões da unidade de compartilhamento de dados especificadas.
    4. No menu suspenso, selecione o esquema que você deseja adicionar à unidade de compartilhamento de dados. Você pode selecionar apenas um esquema por vez. Em seguida, selecione as permissões diretas que deseja conceder no esquema selecionado.

5. Selecione permissões com escopo definido para os objetos de esquema, como tabelas, visualizações e funções definidas pelo usuário. As permissões são concedidas em todos os objetos correspondentes no esquema. Podem ser objetos existentes ou aqueles adicionados no futuro. Quando aplicada, não é possível remover uma permissão de um objeto sem revogar as permissões com escopo.
6. Selecione Conceder para concluir a concessão de permissões com escopo definido de esquema.

2. Se você selecionar Conceder permissões diretas, a página Conceder permissões diretas será exibida, na qual você poderá conceder permissões diretas em cada nível de objeto, como esquema, tabela, visualização ou função definida pelo usuário. Para conceder permissões diretas, você deve primeiro adicionar os esquemas relevantes à unidade de compartilhamento de dados.

   1. Em seguida, selecione Conceder permissões diretas a esquemas para aplicar permissões diretas em esquemas específicos. Depois, selecione as permissões de esquema para os objetos de esquema, como tabelas, visualizações e funções definidas pelo usuário, e escolha o esquema que deseja adicionar à unidade de compartilhamento de dados. Selecione Conceder para concluir a adição do esquema à unidade de compartilhamento de dados.
   2. Depois de adicionar um esquema à unidade de compartilhamento de dados, você pode prosseguir com a adição de permissões diretas para os objetos do esquema. Selecione Conceder permissões diretas novamente. A página Conceder permissões diretas é exibida. Em seguida, acesse as guias de permissões diretas para objetos de esquema.
   3. Selecione Conceder permissões diretas para tabelas e visualizações para conceder permissões diretas em nível de objeto a esses objetos. Selecione as permissões diretas e os objetos necessários na lista. Use o campo de pesquisa para encontrar objetos de unidade de compartilhamento de dados. Selecione “Conceder” para concluir a adição de tabelas e visualizações à unidade de compartilhamento de dados.
   4. Selecione Conceder permissões diretas a funções definidas pelo usuário para conceder permissões diretas em nível de objeto em funções definidas pelo usuário. Selecione as permissões diretas e os objetos necessários na lista. Use o campo de pesquisa para encontrar objetos de unidade de compartilhamento de dados. Selecione Conceder para concluir a adição de perfis definidos pelo usuário à unidade de compartilhamento de dados.

3. Você também pode escolher se deseja Adicionar objetos futuros. Quando você opta por incluir objetos da unidade de compartilhamento de dados adicionados ao esquema, isso significa que objetos adicionados ao esquema são adicionados automaticamente à unidade de compartilhamento de dados.
4. Escolha Adicionar para concluir a seção e adicionar os objetos. Eles estão listados em Objetos de compartilhamento de dados.
5. Depois de adicionar objetos, você poderá selecionar objetos individuais e editar as permissões. Se você selecionar um esquema, uma caixa de diálogo será exibida perguntando se você gostaria de adicionar Permissões em escopo. Isso faz com que cada objeto existente ou adicionado ao esquema tenha um conjunto pré-selecionado de permissões, indicado para o tipo de objeto. Por exemplo, o administrador pode definir se todas as tabelas adicionadas têm permissões SELECT e UPDATE, por exemplo.
6. Todos os objetos da unidade de compartilhamento de dados estão listados nas seções Permissões com escopo ou Permissões diretas.
7. Na seção Consumidores de dados, você pode adicionar namespaces ou contas da AWS como consumidores da unidade de compartilhamento de dados.
8. Escolha Criar compartilhamento de dados para salvar as alterações.

Depois que você cria a unidade de compartilhamento de dados, ela é exibida na lista em Unidades de compartilhamento de dados criadas em meu namespace. Se escolher uma unidade de compartilhamento de dados na lista, você poderá exibir os consumidores, os objetos e outras propriedades.

SQL

Com SQL, o proprietário da unidade de compartilhamento de dados deve conceder USAGE nos esquemas que deseja adicionar à unidade de compartilhamento de dados. GRANT é usado para conceder várias ações no esquema, incluindo CREATE e USAGE. Os esquemas mantêm objetos compartilhados:

```
CREATE SCHEMA myshared_schema1;
CREATE SCHEMA myshared_schema2;

GRANT USAGE ON SCHEMA myshared_schema1 TO DATASHARE my_datashare;
GRANT CREATE, USAGE ON SCHEMA myshared_schema2 TO DATASHARE my_datashare;
```

Como alternativa, o administrador pode continuar executando comandos ALTER para adicionar um esquema à unidade de compartilhamento de dados. Somente permissões USAGE são concedidas quando um esquema é adicionado assim.

```
ALTER DATASHARE my_datashare ADD SCHEMA myshared_schema1;
```

Depois de adicionar esquemas, o administrador poderá conceder permissões da unidade de compartilhamento de dados em objetos no esquema. Elas podem ser permissões de leitura e gravação: O exemplo GRANT ALL mostra como conceder todas as permissões.

```
GRANT SELECT, INSERT ON TABLE myshared_schema1.table1, myshared_schema1.table2,
 myshared_schema2.table1
TO DATASHARE my_datashare;

GRANT ALL ON TABLE myshared_schema1.table4 TO DATASHARE my_datashare;
```

Você pode continuar executando comandos como ALTER DATASHARE para adicionar tabelas. Quando se faz isso, somente permissões SELECT são concedidas em objetos adicionados.

```
ALTER DATASHARE my_datashare ADD TABLE myshared_schema1.table1,
 myshared_schema1.table2, myshared_schema2.table1;
```

Você pode conceder permissões com escopo definido a uma unidade de compartilhamento de dados em todos os objetos de um tipo dentro de um banco de dados ou esquema. As unidades de compartilhamento de dados que têm permissões com escopo têm as permissões especificadas em todos os objetos atuais e futuros dentro do banco de dados ou esquema.

Você pode visualizar o escopo das permissões com escopo em nível de banco de dados em SVV_DATABASE_PRIVILEGES. Você pode visualizar o escopo das permissões com escopo em nível de esquema em SVV_SCHEMA_PRIVILEGES.

A seguir está a sintaxe para conceder permissões com escopo definido para unidades de compartilhamento de dados. Para ter mais informações sobre permissões com escopo definido, consulte Permissões em escopo.

```
GRANT { CREATE | USAGE | ALTER | DROP } [,...] | ALL [ PRIVILEGES ] }FOR SCHEMAS IN
DATABASE db_name
TO DATASHARE { datashare_name}
```

```
GRANT { { SELECT | INSERT | UPDATE | DELETE | DROP | ALTER | TRUNCATE | REFERENCES }
 [, ...] } | ALL [PRIVILEGES] } }FOR TABLES IN
{SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
TO DATASHARE { datashare_name}

GRANT { EXECUTE | ALL [ PRIVILEGES ] }FOR FUNCTIONS IN
{SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
TO DATASHARE { datashare_name}
```

Adicionar consumidores de dados às unidades de compartilhamento de dados

Você pode adicionar um ou mais consumidores de dados aos datashares. Os consumidores de dados podem ser namespaces que identificaram exclusivamente clusters do Amazon Redshift ou Contas da AWS.

Você deve optar explicitamente por desativar ou ativar o compartilhamento de sua unidade de compartilhamento de dados para clusters com acesso público.

- Selecione Adicionar namespaces à unidade de compartilhamento de dados. Os namespaces são um identificador exclusivo global (GUID) do cluster do Amazon Redshift.
- Escolha Add Contas da AWS (Adicionar) à unidade de compartilhamento de dados. As Contas da AWS especificadas devem ter permissões de acesso à unidade de compartilhamento de dados.

Autorizar ou remover autorização de unidades de compartilhamento de dados (pré-visualização)

Como administrador de produtor, escolha os consumidores de dados aos quais conceder autorização para acessar as unidades de compartilhamento de dados ou remover a autorização. Consumidores autorizados de dados recebem notificações para tomar ações em datashares. Se você estiver adicionando um namespace como um consumidor de dados, não precisará executar a autorização.

Pré-requisito: para autorizar ou remover a autorização para o datashare, deve haver pelo menos um consumidor de dados adicionado ao datashare.

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.

2. No menu de navegação, escolha Datashares (Unidades de compartilhamento de dados). A página de lista de datashares é exibida.
3. Escolha Na minha conta.
4. Na seção Datashares na minha conta, execute uma das seguintes ações:
   - Escolha um ou mais clusters consumidores que deseja autorizar. A página Autorizar consumidores de dados é exibida. Em seguida, escolha Authorize (Autorizar).

     Se você selecionou Publicar no AWS Glue Data Catalog ao criar a unidade de compartilhamento de dados, só poderá conceder autorização para a unidade de compartilhamento de dados a uma conta do Lake Formation.

     Para a unidade de compartilhamento de dados AWS Data Exchange, só é possível autorizar uma unidade de compartilhamento de dados de cada vez.

     Ao autorizar uma unidade de compartilhamento de dados do AWS Data Exchange, você está compartilhando a unidade de compartilhamento de dados com o serviço AWS Data Exchange e permitindo que o AWS Data Exchange gerencie o acesso à unidade de compartilhamento de dados em seu nome. O AWS Data Exchange permite o acesso aos consumidores adicionando contas de consumidores como consumidores de dados à unidade de compartilhamento de dados do AWS Data Exchange quando eles assinam os produtos. O AWS Data Exchange não tem acesso de leitura à unidade de compartilhamento de dados.
   - Escolha um ou mais clusters consumidores dos quais deseja remover a autorização. Em seguida, escolha Remover autorização.

Depois que os consumidores de dados são autorizados, eles podem acessar objetos de datashare e criar um banco de dados do consumidor para consultar os dados.

Após a remoção da autorização, os consumidores de dados perdem o acesso à unidade de compartilhamento de dados imediatamente.

Gerenciamento de unidades de compartilhamento de dados de outras contas como consumidor

Remover a associação da unidade de compartilhamento de dados dos consumidores de dados

Como administrador de consumidor, você pode remover a associação de unidades de compartilhamento de dados de consumidores de dados.

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. No menu de navegação, escolha Datashares (Unidades de compartilhamento de dados). A página de lista de datashares é exibida.
3. Selecione De outras contas.
4. Na seção Conjuntos de dados de outras contas, escolha o datashare para remover a associação dos consumidores de dados.
5. Na seção Consumidores de dados, escolha um ou mais consumidores de dados para remover associação. Em seguida, escolha Remover associação.
6. Quando a página Remover associação aparecer, escolha Remover associação.

Após a remoção da associação, os consumidores de dados perderão o acesso ao datashare. Você pode fazer alterações na associação de consumidores de dados a qualquer momento.

Recusar unidades de compartilhamento de dados

Como administrador de consumidor, você pode rejeitar a unidade de compartilhamento de dados cujo estado esteja disponível ou ativo. Depois que você rejeitar uma unidade de compartilhamento de dados, os usuários do cluster consumidor perderão acesso a ela. O Amazon Redshift não retornará a unidade de compartilhamento de dados rejeitada se você chamar a operação de API `DescribeDataSharesForConsumer`. Se o administrador do produtor executar a operação de API `DescribeDataSharesForProducer`, ele verá que a unidade de compartilhamento de dados foi rejeitada. Depois que a unidade de compartilhamento de dados é rejeitada, o administrador do produtor pode autorizar a unidade de compartilhamento de dados para um cluster de consumidor novamente, e o administrador do consumidor pode escolher associar a conta da AWS à unidade de compartilhamento de dados ou rejeitá-la.

Se sua conta da AWS tiver uma associação a uma unidade de compartilhamento de dados e uma associação pendente a uma unidade de compartilhamento de dados gerenciada pelo Lake Formation, rejeitar a associação à unidade de compartilhamento de dados gerenciada pelo Lake Formation também rejeitará a unidade de compartilhamento de dados original. Para rejeitar uma associação específica, o administrador do produtor pode remover a autorização de uma unidade de compartilhamento de dados especificada. Essa ação não afeta outras unidades de compartilhamento de dados.

Para rejeitar uma unidade de compartilhamento de dados, use o console da AWS, a operação de API `RejectDataShare` ou `reject-datashare` na AWS CLI.

Para rejeitar uma unidade de compartilhamento de dados usando o console da AWS:

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. No menu de navegação, escolha Unidades de compartilhamento de dados.
3. Selecione De outras contas.
4. Na seção Datashares de outras contas, escolha o datashare que deseja recusar. Quando a página Decline datashare (Recusar unidade de compartilhamento de dados) for exibida, escolha Decline (Recusar).

Após recusar os datashares, você não pode reverter a alteração. O Amazon Redshift remove as unidades de compartilhamento de dados da lista. Para ver a unidade de compartilhamento de dados novamente, o administrador do produtor deve autorizar novamente.

Gerenciamento de unidades de compartilhamento de dados existentes

Com o Amazon Redshift, é possível gerenciar unidades de compartilhamento de dados existentes para controlar o acesso aos dados em um cluster do Amazon Redshift. As seções a seguir apresentam instruções detalhadas sobre como gerenciar unidades de compartilhamento de dados em seu ambiente do Amazon Redshift.

Visualizar datashares

Visualize os datashares na guia DATASHARES ou CLUSTERS.

- Use a guia DATASHAERS para listar os datashares na sua conta ou de outras contas.
  - Para visualizar os datashares criados em sua conta, escolha Na minha conta e escolha o datashare que deseja visualizar.
  - Para exibir os datashares compartilhados de outras contas, escolha De outras contas e escolha o datashare que deseja visualizar.
- Use a guia CLUSTERS para listar os datashares em seu cluster ou de outros clusters.

  Conectar a um banco de dados Para obter mais informações, consulte Conectar-se a um banco de dados.

  Em seguida, escolha um datashare na seção Datashares de outros clusters ou Datashares criados no meu cluster para visualizar seus detalhes.

Remover objetos do datashare de datashares

Você pode remover um ou mais objetos de uma unidade de compartilhamento de dados usando o procedimento a seguir.

Para remover um ou mais objetos de uma unidade de compartilhamento de dados.

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. No menu de navegação, escolha Clusters e o seu cluster. A página de detalhes do cluster é exibida.
3. Selecione Datashares.
4. Na seção Datashares criados na minha conta, escolha Conectar ao banco de dados. Para obter mais informações, consulte Conectar-se a um banco de dados.
5. Escolha o pipeline que você deseja editar e escolha Editar. A página de detalhes do datashare é exibida.
6. Para remover um ou mais objetos do datashare para o datashare, siga um destes procedimentos:

   - Para remover esquemas do datashare, escolha um ou mais esquemas. Em seguida, escolha Remover. O Amazon Redshift remove os esquemas especificados e todos os objetos dos esquemas especificados do datashare.
   - Para remover tabelas e visualizações do datashare, escolha uma ou mais tabelas e visualizações. Em seguida, escolha Remover. Ou escolha Remover por esquema para remover todas as tabelas e visualizações nos esquemas especificados.
   - Para remover funções definidas pelo usuário do datashare, escolha uma ou mais funções definidas pelo usuário. Em seguida, escolha Remover. Ou escolha Remover por esquema para remover todas as funções definidas pelo usuário nos esquemas especificados.

Remover consumidores de dados de datashares

Você pode remover um ou mais consumidores de dados de um datashare. Os consumidores de dados podem ser namespaces que identificaram exclusivamente clusters do Amazon Redshift ou contas da AWS.

Escolha um ou mais consumidores de dados dos IDs de namespace ou da conta da AWS e selecione Remover.

O Amazon Redshift remove os consumidores de dados especificados do datashare. Eles perderão imediatamente o acesso ao datashare.

Editar datashares criados em sua conta

Edite os datashares criados em sua conta usando o console. Conecte-se a um banco de dados primeiro para ver a lista de datashares criados em sua conta.

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. No menu de navegação, escolha Clusters e o seu cluster. A página de detalhes do cluster é exibida.
3. Selecione Datashares.
4. Na seção Datashares criados na minha conta, escolha Conectar ao banco de dados. Para obter mais informações, consulte Conectar-se a um banco de dados.
5. Escolha o pipeline que você deseja editar e escolha Editar. A página de detalhes do datashare é exibida.
6. Efetue quaisquer alterações na seção Objetos do datashare ou Consumidores de dados.

> **Note**
>
> Se você optar por publicar a unidade de compartilhamento de dados no AWS Glue Data Catalog, não poderá editar a configuração de publicação da unidade de compartilhamento de dados para outras contas do Amazon Redshift.

7. Escolha Salvar alterações.

O Amazon Redshift atualiza seu datashare com as alterações.

Excluir unidades de compartilhamento de dados criadas na conta

Exclua os datashares criados em sua conta usando o console. Conecte-se a um banco de dados primeiro para ver a lista de datashares criados em sua conta.

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. No menu de navegação, escolha Clusters e o seu cluster. A página de detalhes do cluster é exibida.

3. Selecione Datashares. A lista de datashare é exibida.
4. Na seção Datashares criados na minha conta, escolha Conectar ao banco de dados. Para obter mais informações, consulte Conectar-se a um banco de dados.
5. Escolha um ou mais datashares que você deseja excluir e escolha Excluir. A página Excluir datashares será exibida.

   A exclusão de uma unidade de compartilhamento de dados compartilhada com o Lake Formation não remove automaticamente as permissões associadas no Lake Formation. Para removê-las, acesse o console do Lake Formation.
6. Digite Excluir para confirmar a exclusão dos datashares especificados.
7. Escolha Excluir.

Depois que os datashares são excluídos, os consumidores de datashare perdem acesso aos conjuntos de dados.

Consulta a unidades de compartilhamento de dados

Com o Amazon Redshift, é possível consultar dados entre unidades de compartilhamento de dados de clusters do produtor para acessar com segurança dados em tempo real sem os copiar ou transferir. As seções a seguir apresentam os detalhes da configuração e consulta de unidades de compartilhamento de dados em seu ambiente do Amazon Redshift.

Criar bancos de dados a partir de datashares

Para iniciar a consulta de dados no datashare, crie um banco de dados a partir de um datashare. Você pode criar apenas um banco de dados a partir de um datashare especificado.

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. No menu de navegação, escolha Clusters e o seu cluster. A página de detalhes do cluster é exibida.
3. Selecione Datashares. A lista de datashare é exibida.
4. Na seção Conjuntos de dados de outros clusters, selecione Conectar ao banco de dados. Para obter mais informações, consulte Conectar-se a um banco de dados.
5. Escolha um datashare do qual você deseja criar bancos de dados e escolha Criar banco de dados a partir do datashare. A página Criar banco de dados a partir do datashare é exibida.

6. Em Nome do banco de dados, especifique um nome do banco de dados. O nome do banco de dados deve conter de 1 a 64 caracteres alfanuméricos (somente minúsculas) e não pode ser uma palavra reservada.
7. Escolha Criar.

Depois que o banco de dados é criado, você pode consultar dados no banco de dados.

Gerenciamento de unidades de compartilhamento de dados do AWS Data Exchange

Com o Amazon Redshift, é possível compartilhar e receber dados em tempo real do AWS Data Exchange de maneira segura, sem precisar criar e gerenciar extrações ou pipelines de dados. Ao gerenciar unidades de compartilhamento de dados do AWS Data Exchange, é possível assinar produtos de dados de terceiros e integrar fluxos de dados em tempo real diretamente ao data warehouse do Amazon Redshift. As seções a seguir demonstram o gerenciamento de unidades de compartilhamento de dados do AWS Data Exchange em clusters do Amazon Redshift.

Criar conjuntos de dados no AWS Data Exchange

Crie um conjunto de dados no AWS Data Exchange.

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. No menu de navegação, escolha Clusters e o seu cluster. A página de detalhes do cluster é exibida.
3. Selecione Datashares.
4. Na seção Datashares created in my account (Unidades de compartilhamento de dados criadas em minha conta), escolha uma unidade de compartilhamento de dados do AWS Data Exchange.
5. Selecione Criar conjunto de dados no AWS Data Exchange. Para obter mais informações, consulte Publicar um novo produto.

Editar unidades de compartilhamento de dados AWS Data Exchange

Edite unidades de compartilhamento de dados do AWS Data Exchange usando o console. Conecte-se a um banco de dados primeiro para ver a lista de datashares criados em sua conta.

Para unidades de compartilhamento de dados do AWS Data Exchange, não é possível fazer alterações nos consumidores de dados.

Para editar a configuração publicamente acessível de unidades de compartilhamento de dados do AWS Data Exchange, use o editor de consultas v2. O Amazon Redshift gera um valor aleatório único para definir a variável de sessão para permitir a desativação dessa configuração. Para obter mais informações, consulte Observações de uso do ALTER DATASHARE.

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. No menu de navegação, escolha Clusters e o seu cluster. A página de detalhes do cluster é exibida.
3. No menu do navegador, selecione Editor e Query editor v2 (Editor de consultas v2).
4. Se esta for a primeira vez que você estiver usando o editor de consultas v2, configure a Conta da AWS. Por padrão, uma chave de propriedade da AWS é usada para criptografar recursos. Para obter mais informações sobre como configurar sua Conta da AWS, consulte "Configurar sua Conta da AWS" no Guia de gerenciamento de clusters do Amazon Redshift.
5. Para se conectar ao cluster no qual está sua unidade de compartilhamento de dados do AWS Data Exchange, escolha Database (Banco de dados) e o nome do cluster no painel de exibição em árvore. Caso seja solicitado, insira os parâmetros de conexão.
6. Copie a seguinte instrução SQL. O exemplo a seguir altera a configuração publicamente acessível da unidade de compartilhamento de dados salesshare.

```
ALTER DATASHARE salesshare SET PUBLICACCESSIBLE FALSE;
```

7. Para executar a instrução SQL copiada, escolha Queries (Consultas) e cole a instrução SQL copiada na área de consulta. Em seguida, escolha Run (Executar).

   É exibido este erro:

```
ALTER DATASHARE salesshare SET PUBLICACCESSIBLE FALSE;
ERROR:  Alter of ADX-managed datashare salesshare requires session variable
 datashare_break_glass_session_var to be set to value 'c670ba4db22f4b'
```

   O valor "c670ba4db22f4b" é um valor único aleatório que o Amazon Redshift gera quando ocorre uma operação não recomendada.
8. Copie e cole o exemplo de instrução a seguir na área de consulta. Então execute o comando. O comando `SET datashare_break_glass_session_var` aplica uma permissão para permitir uma operação não recomendada para uma unidade de compartilhamento de dados do AWS Data Exchange.

```
SET datashare_break_glass_session_var to 'c670ba4db22f4b';
```

9. Execute a instrução ALTER DATASHARE novamente.

```
ALTER DATASHARE salesshare;
```

O Amazon Redshift atualiza seu datashare com as alterações.

Excluir unidades de compartilhamento de dados AWS Data Exchange criadas em sua conta

Exclua as unidades de compartilhamento de dados do AWS Data Exchange criadas em sua conta usando o console. Conecte-se a um banco de dados primeiro para ver a lista de datashares criados em sua conta.

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. No menu de navegação, escolha Clusters e o seu cluster. A página de detalhes do cluster é exibida.
3. No menu do navegador, selecione Editor e Query editor v2 (Editor de consultas v2).
4. Se esta for a primeira vez que você estiver usando o editor de consultas v2, configure a Conta da AWS. Por padrão, uma chave de propriedade da AWS é usada para criptografar recursos. Para obter mais informações sobre como configurar sua Conta da AWS, consulte “Configurar sua Conta da AWS” no Guia de gerenciamento de clusters do Amazon Redshift.
5. Para se conectar ao cluster no qual está sua unidade de compartilhamento de dados do AWS Data Exchange, escolha Database (Banco de dados) e o nome do cluster no painel de exibição em árvore. Caso seja solicitado, insira os parâmetros de conexão.
6. Copie a seguinte instrução SQL. O exemplo a seguir descarta a unidade de compartilhamento de dados salesshare.

```
DROP DATASHARE salesshare
```

7. Para executar a instrução SQL copiada, escolha Queries (Consultas) e cole a instrução SQL copiada na área de consulta. Em seguida, escolha Run (Executar).

   É exibido este erro:

```
ERROR:  Drop of ADX-managed datashare salesshare requires session variable
 datashare_break_glass_session_var to be set to value '620c871f890c49'
```

O valor “620c871f890c49” é um valor único aleatório que o Amazon Redshift gera quando ocorre uma operação não recomendada.

8. Copie e cole o exemplo de instrução a seguir na área de consulta. Então execute o comando. O comando `SET datashare_break_glass_session_var` aplica uma permissão para permitir uma operação não recomendada para uma unidade de compartilhamento de dados do AWS Data Exchange.

```
SET datashare_break_glass_session_var to '620c871f890c49';
```

9. Execute a instrução DROP DATASHARE novamente.

```
DROP DATASHARE salesshare;
```

Depois que a unidade de compartilhamento de dados é excluída, os consumidores da unidade de compartilhamento de dados perdem acesso a ela.

Excluir uma unidade de compartilhamento de dados do AWS Data Exchange compartilhada pode violar os termos do produto de dados do AWS Data Exchange.

Conceitos básicos do compartilhamento de dados somente leitura com a interface SQL

Com o Amazon Redshift, é possível compartilhar dados em tempo real com segurança entre clusters do Amazon Redshift, o que permite que os consumidores de dados consultem e acessem os dados em tempo real sem a necessidade de copiá-los ou replicá-los. Com o compartilhamento de dados, é possível criar e configurar unidades de compartilhamento de dados, que são objetos no lado do produtor que fazem referência aos objetos do banco de dados que você deseja compartilhar.

Você pode compartilhar dados para fins de leitura entre diferentes clusters do Amazon Redshift dentro ou entre Contas da AWS, ou entre Regiões da AWS.

Tópicos

- Compartilhar o acesso de leitura aos dados em uma Conta da AWS

- Trabalhar com visualizações no compartilhamento de dados do Amazon Redshift
- Adicionar tabelas de data lake a uma unidade de compartilhamento de dados
- Compartilhamento de dados entre Contas da AWS
- Compartilhamento de dados entre Regiões da AWS
- Compartilhar dados licenciados do Amazon Redshift no AWS Data Exchange
- Conceitos básicos das unidades de compartilhamento de dados gerenciadas pelo AWS Lake Formation

Compartilhar o acesso de leitura aos dados em uma Conta da AWS

Com o Amazon Redshift, você pode compartilhar o acesso de leitura aos dados entre diferentes usuários ou grupos do banco de dados na mesma conta da AWS. Com esse recurso, é possível controlar os privilégios de acesso aos dados em um nível granular, garantindo que somente usuários ou grupos autorizados possam ler conjuntos de dados específicos.

Compartilhar dados para fins de leitura como administrador de produtor ou proprietário de banco de dados

1. Crie datashares no seu cluster. Para obter mais informações, consulte CREATE DATASHARE.

   ```
   CREATE DATASHARE salesshare;
   ```

   Superusuário de cluster e proprietários de banco de dados podem criar conjuntos de dados. Cada datashare é associado a um banco de dados durante a criação. Somente objetos desse banco de dados podem ser compartilhados nesse datashare. Vários datashares podem ser criados no mesmo banco de dados com a mesma granularidade ou diferente de objetos. Não há limite para o número de datashares que um cluster pode criar.

   Também é possível usar o console do Amazon Redshift para criar datashares. Para obter mais informações, consulte Criar uma unidade de compartilhamento de dados.
2. Delegue permissões para operar no datashare. Para ter mais informações, consulte GRANT ou REVOKE.

   O exemplo a seguir concede permissões para `dbuser` em `salesshare`.

   ```
   GRANT ALTER, SHARE ON DATASHARE salesshare TO dbuser;
   ```

Os superusuários do cluster e os proprietários da unidade de compartilhamento de dados podem conceder ou revogar permissões de modificação na unidade de compartilhamento de dados para usuários adicionais.

3. Adicione ou remova objetos do datashare. Para adicionar objetos a um datashare, adicione o esquema antes de adicionar objetos. Quando você adiciona um esquema, o Amazon Redshift não adiciona todos os objetos abaixo dele. Adicione esses dados explicitamente. Para obter mais informações, consulte ALTER DATASHARE.

```
ALTER DATASHARE salesshare ADD SCHEMA PUBLIC;
ALTER DATASHARE salesshare ADD TABLE public.tickit_sales_redshift;
ALTER DATASHARE salesshare ADD ALL TABLES IN SCHEMA PUBLIC;
```

Você também pode adicionar visualizações a um datashare.

```
CREATE VIEW public.sales_data_summary_view AS SELECT * FROM
 public.tickit_sales_redshift;
ALTER DATASHARE salesshare ADD VIEW public.sales_data_summary_view;
```

Use ALTER DATASHARE para compartilhar esquemas, tabelas, visualizações e funções em um determinado esquema. Os superusuários, os proprietários da unidade de compartilhamento de dados ou os usuários que têm a permissão ALTER ou ALL na unidade de compartilhamento de dados podem alterar a unidade de compartilhamento de dados para adicionar objetos ou remover objetos. Os usuários devem ter permissões para adicionar ou remover objetos da unidade de compartilhamento de dados. Os usuários também devem ser os proprietários dos objetos ou ter permissões SELECT, USAGE ou ALL nos objetos.

Você também pode usar GRANT para adicionar objetos à unidade de compartilhamento de dados. Este exemplo mostra como:

```
GRANT SELECT ON TABLE public.tickit_sales_redshift TO DATASHARE salesshare;
```

Essa sintaxe é funcionalmente equivalente a `ALTER DATASHARE salesshare ADD TABLE public.tickit_sales_redshift;`.

Use a cláusula INCLUDENEW para adicionar novas tabelas, visualizações ou funções definidas pelo usuário (UDFs) do SQL criadas em um esquema especificado ao datashare. Somente superusuários podem alterar essa propriedade para cada par datashare-esquema.

```
ALTER DATASHARE salesshare ADD SCHEMA PUBLIC;
ALTER DATASHARE salesshare SET INCLUDENEW = TRUE FOR SCHEMA PUBLIC;
```

Também é possível usar o console do Amazon Redshift para adicionar ou remover objetos de datashare. Para obter mais informações, consulte Adicionar objetos de unidade de compartilhamento de dados às unidades de compartilhamento de dados, Remover objetos do datashare de datashares e Editar datashares criados em sua conta.

4. Adicione consumidores ou remova consumidores de datashares. O exemplo a seguir adiciona o namespace do consumidor ao `salesshare`. O namespace é o identificador global exclusivo (GUID) do namespace do cluster de consumidor na conta. Para ter mais informações, consulte GRANT ou REVOKE.

```
GRANT USAGE ON DATASHARE salesshare TO NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Você só pode conceder permissões a um consumidor de unidade de compartilhamento de dados em uma instrução GRANT.

Os superusuários de cluster e os proprietários de objetos da unidade de compartilhamento de dados ou os usuários que tenham a permissão SHARE nessa unidade podem adicionar a ou remover consumidores de uma unidade de compartilhamento de dados. Para fazer isso, eles usam GRANT USAGE ou REVOKE USE.

Para localizar o namespace do cluster que você vê no momento, você pode usar o comando SELECT CURRENT_NAMESPACE. Para localizar o namespace de um cluster diferente dentro da mesma Conta da AWS, acesse a página de detalhes do cluster do console do Amazon Redshift. Nessa página, localize o campo de namespace recém-adicionado.

Também é possível usar o console do Amazon Redshift para adicionar ou remover consumidores de dados de unidades de compartilhamento de dados. Para obter mais informações, consulte Adicionar consumidores de dados às unidades de compartilhamento de dados e Remover consumidores de dados de datashares.

5. (Opcional) Adicione restrições de segurança ao datashare. O exemplo a seguir mostra que o cluster de consumidores com um acesso IP público tem permissão para ler o datashare. Para obter mais informações, consulte ALTER DATASHARE.

```
ALTER DATASHARE salesshare SET PUBLICACCESSIBLE = TRUE;
```

Você pode modificar propriedades sobre o tipo de consumidores após a criação de datashare. Por exemplo, você pode definir que os clusters que desejam consumir dados de um dado datashare não podem ser acessíveis publicamente. Consultas de clusters de consumidores que não atendem às restrições de segurança especificadas na unidade de compartilhamento de dados são rejeitadas no tempo de execução da consulta.

Também é possível usar o console do Amazon Redshift para editar datashares. Para obter mais informações, consulte Editar datashares criados em sua conta.

6. Liste os datashares criados no cluster e examine o conteúdo do datashare.

   O exemplo a seguir exibe as informações de uma unidade de compartilhamento de dados chamada `salesshare`.

```
DESC DATASHARE salesshare;

 producer_account  |          producer_namespace          | share_type | share_name
 | object_type |           object_name          |   include_new
-------------------+--------------------------------------+------------
+------------+-------------+----------------------------------+-------------------
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_users_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_venue_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_category_redshift|
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_date_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_event_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_listing_redshift |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_sales_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | schema      | public                         |  t
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | view        | public.sales_data_summary_view |
```

O exemplo a seguir exibe os datashares de saída em um cluster de produtores.

```
SHOW DATASHARES LIKE 'sales%';
```

A saída será semelhante à seguinte.

```
share_name | share_owner  | source_database | consumer_database | share_type |
 createdate      | is_publicaccessible  | share_acl | producer_account |
 producer_namespace
-----------+--------------+-----------------+-------------------+------------
+---------------------+----------------------+-----------+-----------------
+---------------------------------------
salesshare |    100       | dev             |                   |  OUTBOUND
 | 2020-12-09 02:27:08 |          True        |           |   123456789012   |
 13b8833d-17c6-4f16-8fe4-1a018f5ed00d
```

Para obter mais informações, consulte DESC DATASHARE e SHOW DATASHARES.

Você também pode usar SVV_DATASHARES, SVV_DATASHARE_CONSUMERS e SVV_DATASHARE_OBJECTS para exibir os datashares, os objetos dentro do datashare e os consumidores de datashare.

7. Descarte datashares. Para obter mais informações, consulte DROP DATASHARE.

   Você pode excluir os objetos do datashare a qualquer momento usando DROP DATASHARE. Superusuários de cluster e proprietários de datashare podem descartar datashares.

   O exemplo a seguir descarta uma unidade de compartilhamento de dados chamada `salesshare`.

   ```
   DROP DATASHARE salesshare;
   ```

   Também é possível usar o console do Amazon Redshift para excluir datashares. Para obter mais informações, consulte Excluir unidades de compartilhamento de dados criadas na conta.

8. Use ALTER DATASHARE para remover objetos de datashares em qualquer ponto do datashare. Use REVOKE USAGE ON para revogar permissões no datashare para determinados consumidores. Ele revoga as permissões USAGE em objetos dentro de uma unidade de compartilhamento de dados e interrompe instantaneamente o acesso a todos os clusters de

consumidores. Listar conjuntos de dados e as consultas de metadados, como listar bancos de dados e tabelas, não retorna os objetos compartilhados depois que o acesso é revogado.

```
ALTER DATASHARE salesshare REMOVE TABLE public.tickit_sales_redshift;
```

Também é possível usar o console do Amazon Redshift para editar datashares. Para obter mais informações, consulte Editar datashares criados em sua conta.

9. Revogue o acesso ao datashare a partir de namespaces se você não quiser mais compartilhar os dados com os consumidores.

```
REVOKE USAGE ON DATASHARE salesshare FROM NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Também é possível usar o console do Amazon Redshift para editar datashares. Para obter mais informações, consulte Editar datashares criados em sua conta.

Compartilhe dados para fins de leitura como administrador de consumidor

1. Liste os datashares que são disponibilizados para você e visualize o conteúdo de datashares. Para obter mais informações, consulte DESC DATASHARE e SHOW DATASHARES.

   O exemplo a seguir exibe as informações de datashares de entrada de um namespace de produtor especificado. Ao executar o DESC DATASHARE como administrador do consumidor, você deve especificar a opção NAMESPACE para visualizar as unidades de compartilhamento de dados de entrada.

```
DESC DATASHARE salesshare OF NAMESPACE '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';


 producer_account  |          producer_namespace          | share_type | share_name
 | object_type |           object_name           |   include_new
-------------------+--------------------------------------+------------
+------------+-------------+----------------------------------+------------------
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_users_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_venue_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_category_redshift |
```

```
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_date_redshift     |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_event_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_listing_redshift  |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_sales_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | schema      | public                          |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | view        | public.sales_data_summary_view  |
```

Somente superusuários de cluster podem fazer isso. Você também pode usar SVV_DATASHARES para visualizar os datashares e SVV_DATAASHARE_OBJECTS para vsualizar os objetos dentro do datashare.

O exemplo a seguir exibe os datashares de entrada em um cluster de consumidores.

```
SHOW DATASHARES LIKE 'sales%';


 share_name | share_owner | source_database | consumer_database | share_type
 | createdate | is_publicaccessible | share_acl | producer_account |
 producer_namespace
------------+-------------+-----------------+-------------------+------------
+------------+---------------------+-----------+------------------
+--------------------------------------
 salesshare |             |                 |                   | INBOUND
    |            |         t           |           |   123456789012   |
 13b8833d-17c6-4f16-8fe4-1a018f5ed00d
```

2. Como superusuário de banco de dados, você pode criar bancos de dados locais que façam referência às unidades de compartilhamento de dados. Para obter mais informações, consulte CREATE DATABASE.

```
CREATE DATABASE sales_db FROM DATASHARE salesshare OF NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Se você quiser um controle mais granular sobre o acesso aos objetos no banco de dados local, use a cláusula WITH PERMISSIONS ao criar o banco de dados. Com essa cláusula, é possível conceder permissões no nível do objeto para objetos no banco de dados na etapa 4.

```
CREATE DATABASE sales_db WITH PERMISSIONS FROM DATASHARE salesshare OF NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Você pode ver os bancos de dados criados a partir do datashare consultando a visualização SVV_REDSHIFT_DATABASES. Você pode se conectar diretamente a esses bancos de dados ou a um banco de dados local em seu cluster de consumidor e executar uma consulta entre os bancos de dados criados nas unidades de compartilhamento de dados. Você não pode criar um datashare em cima de objetos de banco de dados criados a partir de um datashare existente. No entanto, você pode copiar os dados em uma tabela separada no cluster de consumidores, executar qualquer processamento necessário e, em seguida, compartilhar os novos objetos que foram criados.

Também é possível usar o console do Amazon Redshift para criar bancos de dados a partir de datashares. Para obter mais informações, consulte Criar bancos de dados a partir de datashares.

3. (Opcional) Crie esquemas externos para consultar e atribuir permissões granulares a esquemas específicos no banco de dados do consumidor importado no cluster de consumidores. Para obter mais informações, consulte CREATE EXTERNAL SCHEMA.

```
CREATE EXTERNAL SCHEMA sales_schema FROM REDSHIFT DATABASE 'sales_db' SCHEMA
 'public';
```

4. Conceda permissões em bancos de dados e referências de esquema criadas a partir das unidades de compartilhamento de dados para usuários e funções no cluster consumidor conforme necessário. Para ter mais informações, consulte GRANT ou REVOKE.

```
GRANT USAGE ON DATABASE sales_db TO Bob;
```

```
GRANT USAGE ON SCHEMA sales_schema TO ROLE Analyst_role;
```

Se tiver criado o banco de dados sem WITH PERMISSIONS, você só poderá atribuir permissões em todo o banco de dados criado a partir da unidade de compartilhamento de dados para usuários e funções. Em alguns casos, você precisa de controles refinados em um subconjunto

de objetos de banco de dados criados a partir do datashare. Nesse caso, você pode criar uma referência de esquema externo que aponte para esquemas específicos no datashare (conforme descrito na etapa anterior) e fornecer permissões detalhadas no nível do esquema.

Você também pode criar exibições de vinculação tardia sobre objetos compartilhados e usá-las para atribuir permissões detalhadas. Você também pode considerar que os clusters de produtores criem conjuntos de dados adicionais para você com os detalhes necessários.

Se tiver criado o banco de dados com WITH PERMISSIONS na etapa 2, você deverá atribuir permissões no nível de objeto para objetos no banco de dados compartilhado. Um usuário com apenas a permissão USAGE não poderá acessar nenhum objeto em um banco de dados criado com WITH PERMISSIONS até receber permissões adicionais no nível de objeto.

```
GRANT SELECT ON sales_db.public.tickit_sales_redshift to Bob;
```

5. Consultar dados nos objetos compartilhados nos datashares.

   Usuários e funções com permissões em bancos de dados consumidores e esquemas em clusters consumidores podem explorar e navegar nos metadados de quaisquer objetos compartilhados. Eles também podem explorar e navegar por objetos locais em um cluster de consumidores. Para fazer isso, eles usam drivers JDBC ou ODBC ou visualizações SVV_ALL e SVV_REFISH.

   Os clusters de produtores podem ter muitos esquemas no banco de dados, tabelas e visualizações dentro de cada esquema. Os usuários do lado do consumidor podem ver apenas o subconjunto de objetos que são disponibilizados através do datashare. Esses usuários não podem ver todos os metadados do cluster de produtores. Essa abordagem ajuda a fornecer controle de segurança de metadados detalhados com compartilhamento de dados.

   Você continua se conectando a bancos de dados de cluster locais. Mas agora, você também pode ler dos bancos de dados e esquemas que são criados a partir do datashare usando a notação database.schema.table de três partes. Você pode executar consultas que abrangem todos e todos os bancos de dados que são visíveis para você. Estes podem ser bancos de dados locais no cluster ou bancos de dados criados a partir dos datashares. Os clusters de consumidores não podem se conectar aos bancos de dados criados a partir dos datashares.

   Você pode acessar os dados usando a qualificação completa. Para obter mais informações, consulte Exemplos de consulta entre bancos de dados.

```
SELECT * FROM sales_db.public.tickit_sales_redshift ORDER BY 1,2 LIMIT 5;

 salesid | listid | sellerid | buyerid | eventid | dateid | qtysold | pricepaid |
 commission |      saletime
---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------
       1 |      1 |    36861 |   21191 |    7872 |   1875 |       4 |    728.00 |
   109.20 | 2008-02-18 02:36:48
       2 |      4 |     8117 |   11498 |    4337 |   1983 |       2 |     76.00 |
    11.40 | 2008-06-06 05:00:16
       3 |      5 |     1616 |   17433 |    8647 |   1983 |       2 |    350.00 |
    52.50 | 2008-06-06 08:26:17
       4 |      5 |     1616 |   19715 |    8647 |   1986 |       1 |    175.00 |
    26.25 | 2008-06-09 08:38:52
       5 |      6 |    47402 |   14115 |    8240 |   2069 |       2 |    154.00 |
    23.10 | 2008-08-31 09:17:02
```

Você só pode usar instruções SELECT em objetos compartilhados. No entanto, você pode criar tabelas no cluster de consumidores consultando os dados dos objetos compartilhados em um banco de dados local diferente.

Além das consultas, os consumidores podem criar visualizações em objetos compartilhados. Somente visualizações de vinculação tardia ou visões materializadas são aceitas. O Amazon Redshift não oferece suporte a visualizações regulares em dados compartilhados. As visualizações criadas pelos consumidores podem abranger vários bancos de dados locais ou bancos de dados criados a partir de conjuntos de dados. Para obter mais informações, consulte CREATE VIEW.

```
// Connect to a local cluster database

// Create a view on shared objects and access it.
CREATE VIEW sales_data
AS SELECT *
FROM sales_db.public.tickit_sales_redshift
WITH NO SCHEMA BINDING;

SELECT * FROM sales_data;
```

Trabalhar com visualizações no compartilhamento de dados do Amazon Redshift

Um cluster de produtores pode compartilhar visualizações regulares, de vinculação tardia e materializadas. Ao compartilhar visualizações regulares, de vinculação tardia ou visões materializadas, você não precisa compartilhar as tabelas base. A tabela a seguir mostra como as visualizações são compatíveis com o compartilhamento de dados.

Exibir nome	Essa visualização pode ser adicionada a um datashare?	Um consumidor pode criar essa exibição em objetos do datashare entre clusters?
Visualização regular	Sim	Não
Visualização de vinculação tardia	Sim	Sim
Visão materializada	Sim	Sim, mas somente com uma atualização completa

A consulta a seguir mostra a saída de uma visualização regular que é aceita com compartilhamento de dados. Para obter informações sobre a definição de visualização regular, consulte CREATE VIEW.

```
SELECT * FROM tickit_db.public.myevent_regular_vw
ORDER BY eventid LIMIT 5;

   eventid  |  eventname
  ----------+-------------
     3835   | LeAnn Rimes
     3967   | LeAnn Rimes
     4856   | LeAnn Rimes
     4948   | LeAnn Rimes
     5131   | LeAnn Rimes
```

A consulta a seguir mostra a saída de uma visualização de vinculação tardia que é compatível com compartilhamento de dados. Para obter mais informações sobre a definição de visualização de vinculação tardia, consulte CREATE VIEW.

```
SELECT * FROM tickit_db.public.event_lbv
ORDER BY eventid LIMIT 5;

 eventid | venueid | catid | dateid |          eventname           |      starttime
 --------+---------+-------+--------+------------------------------
+---------------------
     1   |   305   |   8   |  1851  |        Gotterdammerung       | 2008-01-25
 14:30:00
     2   |   306   |   8   |  2114  |           Boris Godunov      | 2008-10-15
 20:00:00
     3   |   302   |   8   |  1935  |              Salome          | 2008-04-19
 14:30:00
     4   |   309   |   8   |  2090  |  La Cenerentola (Cinderella) | 2008-09-21
 14:30:00
     5   |   302   |   8   |  1982  |          Il Trovatore        | 2008-06-05
 19:00:00
```

A consulta a seguir mostra a saída de uma visão materializada que é aceita com o compartilhamento de dados. Para obter mais informações sobre a definição de visão materializada, consulte CREATE MATERIALIZED VIEW.

```
SELECT * FROM tickit_db.public.tickets_mv;

  catgroup | qtysold
 ----------+---------
  Concerts | 195444
    Shows  | 149905
```

Você pode manter tabelas comuns em todos os locatários em um cluster de produtores. Você também pode compartilhar subconjuntos de dados filtrados por colunas de dimensão, como `tenant_id` (`account_id` ou `namespace_id`), para clusters de consumidores. Para fazer isso, você pode definir uma exibição na tabela base com um filtro nessas colunas de ID, por exemplo `current_aws_account = tenant_id`. No lado do consumidor, quando você consulta a exibição, você verá apenas as linhas que se qualificam para sua conta. Para fazer isso, você pode usar as funções de contexto do Amazon Redshift `current_aws_account` e `current_namespace`.

A consulta a seguir retorna o ID da conta em que reside o cluster atual do Amazon Redshift. Você pode executar essa consulta se estiver conectado ao Amazon Redshift.

```
select current_user, current_aws_account;
```

```
current_user | current_aws_account
-------------+--------------------
dwuser       |    111111111111
(1row)
```

A consulta a seguir retorna o namespace do atual cluster do Amazon Redshift. Você pode executar essa consulta se estiver conectado ao banco de dados.

```
select current_user, current_namespace;

current_user | current_namespace
-------------+--------------------------------------
dwuser       | 86b5169f-01dc-4a6f-9fbb-e2e24359e9a8
(1 row)
```

Atualização incremental para visões materializadas em uma unidade de compartilhamento de dados

O Amazon Redshift comporta a atualização incremental para visões materializadas em uma unidade de compartilhamento de dados do consumidor quando as tabelas base ou visões materializadas são compartilhadas. A atualização incremental é uma operação em que o Amazon Redshift identifica alterações em uma ou mais tabelas base que ocorreram após a atualização anterior e atualiza somente os registros correspondentes na visão materializada. Para obter mais informações sobre este comportamento, consulte CREATE MATERIALIZED VIEW.

Adicionar tabelas de data lake a uma unidade de compartilhamento de dados

Com uma unidade de compartilhamento de dados, um produtor de dados pode compartilhar de maneira segura objetos de banco de dados com alta granularidade, como esquemas e tabelas, com consumidores na mesma conta ou em diferentes contas da AWS. O produtor também pode compartilhar objetos entre regiões. Este tópico descreve como adicionar objetos de um data lake, especificamente do catálogo de dados do AWS Glue, a uma unidade de compartilhamento de dados. Ele aborda dois casos de uso:

- Adicionar uma visualização de vinculação tardia a uma unidade de compartilhamento de dados que faça referência a uma tabela de um data lake: isso é conveniente para o consumidor, pois a configuração preliminar, como a definição de permissões nos dados de origem externa (por exemplo, com o Lake Formation), provavelmente já está concluída. Outra vantagem é que uma visualização adicionada a uma unidade de compartilhamento de dados pode associar tabelas do data lake a tabelas nativas do Redshift.

- Adicionar diretamente uma tabela de um esquema externo a uma unidade de compartilhamento de dados: isso torna os objetos do data lake disponíveis aos consumidores sem a necessidade de adicionar camadas ou lógica. Os consumidores podem consultar a tabela ou associá-las a tabelas no banco de dados do consumidor.

Esses casos se aplicam depois que você faz referência a uma tabela do catálogo de dados da AWS no Redshift usando CREATE EXTERNAL SCHEMA. Qualquer tabela do catálogo de dados da AWS pode ser a fonte.

> **Note**
>
> As tabelas de data lake que você adiciona a uma unidade de compartilhamento de dados podem incluir tabelas registradas no Lake Formation e tabelas do catálogo de dados do AWS Glue.

Criar um esquema e uma tabela externos

Nas seções a seguir, você criará um esquema externo e uma tabela externa para adicioná-los à unidade de compartilhamento de dados. Estas etapas são preliminares. Se já tiver feito isso, ignore esta seção.

1. No produtor, crie um esquema externo que faça referência aos dados do data lake armazenados no Amazon S3. O esquema externo faz referência ao Catálogo de Dados da AWS. O perfil e a região na amostra são exemplos:

```
CREATE EXTERNAL SCHEMA external_schema_name FROM DATA CATALOG
DATABASE 'glue_database_name'
IAM_ROLE 'arn:aws:iam::123456789012:role/sample-role'
REGION 'us-east-1';
```

2. Crie uma tabela de data lake no esquema externo.

```
CREATE EXTERNAL TABLE external_schema_name.sales(
salesid INTEGER,
sellerid INTEGER,
buyerid INTEGER,
saledate DATE,
pricepaid DECIMAL(8,2))
ROW FORMAT delimited
```

```
FIELDS TERMINATED BY '\t'
STORED AS textfile
LOCATION 's3://redshift-downloads/tickit/spectrum/sales/';
```

A amostra inclui o LOCATION. Ela deve estar no formulário `s3://{bucket_name}/{folder}/` em que a pasta está especificada. A pasta deve ter pelo um caractere de extensão. Opcionalmente, é possível incluir subpastas. Para ver outros exemplos de criação de tabelas em um data lake, consulte Exemplos referentes a CREATE EXTERNAL TABLE.

> **Note**
>
> O compartilhamento é possível somente em tabelas em que o perfil do IAM no produtor tem acesso SELECT na tabela.

Adicionar uma visualização de vinculação tardia que faça referência a uma tabela de data lake a uma unidade de compartilhamento de dados

Quando você cria tabelas com base em um esquema externo do catálogo de dados da AWS e deseja adicioná-las a uma unidade de compartilhamento de dados, a maneira mais comum de fazer isso é adicionar uma visualização de vinculação tardia do Redshift que faça referência à tabela que você criou, a qual contém dados do data lake. Este procedimento mostra as etapas:

1. Crie uma visualização de vinculação tardia que faça referência à tabela externa que você criou anteriormente:

   ```
   CREATE VIEW lbv AS
   select * from external_schema_name.sales, other_schema.t1
   WITH NO SCHEMA BINDING;
   ```

2. Adicione o esquema da visualização à unidade de compartilhamento de dados. Esse é o esquema local que contém a visualização de vinculação tardia.

   ```
   ALTER DATASHARE dsx_datashare ADD SCHEMA public;
   ```

3. Adicione o esquema que contém a tabela referida pela visualização de vinculação tardia à unidade de compartilhamento de dados. É necessário adicionar o esquema para qualquer tabela base referida em uma visualização adicionada a uma unidade de compartilhamento de dados, independentemente de o esquema conter objetos de banco de dados locais ou objetos de um

data lake. Observe que você deve adicionar esse esquema antes de adicionar a visualização de vinculação tardia.

```
ALTER DATASHARE dsx_datashare ADD SCHEMA external_schema_name;
ALTER DATASHARE dsx_datashare ADD SCHEMA other_schema;
```

4. Adicione a visualização à unidade de compartilhamento de dados usando um comando SQL. Observe que o nome da tabela inclui o prefixo do esquema.

```
ALTER DATASHARE my_datashare ADD TABLE public.lbv;
```

5. Confirme se a visualização e os esquemas foram adicionados com êxito à unidade de compartilhamento de dados:

```
SELECT * FROM svv_datashare_objects WHERE share_name = 'my_datashare';
```

6. O administrador do consumidor cria um banco de dados por meio da unidade de compartilhamento de dados e, em seguida, permite que os usuários consumidores o utilizem.

Ao concluir as etapas, os usuários consumidores do banco de dados com acesso à visualização da unidade de compartilhamento de dados podem consultar os dados.

Adicionar uma tabela de data lake diretamente a uma unidade de compartilhamento de dados

Adicionar uma tabela em um esquema externo a uma unidade de compartilhamento de dados é semelhante a adicionar uma visualização. Isso funciona bem para um caso em que um consumidor deseja consultar a tabela do data lake em seu estado original ou quer associá-la a tabelas no data warehouse do consumidor. As etapas a seguir mostram como adicionar uma tabela de data lake a uma unidade de compartilhamento de dados usando SQL.

1. Crie um esquema externo e uma tabela externa, conforme descrito na primeira seção deste tópico.
2. Descubra as tabelas existentes no esquema externo para confirmar se a tabela que você criou está disponível:

```
SELECT * FROM svv_external_tables WHERE schemaname = 'external_schema_name';
```

3. Adicione o esquema externo à unidade de compartilhamento de dados.

```
ALTER DATASHARE my_datashare ADD SCHEMA external_schema_name;
```

4. Adicione a tabela externa à unidade de compartilhamento de dados. Observe que o nome da tabela inclui o prefixo do esquema:

```
ALTER DATASHARE my_datashare ADD TABLE external_schema_name.sales;
```

5. Confirme se a tabela foi adicionada com êxito à unidade de compartilhamento de dados:

```
SELECT * FROM svv_datashare_objects WHERE share_name = 'my_datashare';
```

   Para ter instruções mais detalhadas, consulte Compartilhar o acesso de leitura aos dados em uma Conta da AWS.

6. No consumidor, que é o banco de dados que recebe os dados compartilhados, o administrador associa a unidade de compartilhamento de dados a fim de disponibilizar as tabelas compartilhadas para consulta dos usuários. Para ter mais informações sobre como realizar essa etapa, consulte Gerenciamento de unidades de compartilhamento de dados de outras contas como consumidor.

Depois que os administradores concluírem as etapas, os usuários do banco de dados do consumidor podem gravar consultas para recuperar dados da tabela compartilhada e associá-los a outras tabelas do consumidor.

Observações sobre uso para adicionar objetos de data lake a uma unidade de compartilhamento de dados

Há vários itens a serem observados quando você usa tabelas e visualizações de um data lake em uma unidade de compartilhamento de dados:

- Registro em log com o AWS CloudTrail: a conta do produtor de dados pode usar logs do AWS CloudTrail para auditar quando as tabelas de data lake compartilhadas por meio de uma unidade de compartilhamento de dados são acessadas:
  - Uso de dados de log para controlar o acesso aos dados: os logs do CloudTrail registram detalhes sobre quem acessa as tabelas compartilhadas, incluindo produtores e consumidores da unidade de compartilhamento de dados do Redshift. Os identificadores estão disponíveis no campo `ExternalId` dos logs `AssumeRole` do CloudTrail. O proprietário dos dados pode configurar outras limitações de acesso aos dados em uma política do IAM por meio de ações.

Para ter mais informações sobre como definir o acesso aos dados por meio de políticas, consulte Acesso às Contas da AWS de propriedade de terceiros.

- Permissões de segurança e de consumidor: para tabelas registradas do Lake Formation, os recursos do Amazon S3 são protegidos pelo Lake Formation e disponibilizados por meio das credenciais fornecidas pelo Lake Formation.

Considerações sobre faturamento para adicionar objetos de data lake a uma unidade de compartilhamento de dados

O texto abaixo detalha como os custos são atribuídos para armazenamento e verificação de objetos do data lake em uma unidade de compartilhamento de dados:

- Quando um consumidor consulta objetos compartilhados de um data lake, o custo da verificação é cobrado do consumidor.
  - Quando o consumidor é um cluster RA3 ou DC2 provisionado, o Redshift usa o Redshift Spectrum para verificar dados do Amazon S3. Portanto, o custo do Spectrum é faturado na conta do consumidor.
  - Quando o consumidor é um cluster RG provisionado, não há cobrança separada para o Spectrum, pois as consultas de data lake são executadas na computação em cluster, cobrada com os preços de computação por hora de nó do RG.
  - Quando o consumidor é um grupo de trabalho do Amazon Redshift sem servidor, não há cobrança separada pelo Spectrum.
- Os custos de armazenamento e operações do Amazon S3, como a listagem de buckets, são faturados na conta à qual pertence cada bucket do Amazon S3.

Para obter detalhes adicionais sobre o faturamento do Amazon Redshift sem servidor, consulte Faturamento do Amazon Redshift Serverless. Mais informações sobre faturamento e preços estão disponíveis em Preço do Amazon Redshift.

Compartilhamento de dados entre Contas da AWS

Você pode compartilhar dados para fins de leitura em Contas da AWS. O compartilhamento de dados entre Contas da AWS funciona de forma semelhante ao compartilhamento de dados dentro de uma conta. A diferença é que há um handshake bidirecional necessário para compartilhar dados entre Contas da AWS. Os administradores de uma conta de produtor podem autorizar contas de consumidor a acessar conjuntos de dados ou optar por não autorizar qualquer acesso. Para usar

uma unidade de compartilhamento de dados autorizada, um administrador de conta de consumidor pode associar a unidade de compartilhamento de dados. O administrador pode associar a unidade de compartilhamento de dados a uma Conta da AWS completa ou a clusters específicos na conta de consumidor ou recusar a unidade de compartilhamento de dados. Para obter mais informações sobre compartilhamento de dados em uma conta, consulte Compartilhar o acesso de leitura aos dados em uma Conta da AWS.

Uma unidade de compartilhamento de dados pode ter consumidores de dados que são namespaces na mesma conta ou em diferentes Contas da AWS. Você não precisa criar datashares separados para compartilhar dentro de uma conta e compartilhamento entre contas.

Para o compartilhamento de dados entre contas, o cluster de produtores e consumidores devem ser criptografados.

Ao compartilhar dados com as Contas da AWS, os administradores de produtor compartilham com a Conta da AWS como uma entidade. Um administrador de consumidor pode decidir quais namespaces na conta do consumidor terão acesso a uma unidade de compartilhamento de dados.

Tópicos

- ações do administrador de produtor
- Ações de administrador de conta de consumidor
- ações do administrador de consumidor

ações do administrador de produtor

Com o Amazon Redshift, você pode realizar tarefas administrativas em clusters do produtor para gerenciar a ingestão de dados e o processamento de carregamento.

Se você for um administrador de produtor ou proprietário de banco de dados, siga estas etapas:

1. Crie datashares em seu cluster e adicione objetos do datashare aos datashares. Para obter etapas mais detalhadas sobre como criar datashares e adicionar objetos do datashare a datashares, consulte Compartilhar o acesso de leitura aos dados em uma Conta da AWS. Para obter informações sobre CREATE DATASHARE e ALTER DATASHARE, consulte CREATE DATASHARE e ALTER DATASHARE.

   O exemplo a seguir adiciona diferentes objetos da unidade de compartilhamento de dados à unidade de compartilhamento de dados `salesshare`.

```
-- Add schema to datashare
ALTER DATASHARE salesshare ADD SCHEMA PUBLIC;

-- Add table under schema to datashare
ALTER DATASHARE salesshare ADD TABLE public.tickit_sales_redshift;

-- Add view to datashare
ALTER DATASHARE salesshare ADD TABLE public.sales_data_summary_view;

-- Add all existing tables and views under schema to datashare (does not include
 future table)
ALTER DATASHARE salesshare ADD ALL TABLES in schema public;
```

Também é possível usar o console do Amazon Redshift para criar ou editar datashares. Para obter mais informações, consulte Criar uma unidade de compartilhamento de dados e Editar datashares criados em sua conta.

2. Delegue permissões para operar no datashare. Para ter mais informações, consulte GRANT ou REVOKE.

   O exemplo a seguir concede permissões para `dbuser` em `salesshare`.

   ```
   GRANT ALTER, SHARE ON DATASHARE salesshare TO dbuser;
   ```

   Os superusuários do cluster e os proprietários da unidade de compartilhamento de dados podem conceder ou revogar permissões de modificação na unidade de compartilhamento de dados para usuários adicionais.

3. Adicione consumidores ou remova consumidores de datashares. O exemplo a seguir adiciona o ID da Conta da AWS a `salesshare`. Para ter mais informações, consulte GRANT ou REVOKE.

   ```
   GRANT USAGE ON DATASHARE salesshare TO ACCOUNT '123456789012';
   ```

   Você só pode conceder permissões a um consumidor de dados em uma instrução GRANT.

   Os superusuários de cluster e os proprietários de objetos da unidade de compartilhamento de dados ou os usuários que têm permissões SHARE nessa unidade podem adicionar ou remover consumidores de uma unidade de compartilhamento de dados. Para fazer isso, eles usam GRANT USAGE ou REVOKE USE.

Também é possível usar o console do Amazon Redshift para adicionar ou remover consumidores de dados de unidades de compartilhamento de dados. Para obter mais informações, consulte Adicionar consumidores de dados às unidades de compartilhamento de dados e Remover consumidores de dados de datashares.

4. (Opcional) Revogue o acesso à unidade de compartilhamento de dados de Contas da AWS se você não desejar compartilhar mais os dados com os consumidores.

```
REVOKE USAGE ON DATASHARE salesshare FROM ACCOUNT '123456789012';
```

Se você for um administrador de conta de produtor — siga estas etapas:

Depois que você conceder o uso à Conta da AWS, o status da unidade de compartilhamento de dados se torna `pending_authorization`. O administrador da conta do produtor deve autorizar os conjuntos de dados usando o console do Amazon Redshift e escolher os consumidores de dados.

Faça login em https://console.aws.amazon.com/redshiftv2/. Escolha os consumidores de dados cuja autorização de acesso a unidades de compartilhamento de dados será concedida ou removida. Consumidores autorizados de dados recebem notificações para tomar ações em datashares. Se você estiver adicionando um namespace como um consumidor de dados, não precisará executar a autorização. Depois que os consumidores de dados são autorizados, eles podem acessar objetos de datashare e criar um banco de dados do consumidor para consultar os dados. Para obter mais informações, consulte Autorizar ou remover autorização de unidades de compartilhamento de dados (pré-visualização).

Compartilhar permissões de gravação de dados entre contas

Com o Amazon Redshift, é possível compartilhar dados entre contas da AWS e conceder permissões de gravação, o que possibilita a colaboração e o compartilhamento de dados entre equipes ou organizações. Com o compartilhamento de dados entre contas, é possível estabelecer uma conta de provedor de dados que cria e gerencia bancos de dados, esquemas e tabelas, que podem então ser compartilhados com segurança com contas de consumidores de dados. As seções a seguir demonstram o processo de configuração do compartilhamento de dados entre contas e da concessão de acesso de gravação no Amazon Redshift.

Ações de administrador de conta de consumidor

Com o Amazon Redshift, é possível gerenciar contas de consumidores e controlar o acesso aos seus recursos de data warehouse.

Se você for um administrador de conta de consumidor — siga estas etapas:

Para associar uma ou mais unidades de compartilhamento de dados de outras contas com toda a Conta da AWS ou namespaces específicos na sua conta, use o console do Amazon Redshift.

Faça login em https://console.aws.amazon.com/redshiftv2/. Depois, associe uma ou mais unidades de compartilhamento de dados que são compartilhadas de outras contas com toda a Conta da AWS ou namespaces específicos na sua conta. Para obter mais informações, consulte Associar uma unidade de compartilhamento de dados de uma Conta da AWS diferente no Amazon Redshift.

Depois que a Conta da AWS ou namespaces específicos são associados, as unidades de compartilhamento de dados ficarão disponíveis para consumo. Você também pode alterar a associação de datashare a qualquer momento. Ao alterar a associação de namespaces individuais de uma Conta da AWS, o Amazon Redshift substitui os namespaces pelas informações da Conta da AWS. Todos os namespaces na conta passam a ter acesso aos dados, incluindo quaisquer namespaces criados após a associação. Ao alterar a associação de uma Conta da AWS para namespaces específicos, o Amazon Redshift substitui as informações da Conta da AWS pelas informações do namespace. Os namespaces que tinham acesso anteriormente perderão o acesso caso não sejam especificados na nova associação.

ações do administrador de consumidor

Com o Amazon Redshift, você pode realizar tarefas administrativas em clusters de consumidor para gerenciar a ingestão de dados e o processamento de carregamento.

Se você for um administrador de consumidor, siga estas etapas:

1. Liste os conjuntos de dados disponibilizados para você e visualize o conteúdo dos datashares. O conteúdo da unidade de compartilhamento de dados só estará disponível quando o administrador do produtor tiver autorizado a unidade de compartilhamento de dados e o administrador do consumidor tiver aceitado e associado a unidade de compartilhamento de dados. Para obter mais informações, consulte DESC DATASHARE e SHOW DATASHARES.

   O exemplo a seguir exibe as informações de datashares de entrada de um namespace de produtor especificado. Ao executar o DESC DATAHSARE como administrador do consumidor, você deve especificar o NAMESPACE e o ID da conta para visualizar as unidades de compartilhamento de dados de entrada. Para unidades de compartilhamento de dados de saída, especifique o nome da unidade de compartilhamento de dados.

   ```
   SHOW DATASHARES LIKE 'sales%';
   ```

```
share_name | share_owner | source_database | consumer_database | share_type
 | createdate | is_publicaccessible | share_acl | producer_account |
 producer_namespace
-----------+-------------+-----------------+-------------------+------------
+------------+---------------------+-----------+------------------
+--------------------------------------
salesshare |             |                 |                   | INBOUND    |
            |        t            |           | 123456789012    | 'dd8772e1-
d792-4fa4-996b-1870577efc0d'
```

```
DESC DATASHARE salesshare OF ACCOUNT '123456789012' NAMESPACE 'dd8772e1-
d792-4fa4-996b-1870577efc0d';


 producer_account |          producer_namespace          | share_type | share_name |
 object_type |           object_name
-----------------+--------------------------------------+------------+------------
+-------------+---------------------------------
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d | INBOUND    | salesshare |
 table       | public.tickit_users_redshift
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d | INBOUND    | salesshare |
 table       | public.tickit_venue_redshift
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d | INBOUND    | salesshare |
 table       | public.tickit_category_redshift
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d | INBOUND    | salesshare |
 table       | public.tickit_date_redshift
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d | INBOUND    | salesshare |
 table       | public.tickit_event_redshift
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d | INBOUND    | salesshare |
 table       | public.tickit_listing_redshift
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d | INBOUND    | salesshare |
 table       | public.tickit_sales_redshift
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d | INBOUND    | salesshare |
 schema      | public
(8 rows)
```

Somente superusuários de cluster podem fazer isso. Você também pode usar SVV_DATASHARES para visualizar os datashares e SVV_DATAASHARE_OBJECTS para vsualizar os objetos dentro do datashare.

O exemplo a seguir exibe os datashares de entrada em um cluster de consumidores.

```
SELECT * FROM SVV_DATASHARES WHERE share_name LIKE 'sales%';

share_name | share_owner | source_database | consumer_database | share_type
 | createdate | is_publicaccessible | share_acl | producer_account |
 producer_namespace
-----------+-------------+-----------------+-------------------+-----------
+------------+---------------------+-----------+-----------------
+--------------------------------------
salesshare |             |                 |                   | INBOUND    |
            |        t            |           | 123456789012      | 'dd8772e1-
d792-4fa4-996b-1870577efc0d'
```

```
SELECT * FROM SVV_DATASHARE_OBJECTS WHERE share_name LIKE 'sales%';
 share_type | share_name | object_type |           object_name           |
 producer_account |          producer_namespace
------------+------------+-------------+---------------------------------
+------------------+--------------------------------------
 INBOUND    | salesshare | table       | public.tickit_users_redshift    |
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d
 INBOUND    | salesshare | table       | public.tickit_venue_redshift    |
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d
 INBOUND    | salesshare | table       | public.tickit_category_redshift |
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d
 INBOUND    | salesshare | table       | public.tickit_date_redshift     |
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d
 INBOUND    | salesshare | table       | public.tickit_event_redshift    |
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d
 INBOUND    | salesshare | table       | public.tickit_listing_redshift  |
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d
 INBOUND    | salesshare | table       | public.tickit_sales_redshift    |
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d
 INBOUND    | salesshare | schema      | public                          |
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d
(8 rows)
```

2. Crie bancos de dados locais que façam referência aos datashares. Especifique o NAMESPACE e o ID da conta ao criar o banco de dados com base na unidade de compartilhamento de dados. Para obter mais informações, consulte CREATE DATABASE.

```
CREATE DATABASE sales_db FROM DATASHARE salesshare OF ACCOUNT '123456789012'
 NAMESPACE 'dd8772e1-d792-4fa4-996b-1870577efc0d';
```

Se você quiser um controle mais granular sobre o acesso aos objetos no banco de dados local, use a cláusula WITH PERMISSIONS ao criar o banco de dados. Com essa cláusula, é possível conceder permissões no nível do objeto para objetos no banco de dados na etapa 4.

```
CREATE DATABASE sales_db WITH PERMISSIONS FROM DATASHARE salesshare OF ACCOUNT
 '123456789012' NAMESPACE 'dd8772e1-d792-4fa4-996b-1870577efc0d';
```

Você pode ver os bancos de dados criados a partir do datashare consultando a visualização SVV_REDSHIFT_DATABASES. Você pode se conectar diretamente a esses bancos de dados ou a um banco de dados local em seu cluster de consumidor e executar uma consulta entre os bancos de dados criados nas unidades de compartilhamento de dados. Você não pode criar um datashare em cima de objetos de banco de dados criados a partir de um datashare existente. Porém, você pode copiar os dados em uma tabela separada no cluster de consumidores, executar qualquer processamento necessário e, em seguida, compartilhar os novos objetos criados.

3. (Opcional) Crie esquemas externos para referenciar e atribuir permissões detalhadas a esquemas específicos no banco de dados do consumidor importado no cluster de consumidores. Para obter mais informações, consulte CREATE EXTERNAL SCHEMA.

```
CREATE EXTERNAL SCHEMA sales_schema FROM REDSHIFT DATABASE 'sales_db' SCHEMA
 'public';
```

4. Conceda permissões em bancos de dados e referências de esquema criadas a partir das unidades de compartilhamento de dados para usuários ou funções no cluster consumidor conforme necessário. Para ter mais informações, consulte GRANT ou REVOKE.

```
GRANT USAGE ON DATABASE sales_db TO Bob;
```

```
GRANT USAGE ON SCHEMA sales_schema TO ROLE Analyst_role;
```

Se tiver criado o banco de dados sem WITH PERMISSIONS, você só poderá atribuir permissões em todo o banco de dados criado a partir da unidade de compartilhamento de dados para usuários ou funções. Em alguns casos, você precisa de controles refinados em um subconjunto de objetos de banco de dados criados a partir do datashare. Nesse caso, você pode criar uma referência de esquema externo apontando para esquemas específicos na unidade de compartilhamento de dados conforme descrito na etapa anterior. Em seguida, você pode fornecer permissões detalhadas no nível do esquema. Você também pode criar exibições de vinculação tardia sobre

objetos compartilhados e usá-las para atribuir permissões detalhadas. Você também pode considerar que os clusters de produtores criem conjuntos de dados adicionais para você com os detalhes necessários. Você pode criar quantas referências de esquema quiser para o banco de dados criado a partir da unidade de compartilhamento de dados conforme necessário.

Se tiver criado o banco de dados com WITH PERMISSIONS na etapa 2, você deverá atribuir permissões no nível de objeto para objetos no banco de dados compartilhado. Um usuário com apenas a permissão USAGE não poderá acessar nenhum objeto em um banco de dados criado com WITH PERMISSIONS até receber permissões adicionais no nível de objeto.

```
GRANT SELECT ON sales_db.public.tickit_sales_redshift to Bob;
```

5. Consultar dados nos objetos compartilhados nos datashares.

   Usuários e funções com permissões em bancos de dados consumidores e esquemas em clusters consumidores podem explorar e navegar nos metadados de quaisquer objetos compartilhados. Eles também podem explorar e navegar por objetos locais em um cluster de consumidores. Para fazer isso, use drivers JDBC ou ODBC ou visualizações SVV_ALL e SVV_REFISH.

   Os clusters de produtores podem ter muitos esquemas no banco de dados, tabelas e visualizações dentro de cada esquema. Os usuários do lado do consumidor podem ver apenas o subconjunto de objetos que são disponibilizados através do datashare. Esses usuários não conseguem ver todos os metadados do cluster de produtor. Essa abordagem ajuda a fornecer controle de segurança de metadados detalhados com compartilhamento de dados.

   Você continua se conectando a bancos de dados de cluster locais. Mas agora, você também pode ler dos bancos de dados e esquemas que são criados a partir do datashare usando a notação database.schema.table de três partes. Você pode executar consultas que abrangem todos e todos os bancos de dados que são visíveis para você. Estes podem ser bancos de dados locais no cluster ou bancos de dados criados a partir dos datashares. Os clusters de consumidores não podem se conectar aos bancos de dados criados a partir dos datashares.

   Você pode acessar os dados usando a qualificação completa. Para obter mais informações, consulte Exemplos de consulta entre bancos de dados.

   ```
   SELECT * FROM sales_db.public.tickit_sales_redshift;
   ```

Você só pode usar instruções SELECT em objetos compartilhados. No entanto, você pode criar tabelas no cluster de consumidores consultando os dados dos objetos compartilhados em um banco de dados local diferente.

Além de executar consultas, os consumidores podem criar visualizações em objetos compartilhados. Somente visualizações de vinculação tardia e visões materializadas são compatíveis. O Amazon Redshift não oferece suporte a visualizações regulares em dados compartilhados. As visualizações criadas pelos consumidores podem abranger vários bancos de dados locais ou bancos de dados criados a partir de conjuntos de dados. Para obter mais informações, consulte CREATE VIEW.

```
// Connect to a local cluster database

// Create a view on shared objects and access it.
CREATE VIEW sales_data
AS SELECT *
FROM sales_db.public.tickit_sales_redshift
WITH NO SCHEMA BINDING;

SELECT * FROM sales_data;
```

Compartilhamento de dados entre Regiões da AWS

Você pode compartilhar dados para fins de leitura entre clusters do Amazon Redshift nas Regiões da AWS. Com o compartilhamento de dados entre regiões, é possível compartilhar dados entre Regiões da AWS sem a necessidade de copiar dados manualmente. Não é necessário descarregar seus dados no Amazon S3 e copiar os dados em um novo cluster do Amazon Redshift ou executar cópia de snapshot entre regiões.

Com o compartilhamento de dados entre regiões, você pode compartilhar dados entre clusters na mesma Conta da AWS ou em Contas da AWS diferentes, mesmo quando os clusters estão em regiões diferentes. Ao compartilhar dados com clusters do Amazon Redshift que estão na mesma Conta da AWS, mas em Regiões da AWS diferentes, siga o mesmo fluxo de trabalho usado para compartilhar dados dentro de uma Conta da AWS. Para obter mais informações, consulte Compartilhar o acesso de leitura aos dados em uma Conta da AWS.

Se os clusters que compartilham dados estiverem em Contas da AWS e Regiões da AWS diferentes, siga o mesmo fluxo de trabalho do compartilhamento de dados entre Contas da AWS e inclua

associações de nível de região no cluster de consumidor. O compartilhamento de dados entre regiões permite associar a unidade de compartilhamento de dados com toda a Conta da AWS, com toda a Região da AWS ou com namespaces específicos em uma Região da AWS. Para obter mais informações sobre compartilhamento de dados entre Contas da AWS, consulte Compartilhamento de dados entre Contas da AWS.

Ao consumir dados de uma região diferente, o consumidor paga a taxa de transferência de dados entre regiões, da região produtora para a região consumidora.

Para usar a unidade de compartilhamento de dados, um administrador de conta de consumidor pode associar a unidade de compartilhamento de dados de uma das três maneiras a seguir.

- Associação com uma Conta da AWS inteira abrangendo todos as suas Regiões da AWS
- Associação com uma Região da AWS específica em uma Conta da AWS
- Associação a namespaces específicos em uma Região da AWS

Quando o administrador escolhe toda a Conta da AWS, todos os namespaces existentes e futuros em diferentes Regiões da AWS na conta têm acesso às unidades de compartilhamento de dados. Um administrador de conta de consumidor também pode escolher namespaces ou Regiões da AWS específicos dentro de uma região para conceder acesso às unidades de compartilhamento de dados.

Se você for um administrador de produtor ou proprietário de banco de dados, crie uma unidade de compartilhamento de dados, adicione objetos de banco de dados e consumidores de dados à unidade de compartilhamento de dados e conceda permissões aos consumidores de dados. Para obter mais informações, consulte ações do administrador de produtor.

Se você for um administrador de conta de produtor, autorize as unidades de compartilhamento de dados usando a AWS Command Line Interface (AWS CLI) ou o console do Amazon Redshift e escolha os consumidores de dados.

Se você for um administrador de conta de consumidor — siga estas etapas:

Para associar uma ou mais unidades de compartilhamento de dados compartilhados de outras contas com toda a Conta da AWS ou regiões e namespaces específicos da Regiões da AWS em uma Região da AWS, use o console do Amazon Redshift.

Com o compartilhamento de dados entre regiões, você pode adicionar clusters em uma Região da AWS específica usando a AWS Command Line Interface (AWS CLI) ou o console do Amazon Redshift.

Para especificar um ou mais regiões da AWS, você pode usar o comando da CLI `associate-data-share-consumer` com a opção `consumer-region` opcional.

Com a CLI, o exemplo a seguir associa o `Salesshare` à Conta da AWS completa com a opção `associate-entire-account`. Você pode associar apenas uma região de cada vez.

```
aws redshift associate-data-share-consumer
--region {PRODUCER_REGION}
--data-share-arn arn:aws:redshift:{PRODUCER_REGION}:{PRODUCER_ACCOUNT}:datashare:
{PRODUCER_CLUSTER_NAMESPACE}/Salesshare
--associate-entire-account
```

O exemplo a seguir associa o `Salesshare` com a região Leste dos EUA (Ohio) (`us-east-2`).

```
aws redshift associate-data-share-consumer
--region {PRODUCER_REGION}
--data-share-arn arn:aws:redshift:{PRODUCER_REGION}:0123456789012:datashare:
{PRODUCER_CLUSTER_NAMESPACE}/Salesshare
--consumer-region 'us-east-2'
```

O exemplo a seguir associa o `Salesshare` a um namespace de consumidor específico em outra Conta da AWS na região Ásia-Pacífico (Sydney) (`ap-southeast-2`).

```
aws redshift associate-data-share-consumer
--data-share-arn arn:aws:redshift:{PRODUCER_REGION}:{PRODUCER_ACCOUNT}:datashare:
{PRODUCER_CLUSTER_NAMESPACE}/Salesshare
--consumer-arn 'arn:aws:redshift:ap-southeast-2:{CONSUMER_ACCOUNT}:namespace:
{ConsumerImmutableClusterId}'
```

Você pode usar o console do Amazon Redshift para associar unidades de compartilhamento de dados com toda a Conta da AWS ou namespaces ou Regiões da AWS específicos em uma Região da AWS. Para isso, faça login em https://console.aws.amazon.com/redshiftv2/. Em seguida, associe uma ou mais unidades de compartilhamento de dados compartilhadas de outras contas com toda a Conta da AWS, com toda a Região da AWS ou com um namespace específico em uma Região da AWS. Para obter mais informações, consulte Associar uma unidade de compartilhamento de dados de uma Conta da AWS diferente no Amazon Redshift.

Depois que a Conta da AWS ou namespaces específicos são associados, as unidades de compartilhamento de dados ficarão disponíveis para consumo. Você também pode alterar a

associação de datashare a qualquer momento. Ao alterar a associação de namespaces individuais de uma Conta da AWS, o Amazon Redshift substitui os namespaces pelas informações da Conta da AWS. Ao alterar a associação de uma Conta da AWS para namespaces específicos, o Amazon Redshift substitui as informações da Conta da AWS pelas informações do namespace. Ao alterar a associação de uma Conta da AWS inteira para regiões e namespaces específicos da AWS, o Amazon Redshift substitui as informações da Conta da AWS pelas informações específicas da região e do namespace.

Se você for um administrador de consumidor, poderá criar bancos de dados locais que façam referência às unidades de compartilhamento de dados e conceder permissões em bancos de dados criados das unidades de compartilhamento de dados a usuários ou perfis no cluster de consumidor, conforme necessário. Também é possível criar visualizações em objetos compartilhados e criar esquemas externos para referenciar e atribuir permissões detalhadas a esquemas específicos no banco de dados de consumidor importado no cluster de consumidor. Para obter mais informações, consulte ações do administrador de consumidor.

Gerenciar o controle de custos para compartilhamento de dados entre regiões

Com o Amazon Redshift, é possível gerenciar o controle de custos do compartilhamento de dados entre regiões configurando esse recurso para que limite a quantidade de dados que são transferidos entre regiões da AWS. Ao gerenciar o controle de custos do compartilhamento de dados entre Regiões, é possível definir limites de transferência de dados, monitorar o uso da transferência de dados e receber notificações quando esses limites estiverem próximos de ser atingidos ou forem excedidos.

Ao consumir dados de uma região diferente, o consumidor paga a taxa de transferência de dados entre regiões, da região produtora para a região consumidora. O preço da transferência de dados varia conforme as diferentes regiões. A cobrança é baseada nos bytes de dados verificados para cada execução de consulta bem-sucedida. Para obter mais informações sobre preço do Amazon Redshift, consulte Preço do Amazon Redshift.

Você é cobrado pelo número de bytes, arredondado para o próximo megabyte, com um mínimo de 10 MB por consulta. Você pode definir controles de custo sobre o uso da consulta e exibir a quantidade de dados que estiverem sendo transferidos por consulta em seu cluster.

Para monitorar e controlar o uso e o custo associado ao uso do compartilhamento de dados entre regiões, você pode criar limites de uso diário, semanal e mensal e definir ações que o Amazon Redshift executará automaticamente se esses limites forem atingidos para ajudar a manter seu orçamento com previsibilidade.

Dependendo dos limites de uso definidos, as ações que o Amazon Redshift realiza podem ser registrar um evento em uma tabela do sistema, enviar um alarme do CloudWatch e notificar um administrador com um Amazon SNS ou desativar o compartilhamento de dados entre regiões para uso posterior.

Para definir limites de uso no console do Amazon Redshift, escolha Configure usage limit (Configurar limite de uso) em Actions (Ações) para seu cluster. Você pode monitorar suas tendências de uso e receber alertas sobre o uso que exceda seus limites definidos com métricas do CloudWatch geradas automaticamente a partir das guias Cluster performance (Performance do cluster) ou Monitoring (Monitoramento). Você pode você criar, modificar e excluir limites de uso de maneira programática usando a AWS CLI ou operações de API do Amazon Redshift.

Compartilhar dados licenciados do Amazon Redshift no AWS Data Exchange

Ao criar unidades de compartilhamento de dados do AWS Data Exchange e adicioná–las a um produto do AWS Data Exchange, os provedores podem licenciar dados no Amazon Redshift para que os consumidores possam detectar, assinar e consultar dados atualizados no Amazon Redshift quando tiverem assinaturas do AWS Data Exchange ativas.

Com unidades de compartilhamento de dados do AWS Data Exchange adicionadas a um produto do AWS Data Exchange, os consumidores têm acesso automaticamente aos dados de um produto quando a assinatura é iniciada e mantêm o acesso enquanto a assinatura estiver ativa.

Tópicos

- Trabalhar com unidades de compartilhamento de dados do AWS Data Exchange como produtor
- Trabalhar com unidades de compartilhamento de dados do AWS Data Exchange como consumidor

Trabalhar com unidades de compartilhamento de dados do AWS Data Exchange como produtor

Com o Amazon Redshift, ao criar e gerenciar unidades de compartilhamento de dados, você pode compartilhar produtos de dados em tempo real com o AWS Data Exchange como produtor.

Se você for um administrador de produtor, siga estas etapas para gerenciar unidades de compartilhamento de dados de AWS Data Exchange no console do Amazon Redshift:

1. Crie unidades de compartilhamento de dados em seu cluster para compartilhar dados no AWS Data Exchange e conceder acesso ao AWS Data Exchange para as unidades de compartilhamento de dados.

Superusuário de cluster e proprietários de banco de dados podem criar conjuntos de dados. Cada datashare é associado a um banco de dados durante a criação. Somente objetos desse banco de dados podem ser compartilhados nesse datashare. Vários datashares podem ser criados no mesmo banco de dados com a mesma granularidade ou diferente de objetos. Não há limite para o número de unidades de compartilhamento de dados que você pode criar em um cluster.

Também é possível usar o console do Amazon Redshift para criar datashares. Para obter mais informações, consulte Criar uma unidade de compartilhamento de dados.

Use a opção MANAGEDBY ADX para conceder implicitamente acesso à unidade de compartilhamento de dados para o AWS Data Exchange ao executar a instrução CREATE DATASHARE. Isso indica que o AWS Data Exchange gerencia essa unidade de compartilhamento de dados. Você só pode usar a opção MANAGEDBY ADX ao criar uma nova unidade de compartilhamento de dados. Não é possível usar a instrução ALTER DATASHARE para modificar uma unidade de compartilhamento de dados existente para adicionar a opção MANAGEDBY ADX. Depois que a unidade de compartilhamento de dados é criada com a opção MANAGEDBY ADX, somente o AWS Data Exchange pode acessar e gerenciar a unidade de compartilhamento de dados.

```
CREATE DATASHARE salesshare
[[SET] MANAGEDBY [=] {ADX} ];
```

2. Adicione objetos às unidades de compartilhamento de dados. O administrador do produtor continua gerenciando objetos de unidades de compartilhamento de dados que estão disponíveis em uma unidade de compartilhamento de dados do AWS Data Exchange.

   Para adicionar objetos a um datashare, adicione o esquema antes de adicionar objetos. Quando você adiciona um esquema, o Amazon Redshift não adiciona todos os objetos abaixo dele. Você deve adicioná-las explicitamente. Para obter mais informações, consulte ALTER DATASHARE.

   ```
   ALTER DATASHARE salesshare ADD SCHEMA PUBLIC;
   ALTER DATASHARE salesshare ADD TABLE public.tickit_sales_redshift;
   ALTER DATASHARE salesshare ADD ALL TABLES IN SCHEMA PUBLIC;
   ```

   Você também pode adicionar visualizações a um datashare.

```
CREATE VIEW public.sales_data_summary_view AS SELECT * FROM
 public.tickit_sales_redshift;
ALTER DATASHARE salesshare ADD TABLE public.sales_data_summary_view;
```

Use ALTER DATASHARE para compartilhar esquemas, tabelas, visualizações e funções em um determinado esquema. Os superusuários, os proprietários da unidade de compartilhamento de dados ou os usuários que têm a permissão ALTER ou ALL na unidade podem alterar a unidade de compartilhamento de dados para adicionar ou remover objetos. Os usuários devem ter permissões para adicionar ou remover objetos da unidade de compartilhamento de dados. Os usuários também devem ser os proprietários dos objetos ou ter permissões SELECT, USAGE ou ALL nos objetos.

Use a cláusula INCLUDENEW para adicionar novas tabelas, visualizações ou funções definidas pelo usuário (UDFs) do SQL criadas em um esquema especificado ao datashare. Somente superusuários podem alterar essa propriedade para cada par datashare-esquema.

```
ALTER DATASHARE salesshare ADD SCHEMA PUBLIC;
ALTER DATASHARE salesshare SET INCLUDENEW = TRUE FOR SCHEMA PUBLIC;
```

Também é possível usar o console do Amazon Redshift para adicionar ou remover objetos de datashare. Para obter mais informações, consulte Adicionar objetos de unidade de compartilhamento de dados às unidades de compartilhamento de dados, Remover objetos do datashare de datashares e Editar unidades de compartilhamento de dados AWS Data Exchange.

3. Para autorizar o acesso às unidades de compartilhamento de dados para o AWS Data Exchange, siga um destes procedimentos:

   - Autorize explicitamente o acesso à unidade de compartilhamento de dados para o AWS Data Exchange usando a palavra-chave ADX na API `aws redshift authorize-data-share`. Isso permite que o AWS Data Exchange reconheça a unidade de compartilhamento de dados na conta de serviço e gerencie a associação de consumidores à unidade de compartilhamento de dados.

     ```
     aws redshift authorize-data-share
     --data-share-arn arn:aws:redshift:us-east-1:{PRODUCER_ACCOUNT}:datashare:
     {PRODUCER_CLUSTER_NAMESPACE}/salesshare
     --consumer-identifier ADX
     ```

Use uma chave condicional `ConsumerIdentifier` para que as APIs `AuthorizeDataShare` e `DeauthorizeDataShare` permitam ou neguem explicitamente que o AWS Data Exchange faça chamadas às duas APIs na política do IAM.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Deny",
            "Action": [
                "redshift:AuthorizeDataShare",
                "redshift:DeauthorizeDataShare"
            ],
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {
                    "redshift:ConsumerIdentifier": "ADX"
                }
            }
        }
    ]
}
```

- Use o console do Amazon Redshift para autorizar ou remover a autorização das unidades de compartilhamento de dados do AWS Data Exchange. Para obter mais informações, consulte Autorizar ou remover autorização de unidades de compartilhamento de dados (pré-visualização).
- Opcionalmente, você pode autorizar implicitamente o acesso à unidade de compartilhamento de dados AWS Data Exchange importando a unidade de compartilhamento de dados para um conjunto de dados do AWS Data Exchange.

Para remover a autorização para acesso à unidade de compartilhamento de dados do AWS Data Exchange, use a palavra-chave `ADX` na operação de API `aws redshift deauthorize-data-share`. Com isso, você permite que o AWS Data Exchange reconheça a unidade de compartilhamento de dados na conta de serviço e gerencie a remoção da associação da unidade de compartilhamento de dados.

```
aws redshift deauthorize-data-share
--data-share-arn arn:aws:redshift:us-east-1:{PRODUCER_ACCOUNT}:datashare:
{PRODUCER_CLUSTER_NAMESPACE}/salesshare
--consumer-identifier ADX
```

4. Liste os datashares criados no cluster e examine o conteúdo do datashare.

   O exemplo a seguir exibe as informações de uma unidade de compartilhamento de dados chamada salesshare. Para obter mais informações, consulte DESC DATASHARE e SHOW DATASHARES.

```
DESC DATASHARE salesshare;

 producer_account  |          producer_namespace          | share_type | share_name
 | object_type |           object_name          |   include_new
------------------+--------------------------------------+------------
+------------+-------------+----------------------------------+-------------------
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_users_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_venue_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_category_redshift|
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_date_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_event_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_listing_redshift |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_sales_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | schema      | public                         |  t
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | view        | public.sales_data_summary_view |
```

   O exemplo a seguir exibe os datashares de saída em um cluster de produtores.

```
SHOW DATASHARES LIKE 'sales%';
```

   A saída será semelhante à seguinte.

```
share_name | share_owner  | source_database | consumer_database | share_type |
 createdate      | is_publicaccessible  | share_acl | producer_account |
 producer_namespace
-----------+--------------+-----------------+-------------------+------------
+--------------------+----------------------+-----------+-----------------
+--------------------------------------
salesshare |    100       | dev             |                   |  OUTBOUND
 | 2020-12-09 02:27:08 |          True        |           |   123456789012   |
 13b8833d-17c6-4f16-8fe4-1a018f5ed00d
```

Para obter mais informações, consulte DESC DATASHARE e SHOW DATASHARES.

Você também pode usar SVV_DATASHARES, SVV_DATASHARE_CONSUMERS e SVV_DATASHARE_OBJECTS para exibir os datashares, os objetos dentro do datashare e os consumidores de datashare.

5. Descarte datashares. Recomendamos não excluir uma unidade de compartilhamento de dados do AWS Data Exchange compartilhada com outras Contas da AWS usando a instrução DROP DATASHARE. Essas contas perderão acesso à unidade de compartilhamento de dados. Essa ação é irreversível. Isso pode violar os termos da oferta de produtos de dados no AWS Data Exchange. Para excluir uma unidade de compartilhamento de dados do AWS Data Exchange, consulte Observações sobre o uso de DROP DATASHARE.

   O exemplo a seguir descarta uma unidade de compartilhamento de dados chamado salesshare.

   ```
   DROP DATASHARE salesshare;
   ERROR:  Drop of ADX-managed datashare salesshare requires session variable
    datashare_break_glass_session_var to be set to value '620c871f890c49'
   ```

   Para permitir o descarte de uma unidade de compartilhamento de dados do AWS Data Exchange, defina a variável datashare_break_glass_session_var e execute a instrução DROP DATASHARE novamente. Para excluir uma unidade de compartilhamento de dados do AWS Data Exchange, consulte Observações sobre o uso de DROP DATASHARE.

   Também é possível usar o console do Amazon Redshift para excluir datashares. Para obter mais informações, consulte Excluir unidades de compartilhamento de dados AWS Data Exchange criadas em sua conta.

6. Use ALTER DATASHARE para remover objetos de datashares em qualquer ponto do datashare. Use REVOKE USAGE ON para revogar permissões no datashare para determinados

consumidores. Ele revoga as permissões USAGE em objetos dentro de uma unidade de compartilhamento de dados e interrompe instantaneamente o acesso a todos os clusters de consumidores. Listar conjuntos de dados e as consultas de metadados, como listar bancos de dados e tabelas, não retorna os objetos compartilhados depois que o acesso é revogado.

```
ALTER DATASHARE salesshare REMOVE TABLE public.tickit_sales_redshift;
```

Também é possível usar o console do Amazon Redshift para editar datashares. Para obter mais informações, consulte Editar unidades de compartilhamento de dados AWS Data Exchange.

7. Conceder ou revogar GRANT USAGE de unidades de compartilhamento de dados do AWS Data Exchange. Não é possível conceder nem revogar GRANT USAGE na unidade de compartilhamento de dados do AWS Data Exchange. O exemplo a seguir mostra um erro quando a permissão GRANT USAGE é concedida a uma Conta da AWS para uma unidade de compartilhamento de dados que o AWS Data Exchange gerencia.

```
CREATE DATASHARE salesshare MANAGEDBY ADX;
```

```
GRANT USAGE ON DATASHARE salesshare TO ACCOUNT '012345678910';
ERROR:  Permission denied to add/remove consumer to/from datashare salesshare.
 Datashare consumers are managed by ADX.
```

Para ter mais informações, consulte GRANT ou REVOKE.

Se você for um administrador de produtor, siga estas etapas para criar e publicar um produto de unidade de compartilhamento de dados no console do AWS Data Exchange:

- Quando a unidade de compartilhamento de dados AWS Data Exchange é criada, o produtor cria um novo conjunto de dados, importa ativos, cria uma revisão, e cria e publica um novo produto.

  Use o console do Amazon Redshift para criar conjuntos de dados. Para obter mais informações, consulte Criar conjuntos de dados no AWS Data Exchange.

  Para obter mais informações, consulte Fornecer produtos de dados no AWS Data Exchange.

Trabalhar com unidades de compartilhamento de dados do AWS Data Exchange como consumidor

Com o Amazon Redshift, você pode acessar e analisar conjuntos de dados do AWS Data Exchange sem precisar armazenar ou gerenciar cópias de dados.

Se você for um consumidor, siga estas etapas para descobrir produtos de dados que contenham unidades de compartilhamento de dados do AWS Data Exchange e consultar dados do Amazon Redshift:

1. No console do AWS Data Exchange, descubra e assine produtos de dados que contenham unidades de compartilhamento de dados do AWS Data Exchange.

   Depois que sua assinatura começar, será possível acessar dados licenciados do Amazon Redshift que são importados como ativos para conjuntos de dados que contêm unidades de compartilhamento de dados do AWS Data Exchange.

   Para obter mais informações sobre como começar a usar produtos de dados que contêm unidades de compartilhamento de dados AWS Data Exchange, consulte Assinar produtos de dados no AWS Data Exchange.
2. No console do Amazon Redshift, crie um cluster do Amazon Redshift, se necessário.

   Para obter informações sobre como criar um cluster, consulte Criar um cluster.
3. Liste os datashares que são disponibilizados para você e visualize o conteúdo de datashares. Para obter mais informações, consulte DESC DATASHARE e SHOW DATASHARES.

   O exemplo a seguir exibe as informações de datashares de entrada de um namespace de produtor especificado. Ao executar o DESC DATASHARE como administrador do consumidor, você deve especificar a opção ACCOUNT e NAMESPACE para visualizar a unidade de compartilhamento de dados de entrada.

```
DESC DATASHARE salesshare of ACCOUNT '123456789012' NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';

 producer_account  |          producer_namespace          | share_type | share_name
 | object_type |           object_name           |   include_new
-------------------+--------------------------------------+------------
+------------+-------------+----------------------------------+------------------
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_users_redshift    |
```

```
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_venue_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_category_redshift |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_date_redshift     |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_event_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_listing_redshift  |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_sales_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | schema      | public                          |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | view        | public.sales_data_summary_view  |
```

Somente superusuários de cluster podem fazer isso. Você também pode usar SVV_DATASHARES para visualizar os datashares e SVV_DATAASHARE_OBJECTS para vsualizar os objetos dentro do datashare.

O exemplo a seguir exibe os datashares de entrada em um cluster de consumidores.

```
SHOW DATASHARES LIKE 'sales%';


 share_name | share_owner | source_database | consumer_database | share_type
 | createdate | is_publicaccessible | share_acl | producer_account |
 producer_namespace
------------+-------------+-----------------+-------------------+------------
+------------+---------------------+-----------+------------------
+--------------------------------------
 salesshare |             |                 |                   | INBOUND
    |            |         t           |           |   123456789012   |
 13b8833d-17c6-4f16-8fe4-1a018f5ed00d
```

4. Crie bancos de dados locais que façam referência aos datashares. Você deve especificar a opção ACCOUNT e NAMESPACE para criar bancos de dados locais para unidades de compartilhamento de dados do AWS Data Exchange. Para obter mais informações, consulte CREATE DATABASE.

```
CREATE DATABASE sales_db FROM DATASHARE salesshare OF ACCOUNT '123456789012'
 NAMESPACE '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Se você quiser um controle mais granular sobre o acesso aos objetos no banco de dados local, use a cláusula WITH PERMISSIONS ao criar o banco de dados. Com essa cláusula, é possível conceder permissões no nível do objeto para objetos no banco de dados na etapa 6.

```
CREATE DATABASE sales_db WITH PERMISSIONS FROM DATASHARE salesshare OF ACCOUNT
 '123456789012' NAMESPACE '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Você pode ver os bancos de dados criados a partir do datashare consultando a visualização SVV_REDSHIFT_DATABASES. Você pode se conectar diretamente a esses bancos de dados ou a um banco de dados local em seu cluster de consumidor e executar uma consulta entre os bancos de dados criados nas unidades de compartilhamento de dados. Você não pode criar um datashare em cima de objetos de banco de dados criados a partir de um datashare existente. No entanto, você pode copiar os dados em uma tabela separada no cluster de consumidores, executar qualquer processamento necessário e, em seguida, compartilhar os novos objetos que foram criados.

Também é possível usar o console do Amazon Redshift para criar bancos de dados a partir de datashares. Para obter mais informações, consulte Criar bancos de dados a partir de datashares.

5. (Opcional) Crie esquemas externos para consultar e atribuir permissões granulares a esquemas específicos no banco de dados do consumidor importado no cluster de consumidores. Para obter mais informações, consulte CREATE EXTERNAL SCHEMA.

```
CREATE EXTERNAL SCHEMA sales_schema FROM REDSHIFT DATABASE 'sales_db' SCHEMA
 'public';
```

6. Conceda permissões em bancos de dados e referências de esquema criadas a partir das unidades de compartilhamento de dados para usuários ou funções no cluster consumidor conforme necessário. Para ter mais informações, consulte GRANT ou REVOKE.

```
GRANT USAGE ON DATABASE sales_db TO Bob;
```

```
GRANT USAGE ON SCHEMA sales_schema TO ROLE Analyst_role;
```

Se tiver criado o banco de dados sem WITH PERMISSIONS, você só poderá atribuir permissões em todo o banco de dados criado a partir da unidade de compartilhamento de dados para usuários e funções. Em alguns casos, você precisa de controles refinados em um subconjunto de objetos de banco de dados criados a partir do datashare. Nesse caso, você pode criar uma referência de esquema externo que aponte para esquemas específicos no datashare (conforme descrito na etapa anterior) e fornecer permissões detalhadas no nível do esquema.

Você também pode criar exibições de vinculação tardia sobre objetos compartilhados e usá-las para atribuir permissões detalhadas. Você também pode considerar que os clusters de produtores criem conjuntos de dados adicionais para você com os detalhes necessários. Você pode criar quantas referências de esquema quiser para o banco de dados criado a partir da unidade de compartilhamento de dados conforme necessário.

Se tiver criado o banco de dados com WITH PERMISSIONS na etapa 4, você deverá atribuir permissões no nível de objeto para objetos no banco de dados compartilhado. Um usuário com apenas a permissão USAGE não poderá acessar nenhum objeto em um banco de dados criado com WITH PERMISSIONS até receber permissões adicionais no nível de objeto.

```
GRANT SELECT ON sales_db.public.tickit_sales_redshift to Bob;
```

7. Consultar dados nos objetos compartilhados nos datashares.

   Usuários e funções com permissões em bancos de dados consumidores e esquemas em clusters consumidores podem explorar e navegar nos metadados de quaisquer objetos compartilhados. Eles também podem explorar e navegar por objetos locais em um cluster de consumidores. Para fazer isso, eles usam drivers JDBC ou ODBC, os comandos SHOW ou as visualizações SVV_ALL e SVV_REDSHIFT.

   Os clusters de produtores podem ter muitos esquemas no banco de dados, tabelas e visualizações dentro de cada esquema. Os usuários do lado do consumidor podem ver apenas o subconjunto de objetos que são disponibilizados através do datashare. Esses usuários não podem ver todos os metadados do cluster de produtores. Essa abordagem ajuda a fornecer controle de segurança de metadados detalhados com compartilhamento de dados.

   Você continua se conectando a bancos de dados de cluster locais. Mas agora, você também pode ler dos bancos de dados e esquemas que são criados a partir do datashare usando a notação database.schema.table de três partes. Você pode executar consultas que abrangem todos e todos os bancos de dados que são visíveis para você. Estes podem ser bancos de

dados locais no cluster ou bancos de dados criados a partir dos datashares. Ou você pode se conectar diretamente a esses bancos de dados de consumidor e executar consultas nos objetos compartilhados utilizando a notação parcial.

Você pode acessar os dados usando a qualificação completa. Para obter mais informações, consulte Exemplos de consulta entre bancos de dados.

```
SELECT * FROM sales_db.public.tickit_sales_redshift ORDER BY 1,2 LIMIT 5;

 salesid | listid | sellerid | buyerid | eventid | dateid | qtysold | pricepaid |
 commission |      saletime
---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------
       1 |      1 |    36861 |   21191 |    7872 |   1875 |       4 |    728.00 |
   109.20 | 2008-02-18 02:36:48
       2 |      4 |     8117 |   11498 |    4337 |   1983 |       2 |     76.00 |
    11.40 | 2008-06-06 05:00:16
       3 |      5 |     1616 |   17433 |    8647 |   1983 |       2 |    350.00 |
    52.50 | 2008-06-06 08:26:17
       4 |      5 |     1616 |   19715 |    8647 |   1986 |       1 |    175.00 |
    26.25 | 2008-06-09 08:38:52
       5 |      6 |    47402 |   14115 |    8240 |   2069 |       2 |    154.00 |
    23.10 | 2008-08-31 09:17:02
```

Você só pode usar instruções SELECT em objetos compartilhados. No entanto, você pode criar tabelas no cluster de consumidores consultando os dados dos objetos compartilhados em um banco de dados local diferente.

Além das consultas, os consumidores podem criar visualizações em objetos compartilhados. Somente visualizações de vinculação tardia ou visões materializadas são aceitas. O Amazon Redshift não oferece suporte a visualizações regulares em dados compartilhados. As visualizações criadas pelos consumidores podem abranger vários bancos de dados locais ou bancos de dados criados a partir de conjuntos de dados. Para obter mais informações, consulte CREATE VIEW.

```
// Connect to a local cluster database

// Create a view on shared objects and access it.
CREATE VIEW sales_data
AS SELECT *
FROM sales_db.public.tickit_sales_redshift
```

```
WITH NO SCHEMA BINDING;

SELECT * FROM sales_data;
```

Conceitos básicos das unidades de compartilhamento de dados gerenciadas pelo AWS Lake Formation

Com o Amazon Redshift, é possível acessar e compartilhar dados em tempo real entre contas da AWS e clusters do Amazon Redshift por meio de unidades de compartilhamento de dados gerenciadas pelo AWS Lake Formation. As unidades de compartilhamento de dados do AWS Lake Formation permitem que os provedores de dados compartilhem dados em tempo real com segurança entre seu data lake do Amazon S3 e qualquer consumidor, inclusive com outras contas da AWS e clusters do Amazon Redshift.

Trabalhar com unidades de compartilhamento de dados gerenciadas pelo Lake Formation como produtor

Com o Amazon Redshift, você pode acessar e analisar dados compartilhados por meio de unidades de compartilhamento de dados do AWS Lake Formation. As unidades de compartilhamento de dados do AWS Lake Formation permitem o compartilhamento seguro de dados entre contas da AWS e clusters do Amazon Redshift sem a necessidade de copiar ou mover os dados subjacentes.

Ao compartilhar dados com o AWS Lake Formation, é possível definir centralmente as permissões do AWS Lake Formation das unidades de compartilhamento de dados do Amazon Redshift e restringir o acesso dos usuários aos objetos dentro de uma unidade de compartilhamento de dados.

Com o Amazon Redshift, é possível compartilhar dados em tempo real com segurança entre contas da AWS e clusters do Amazon Redshift usando unidades de compartilhamento de dados gerenciadas pelo AWS Lake Formation como produtor. Uma unidade de compartilhamento de dados gerenciada pelo Lake Formation é um objeto usado para compartilhar dados em tempo real do cluster do Amazon Redshift com outros serviços e contas da AWS.

Como administrador do cluster produtor ou do grupo de trabalho, siga estas etapas para compartilhar unidades de compartilhamento de dados com o Lake Formation:

1. Crie unidades de compartilhamento de dados em seu cluster e autorize o AWS Lake Formation a acessá-las.

Somente o superusuário do cluster e proprietários de bancos de dados podem criar unidades de compartilhamento de dados. Cada datashare é associado a um banco de dados durante a criação. Somente objetos desse banco de dados podem ser compartilhados nesse datashare. Vários datashares podem ser criados no mesmo banco de dados com a mesma granularidade ou diferente de objetos. Não há limite para o número de unidades de compartilhamento de dados que você pode criar em um cluster.

```
CREATE DATASHARE salesshare;
```

2. Adicione objetos à unidade de compartilhamento de dados. O administrador do cluster produtor ou do grupo de trabalho continua gerenciando objetos da unidade de compartilhamento de dados disponíveis. Para adicionar objetos a um datashare, adicione o esquema antes de adicionar objetos. Quando você adiciona um esquema, o Amazon Redshift não adiciona todos os objetos abaixo dele. Você deve adicioná-las explicitamente. Para obter mais informações, consulte ALTER DATASHARE.

```
ALTER DATASHARE salesshare ADD SCHEMA PUBLIC;
ALTER DATASHARE salesshare ADD TABLE public.tickit_sales_redshift;
ALTER DATASHARE salesshare ADD ALL TABLES IN SCHEMA PUBLIC;
```

Você também pode adicionar visualizações a um datashare. As visualizações compatíveis são as visualizações padrão, as visualizações de vinculação tardia e as visões materializadas.

```
CREATE VIEW public.sales_data_summary_view AS SELECT * FROM
 public.tickit_sales_redshift;
ALTER DATASHARE salesshare ADD TABLE public.tickit_sales_redshift;
```

Use ALTER DATASHARE para compartilhar esquemas, tabelas e visualizações em determinado esquema. Os superusuários, os proprietários da unidade de compartilhamento de dados ou os usuários que têm a permissão ALTER ou ALL na unidade podem alterar a unidade de compartilhamento de dados para adicionar ou remover objetos. Os usuários de bancos de dados devem ser os proprietários dos objetos ou ter permissões SELECT, USAGE ou ALL nos objetos.

Use a cláusula INCLUDENEW para adicionar novas tabelas e visualizações criadas em um esquema especificado à unidade de compartilhamento de dados. Somente superusuários podem alterar essa propriedade para cada par datashare-esquema.

```
ALTER DATASHARE salesshare ADD SCHEMA PUBLIC;
```

```
ALTER DATASHARE salesshare SET INCLUDENEW = TRUE FOR SCHEMA PUBLIC;
```

3. Conceda acesso à unidade de compartilhamento de dados para uma conta de administrador do Lake Formation.

```
GRANT USAGE ON DATASHARE salesshare TO ACCOUNT '012345678910' VIA DATA CATALOG;
```

Para revogar o uso, use o comando a seguir.

```
REVOKE USAGE ON DATASHARE salesshare FROM ACCOUNT '012345678910' VIA DATA CATALOG;
```

4. Autorize o acesso à unidade de compartilhamento de dados para o Lake Formation usando a operação de API `aws redshift authorize-data-share`. Isso permite que o Lake Formation reconheça a unidade de compartilhamento de dados na conta de serviço e gerencie a associação de consumidores à unidade de compartilhamento de dados.

```
aws redshift authorize-data-share
--data-share-arn arn:aws:redshift:us-east-1:{PRODUCER_ACCOUNT}:datashare:
{PRODUCER_CLUSTER_NAMESPACE}/salesshare
--consumer-identifier {"DataCatalog/<consumer-account-id>"}
```

Para remover a autorização das unidades de compartilhamento de dados gerenciadas pelo Lake Formation, use a operação de API `aws redshift deauthorize-data-share`. Com isso, você permite que o AWS Lake Formation reconheça a unidade de compartilhamento de dados na conta de serviço e remova a autorização.

```
aws redshift deauthorize-data-share
--data-share-arn arn:aws:redshift:us-east-1:{PRODUCER_ACCOUNT}:datashare:
{PRODUCER_CLUSTER_NAMESPACE}/salesshare
--consumer-identifier {"DataCatalog/<consumer-account-id>"}
```

A qualquer momento, se o administrador do cluster produtor ou do grupo de trabalho decidir que não há mais necessidade de compartilhar dados com o cluster consumidor ou o grupo de trabalho, ele poderá usar DROP DATASHARE para excluir a unidade de compartilhamento de dados, desautorizar a unidade de compartilhamento de dados ou revogar permissões da unidade de compartilhamento de dados. As permissões e os objetos associados no Lake Formation não são excluídos automaticamente.

```
DROP DATASHARE salesshare;
```

Depois de autorizar a conta do Lake Formation a gerenciar a unidade de compartilhamento de dados, o administrador do Lake Formation poderá descobrir a unidade de compartilhamento de dados compartilhada, associar a unidade de compartilhamento de dados com um ARN do Catálogo de Dados e criar um banco de dados no AWS Glue Data Catalog vinculado à unidade de compartilhamento de dados. Para associar unidades de compartilhamento de dados utilizando a AWS CLI, use o comando associate-data-share-consumer. Para compartilhar uma unidade de compartilhamento de dados entre Regiões da AWS, especifique o parâmetro `--region` no comando `associate-data-share-consumer` ou use o Console da AWS para escolher os consumidores de dados. O exemplo a seguir demonstra como compartilhar uma unidade de compartilhamento de dados gerenciada pelo Lake Formation entre regiões.

```
aws redshift associate-data-share-consumer --region <region-1>
--data-share-arn 'arn:aws:redshift:us-
east-1:12345678912:datashare:035c45ea-61ce-86f0-8b75-19ac6102c3b7/sample_share'
--consumer-arn 'arn:aws:glue:<region-1>:111912345678:catalog'
```

O administrador do Lake Formation também deve criar recursos locais que definam como os objetos na unidade de compartilhamento de dados devem ser mapeados para objetos do Lake Formation. Para obter mais informações sobre como descobrir unidades de compartilhamento de dados e criar recursos locais, consulte Gerenciar permissões para dados em uma unidade de compartilhamento de dados do Amazon Redshift.

Trabalhar com unidades de compartilhamento de dados gerenciadas pelo Lake Formation como consumidor

Com o Amazon Redshift, é possível acessar e analisar dados compartilhados com você por meio de unidades de compartilhamento de dados do AWS Lake Formation. Uma unidade de compartilhamento de dados é um produto de dados que contém uma coleção de objetos de dados, como tabelas ou bancos de dados, de diferentes fontes de dados.

Depois que o administrador do AWS Lake Formation descobre o convite para a unidade de compartilhamento de dados e cria um banco de dados no AWS Glue Data Catalog vinculado à unidade de compartilhamento de dados, o administrador do cluster consumidor ou do grupo de trabalho pode associar o cluster à unidade de compartilhamento de dados e ao banco de dados no AWS Glue Data Catalog, criar um banco de dados local para o cluster consumidor ou o grupo de

trabalho e conceder acesso a usuários e funções no cluster consumidor do Amazon Redshift para começar a realizar consultas. Siga estas etapas para configurar as permissões de consulta.

1. No console do Amazon Redshift, crie um cluster do Amazon Redshift para funcionar como o cluster consumidor ou o grupo de trabalho, se necessário. Para obter informações sobre como criar um cluster, consulte Criar um cluster.
2. Para listar a quais bancos de dados no cluster consumidor ou grupo de trabalho do AWS Glue Data Catalog os usuários têm acesso, execute o comando SHOW DATABASES.

   ```
   SHOW DATABASES FROM DATA CATALOG [ACCOUNT <account-id>,<account-id2>] [LIKE
    <expression>]
   ```

   Isso lista os recursos que estão disponíveis no Catálogo de Dados, como o ARN do banco de dados do AWS Glue, o nome do banco de dados e as informações sobre a unidade de compartilhamento de dados.
3. Usando o ARN do banco de dados do AWS Glue de SHOW DATABASES, crie um banco de dados local no cluster consumidor ou no grupo de trabalho. Para obter mais informações, consulte CREATE DATABASE.

   ```
   CREATE DATABASE lf_db FROM ARN <lake-formation-database-ARN> WITH [NO] DATA CATALOG
    SCHEMA [<schema>];
   ```
4. Conceda acesso em bancos de dados e referências de esquema criadas a partir das unidades de compartilhamento de dados para usuários e funções no cluster consumidor ou no grupo de trabalho conforme necessário. Para obter mais informações, consulte GRANT ou REVOKE. Observe que os usuários criados pelo comando CREATE USER não podem acessar objetos em unidades de compartilhamento de dados que foram compartilhadas com o Lake Formation. Somente usuários com acesso ao Redshift e ao Lake Formation podem acessar unidades de compartilhamento de dados que foram compartilhadas com o Lake Formation.

   ```
   GRANT USAGE ON DATABASE sales_db TO IAM:Bob;
   ```

   Como administrador do cluster consumidor ou do grupo de trabalho, você só pode atribuir permissões em todo o banco de dados criado a partir da unidade de compartilhamento de dados para os usuários e funções. Em alguns casos, você precisa de controles refinados em um subconjunto de objetos de banco de dados criados a partir do datashare.

Você também pode criar exibições de vinculação tardia sobre objetos compartilhados e usá-las para atribuir permissões detalhadas. Você também pode considerar que os clusters produtores ou os grupos de trabalho criem unidades de compartilhamento de dados adicionais para você com a granularidade necessária. Você pode criar quantas referências de esquema quiser para o banco de dados criado a partir do datashare.

5. Os usuários do banco de dados podem usar as visualizações SVV_EXTERNAL_TABLES e SVV_EXTERNAL_COLUMNS para encontrar todas as tabelas ou colunas compartilhadas no banco de dados do AWS Glue.

```
SELECT * from svv_external_tables WHERE redshift_database_name = 'lf_db';

SELECT * from svv_external_columns WHERE redshift_database_name = 'lf_db';
```

6. Consultar dados nos objetos compartilhados nos datashares.

   Usuários e funções com permissões em bancos de dados consumidores e esquemas em clusters consumidores ou grupos de trabalho podem explorar e navegar nos metadados de quaisquer objetos compartilhados. Eles também podem explorar e navegar em objetos locais em um cluster consumidor. Para fazer isso, eles podem usar drivers JDBC ou ODBC, ou as visualizações SVV_ALL e SVV_EXTERNAL.

```
SELECT * FROM lf_db.schema.table;
```

   Você só pode usar instruções SELECT em objetos compartilhados. No entanto, você pode criar tabelas no cluster de consumidores consultando os dados dos objetos compartilhados em um banco de dados local diferente.

```
// Connect to a local cluster database

// Create a view on shared objects and access it.

CREATE VIEW sales_data
AS SELECT *
FROM sales_db.public.tickit_sales_redshift
WITH NO SCHEMA BINDING;

SELECT * FROM sales_data;
```

Conceitos básicos das gravações em vários data warehouses usando o compartilhamento de dados no Amazon Redshift

É possível compartilhar objetos de banco de dados para leitura e gravação entre diferentes clusters do Amazon Redshift ou grupos de trabalho do Amazon Redshift sem servidor dentro da mesma Conta da AWS, entre contas e entre regiões. Os procedimentos neste tópico mostram como configurar o compartilhamento de dados. Você pode conceder permissões como SELECT, INSERT e UPDATE para tabelas diferentes e USAGE e CREATE para esquemas.

Os dados ficam disponíveis em tempo real para todos os data warehouses assim que você confirma uma transação de gravação. Os administradores da conta do produtor podem determinar se namespaces ou regiões específicos terão acesso somente para leitura, acesso para leitura e gravação ou nenhum acesso aos dados. Os procedimentos pressupõem que você esteja trabalhando em um banco de dados em um cluster provisionado ou um grupo de trabalho do Amazon Redshift sem servidor.

Com o Amazon Redshift, você pode gerenciar o compartilhamento de dados com gravações usando o console ou a interface SQL para controlar o acesso e governar dados em clusters do Amazon Redshift e contas da AWS. As seções a seguir fornecem instruções passo a passo sobre como configurar e gerenciar o compartilhamento de dados com gravações usando o Amazon Redshift.

Para obter uma lista de regiões onde o compartilhamento de dados está disponível, consulte Regiões da AWS em que o compartilhamento de dados está disponível. Para ver considerações e limitações referentes a gravações, consulte Considerações sobre compartilhamento de dados no Amazon Redshift.

> **Note**
>
> As gravações em vários warehouses do Amazon Redshift usando compartilhamento de dados só são compatíveis com o patch 186 do Amazon Redshift para clusters provisionados na versão de acompanhamento atual 1.0.78881 ou posterior e para grupos de trabalho do Amazon Redshift sem servidor na versão 1.0.78890 ou posterior.

Tópicos

- Conectar-se a um banco de dados no Amazon Redshift
- Ações do produtor para novas unidades de compartilhamento de dados no Amazon Redshift

- Ações do consumidor para novas unidades de compartilhamento de dados no Amazon Redshift
- Ações do produtor para unidades de compartilhamento de dados existentes no Amazon Redshift
- Ações do consumidor para unidades de compartilhamento de dados existentes no Amazon Redshift

Conectar-se a um banco de dados no Amazon Redshift

Com o Amazon Redshift, você pode estabelecer conexão com o cluster do data warehouse e executar consultas SQL, carregar dados ou realizar tarefas administrativas. A conexão com um banco de dados refere-se ao processo de criar um canal seguro entre uma ferramenta ou aplicação cliente e o cluster do Amazon Redshift. As seções a seguir fornecem instruções passo a passo sobre como se conectar a um banco de dados do Amazon Redshift.

Console

Conecte-se a um banco de dados para visualizar bancos de dados e objetos dentro de bancos de dados ou visualizar unidades de compartilhamento de dados no data warehouse do Amazon Redshift. As credenciais de usuário usadas para se conectar a um banco de dados especificado devem ter as permissões necessárias para exibir todas as unidades de compartilhamento de dados.

Se não houver conexão local, siga um destes procedimentos:

- Se você for um administrador de produtor, acesse a guia Clusters para ver clusters provisionados ou a guia Configuração de namespaces para ver endpoints sem servidor. Selecione o respectivo cluster ou namespace na lista.
- Na página de detalhes do cluster ou namespace, na guia Unidades de compartilhamento de dados, escolha Conectar-se ao banco de dados e execute um dos seguintes procedimentos:
  - Na seção Unidades de compartilhamento de dados de outros namespaces e Contas da AWS, visualize unidades de compartilhamento de dados de outros clusters, namespaces ou contas.
  - Na seção Unidades de compartilhamento de dados criados no meu cluster, visualize as unidades de compartilhamento de dados no cluster.
- Na janela Conectar ao banco de dados, siga um destes procedimentos:
  - Se escolher Criar uma nova conexão, escolha AWS Secrets Manager para usar um segredo armazenado para autenticar o acesso para a conexão.

Ou escolha Credenciais temporárias para usar credenciais de banco de dados para autenticar o acesso para a conexão. Especifique valores para o Nome do banco de dados e o Usuário do banco de dados.

Selecione Conectar.

- Escolha Usar uma conexão recente para se conectar a outro banco de dados que você tem as permissões necessárias.

  O Amazon Redshift faz a conexão automaticamente.

Depois de estabelecer uma conexão com o banco de dados, você pode começar a criar unidades de compartilhamento de dados, consultar unidades de compartilhamento de dados ou criar bancos de dados usando unidades de compartilhamento de dados.

Ações do produtor para novas unidades de compartilhamento de dados no Amazon Redshift

Com o Amazon Redshift, é possível compartilhar dados em tempo real entre clusters do Amazon Redshift ou contas da AWS usando unidades de compartilhamento de dados. Uma unidade de compartilhamento de dados é um objeto consumidor-produtor usado para compartilhar dados em tempo real do cluster do Amazon Redshift com outros clusters ou contas da AWS. Ao criar unidades de compartilhamento de dados, é possível compartilhar dados de maneira segura e, ao mesmo tempo, manter o controle sobre o acesso e garantir que os dados permaneçam atualizados. As seções a seguir apresentam detalhes sobre como criar unidades de compartilhamento de dados e adicionar objetos de banco de dados, como esquemas, tabelas e visualizações, para compartilhar dados em tempo real com segurança.

Tópicos

- Criar uma unidade de compartilhamento de dados no Amazon Redshift
- Adicionar objetos a uma unidade de compartilhamento de dados no Amazon Redshift
- Adicionar consumidores de dados à unidade de compartilhamento de dados no Amazon Redshift
- Autorizar uma unidade de compartilhamento de dados no Amazon Redshift

Criar uma unidade de compartilhamento de dados no Amazon Redshift

Uma unidade de compartilhamento de dados é um contêiner lógico de objetos, permissões e consumidores do banco de dados. Consumidores são clusters provisionados do Amazon Redshift ou namespaces do Amazon Redshift sem servidor em sua conta e em outras Contas da AWS. Cada unidade de compartilhamento de dados é associada ao banco de dados no qual foi criada e somente objetos desse banco de dados podem ser adicionados. Como administrador de produtor, você pode criar unidades de compartilhamento de dados no console e com SQL seguindo um dos procedimentos abaixo.

Console

No console, você pode criar unidades de compartilhamento de dados nas guias Unidades de compartilhamento de dados na página de detalhes do cluster ou namespace. Após a criação da unidade de compartilhamento de dados, você pode criar unidades usando unidades de compartilhamento de dados em um consumidor como administrador do consumidor.

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. No menu de navegação, escolha Clusters e o seu cluster. A página de detalhes do cluster é exibida.
3. Na página de detalhes do cluster ou namespace, na guia Unidades de compartilhamento de dados, na seção Unidades de compartilhamento de dados, conecte-se a um banco de dados se você não tiver uma conexão de banco de dados. Na seção Unidades de compartilhamento de dados criados na minha conta, escolha Criar unidade de compartilhamento de dados. A página Criar compartilhamento de dados é exibida.
4. Escolha Criar datashare. Você só pode criar um datashare a partir de um banco de dados local. Se você ainda não tiver se conectado ao banco de dados, a página Conectar-se ao banco de dados será aberta. Siga as etapas em Conectar-se a um banco de dados para se conectar a um banco de dados. Se houver uma conexão recente, a página Criar compartilhamento de dados será exibida.
5. Na seção Datashare information (Informações da unidade de compartilhamento de dados), escolha uma destas opções:
   - Selecione Unidade de compartilhamento de dados para criar unidades e compartilhar dados para fins de leitura ou gravação entre diferentes data warehouses do Amazon Redshift (clusters provisionados ou endpoints sem servidor) ou na mesma Conta da AWS ou em Contas da AWS diferentes.

- Selecione Unidade de compartilhamento de dados do AWS Data Exchange para criar unidades de compartilhamento de dados e licenciar dados por meio do AWS Data Exchange.

6. Especifique valores para Datashare name (Nome da unidade de compartilhamento de dados), Database name (Nome do banco de dados) e Publicly accessible (Acessível publicamente). Quando você alterar o nome do banco de dados, faça uma nova conexão de banco de dados.
7. Adicione objetos à unidade de compartilhamento de dados usando as seções Permissões com escopo ou Permissões diretas. Para adicionar objetos a uma unidade de compartilhamento de dados, consulte Criar uma unidade de compartilhamento de dados no Amazon Redshift.
8. Na seção Consumidores de dados, você pode optar por publicar no Amazon Redshift ou no AWS Glue Data Catalog, o que inicia o processo de compartilhamento de dados com o Lake Formation. Publicar a unidade de compartilhamento de dados no Amazon Redshift significa compartilhar os dados com outro namespace ou conta do Amazon Redshift que atua como consumidor.

   > **Note**
   >
   > Depois que a unidade de compartilhamento de dados é criada, não é possível editar a configuração para publicar na outra opção.

9. Escolha Criar datashare.

SQL

O seguinte comando cria uma unidade de compartilhamento de dados:

```
CREATE DATASHARE salesshare;
```

No momento da criação da unidade de compartilhamento de dados, cada unidade de compartilhamento de dados é associada a um banco de dados. Somente objetos desse banco de dados podem ser compartilhados nesse datashare. Vários datashares podem ser criados no mesmo banco de dados com a mesma granularidade ou diferente de objetos. Não há limite para o número de datashares que um cluster pode criar. Também é possível usar o console do Amazon Redshift para criar datashares. Para obter mais informações, consulte CREATE DATASHARE.

Você também pode controlar restrições de segurança à unidade de compartilhamento de dados durante a criação. O exemplo a seguir mostra que o consumidor com acesso IP público tem permissão para ler a unidade de compartilhamento de dados.

```
CREATE DATASHARE my_datashare [PUBLICACCESSIBLE = TRUE];
```

A configuração PUBLICACCESSIBLE = TRUE permite aos consumidores consultar a unidade de compartilhamento de dados em clusters acessíveis ao público e grupos de trabalho provisionados. Deixe isso de fora ou o defina explicitamente como falso caso você não queira permitir.

Você pode modificar propriedades sobre o tipo de consumidor após a criação da unidade de compartilhamento de dados. Por exemplo, você pode definir que os clusters que desejam consumir dados de um dado datashare não podem ser acessíveis publicamente. Consultas de clusters de consumidores que não atendem às restrições de segurança especificadas na unidade de compartilhamento de dados são rejeitadas no tempo de execução da consulta. Para obter mais informações, consulte ALTER DATASHARE.

Adicionar objetos a uma unidade de compartilhamento de dados no Amazon Redshift

Você pode adicionar objetos de banco de dados de vários tipos no console e com SQL seguindo um dos procedimentos abaixo.

Console

Você pode adicionar objetos à unidade de compartilhamento de dados usando as seções Permissões com escopo ou Permissões diretas. Selecione Conceder permissões com escopo ou Conceder permissões diretas para adicionar objetos. Selecione o botão Adicionar para adicionar objetos. Uma caixa de diálogo é exibida. Siga estas etapas:

1. Se você selecionar Conceder permissões com escopo, a página Conceder permissões com escopo será exibida, na qual você poderá conceder permissões com escopo em nível de banco de dados ou de esquema. As unidades de compartilhamento de dados que têm permissões com escopo têm as permissões especificadas em todos os objetos atuais e futuros dentro do banco de dados ou esquema. Para mais detalhes, consulte ., Permissões em escopo.

    1. Em seguida, selecione Permissões com escopo definido do banco de dados para conceder permissões com escopo em nível de banco de dados. Quando você concede

permissões com escopo, elas se aplicam ao banco de dados atual durante a criação da unidade de compartilhamento de dados. Essas permissões não podem ser concedidas a objetos individuais e são aplicáveis a objetos existentes e novos (esquemas, tabelas, visualizações e UDFs).

2. Selecione as permissões com escopo definido para esquemas, tabelas, visualizações ou funções definidas pelo usuário. Isso significa que todos os objetos no banco de dados têm as permissões selecionadas concedidas aos consumidores. Selecione Conceder para concluir a concessão de permissões com escopo definido do banco de dados.
3. Em seguida, selecione Permissões com escopo definido de esquema para conceder permissões com escopo em nível de esquema. Quando você concede permissões com escopo definido de esquema, todos os objetos adicionados ao esquema têm as permissões da unidade de compartilhamento de dados especificadas.
4. No menu suspenso, selecione o esquema que você deseja adicionar à unidade de compartilhamento de dados. Você pode selecionar apenas um esquema por vez. Em seguida, selecione as permissões diretas que deseja conceder no esquema selecionado.
5. Selecione permissões com escopo definido para os objetos de esquema, como tabelas, visualizações e funções definidas pelo usuário. As permissões são concedidas em todos os objetos correspondentes no esquema. Podem ser objetos existentes ou aqueles adicionados no futuro. Quando aplicada, não é possível remover uma permissão de um objeto sem revogar as permissões com escopo.
6. Selecione Conceder para concluir a concessão de permissões com escopo definido de esquema.

2. Se você selecionar Conceder permissões diretas, a página Conceder permissões diretas será exibida, na qual você poderá conceder permissões diretas em cada nível de objeto, como esquema, tabela, visualização ou função definida pelo usuário. Para conceder permissões diretas, você deve primeiro adicionar os esquemas relevantes à unidade de compartilhamento de dados.

   1. Em seguida, selecione Conceder permissões diretas a esquemas para aplicar permissões diretas em esquemas específicos. Depois, selecione as permissões de esquema para os objetos de esquema, como tabelas, visualizações e funções definidas pelo usuário, e escolha o esquema que deseja adicionar à unidade de compartilhamento de dados. Selecione Conceder para concluir a adição do esquema à unidade de compartilhamento de dados.

2. Depois de adicionar um esquema à unidade de compartilhamento de dados, você pode prosseguir com a adição de permissões diretas para os objetos do esquema. Selecione Conceder permissões diretas novamente. A página Conceder permissões diretas é exibida. Em seguida, acesse as guias de permissões diretas para objetos de esquema.
3. Selecione Conceder permissões diretas para tabelas e visualizações para conceder permissões diretas em nível de objeto a esses objetos. Selecione as permissões diretas e os objetos necessários na lista. Use o campo de pesquisa para encontrar objetos de unidade de compartilhamento de dados. Selecione “Conceder” para concluir a adição de tabelas e visualizações à unidade de compartilhamento de dados.
4. Selecione Conceder permissões diretas a funções definidas pelo usuário para conceder permissões diretas em nível de objeto em funções definidas pelo usuário. Selecione as permissões diretas e os objetos necessários na lista. Use o campo de pesquisa para encontrar objetos de unidade de compartilhamento de dados. Selecione Conceder para concluir a adição de perfis definidos pelo usuário à unidade de compartilhamento de dados.

3. Você também pode escolher se deseja Adicionar objetos futuros. Quando você opta por incluir objetos da unidade de compartilhamento de dados adicionados ao esquema, isso significa que objetos adicionados ao esquema são adicionados automaticamente à unidade de compartilhamento de dados.
4. Escolha Adicionar para concluir a seção e adicionar os objetos. Eles estão listados em Objetos de compartilhamento de dados.
5. Depois de adicionar objetos, você poderá selecionar objetos individuais e editar as permissões. Se você selecionar um esquema, uma caixa de diálogo será exibida perguntando se você gostaria de adicionar Permissões em escopo. Isso faz com que cada objeto existente ou adicionado ao esquema tenha um conjunto pré-selecionado de permissões, indicado para o tipo de objeto. Por exemplo, o administrador pode definir se todas as tabelas adicionadas têm permissões SELECT e UPDATE, por exemplo.
6. Todos os objetos da unidade de compartilhamento de dados estão listados nas seções Permissões com escopo ou Permissões diretas.
7. Na seção Consumidores de dados, você pode adicionar namespaces ou contas da AWS como consumidores da unidade de compartilhamento de dados.
8. Escolha Criar compartilhamento de dados para salvar as alterações.

Depois que você cria a unidade de compartilhamento de dados, ela é exibida na lista em Unidades de compartilhamento de dados criadas em meu namespace. Se escolher uma unidade de compartilhamento de dados na lista, você poderá exibir os consumidores, os objetos e outras propriedades.

SQL

Com SQL, o proprietário da unidade de compartilhamento de dados deve conceder USAGE nos esquemas que deseja adicionar à unidade de compartilhamento de dados. GRANT é usado para conceder várias ações no esquema, incluindo CREATE e USAGE. Os esquemas mantêm objetos compartilhados:

```
CREATE SCHEMA myshared_schema1;
CREATE SCHEMA myshared_schema2;

GRANT USAGE ON SCHEMA myshared_schema1 TO DATASHARE my_datashare;
GRANT CREATE, USAGE ON SCHEMA myshared_schema2 TO DATASHARE my_datashare;
```

Como alternativa, o administrador pode continuar executando comandos ALTER para adicionar um esquema à unidade de compartilhamento de dados. Somente permissões USAGE são concedidas quando um esquema é adicionado assim.

```
ALTER DATASHARE my_datashare ADD SCHEMA myshared_schema1;
```

Depois de adicionar esquemas, o administrador poderá conceder permissões da unidade de compartilhamento de dados em objetos no esquema. Elas podem ser permissões de leitura e gravação: O exemplo GRANT ALL mostra como conceder todas as permissões.

```
GRANT SELECT, INSERT ON TABLE myshared_schema1.table1, myshared_schema1.table2,
 myshared_schema2.table1
TO DATASHARE my_datashare;

GRANT ALL ON TABLE myshared_schema1.table4 TO DATASHARE my_datashare;
```

Você pode continuar executando comandos como ALTER DATASHARE para adicionar tabelas. Quando se faz isso, somente permissões SELECT são concedidas em objetos adicionados.

```
ALTER DATASHARE my_datashare ADD TABLE myshared_schema1.table1,
 myshared_schema1.table2, myshared_schema2.table1;
```

Você pode conceder permissões com escopo definido a uma unidade de compartilhamento de dados em todos os objetos de um tipo dentro de um banco de dados ou esquema. As unidades de compartilhamento de dados que têm permissões com escopo têm as permissões especificadas em todos os objetos atuais e futuros dentro do banco de dados ou esquema.

Você pode visualizar o escopo das permissões com escopo em nível de banco de dados em SVV_DATABASE_PRIVILEGES. Você pode visualizar o escopo das permissões com escopo em nível de esquema em SVV_SCHEMA_PRIVILEGES.

A seguir está a sintaxe para conceder permissões com escopo definido para unidades de compartilhamento de dados. Para ter mais informações sobre permissões com escopo definido, consulte Permissões em escopo.

```
GRANT { CREATE | USAGE | ALTER | DROP } [,...] | ALL [ PRIVILEGES ] }FOR SCHEMAS IN
DATABASE db_name
TO DATASHARE { datashare_name}

GRANT { { SELECT | INSERT | UPDATE | DELETE | DROP | ALTER | TRUNCATE | REFERENCES }
 [, ...] } | ALL [PRIVILEGES] } }FOR TABLES IN
{SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
TO DATASHARE { datashare_name}

GRANT { EXECUTE | ALL [ PRIVILEGES ] }FOR FUNCTIONS IN
{SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
TO DATASHARE { datashare_name}
```

Adicionar consumidores de dados à unidade de compartilhamento de dados no Amazon Redshift

É possível adicionar um ou mais consumidores de dados às unidades de compartilhamento de dados no console ou com SQL. Os consumidores de dados podem ser namespaces que identificaram exclusivamente clusters do Amazon Redshift ou Contas da AWS.

Console

Você deve optar explicitamente por desativar ou ativar o compartilhamento de sua unidade de compartilhamento de dados para clusters com acesso público.

- Selecione Adicionar namespaces à unidade de compartilhamento de dados. Os namespaces são um identificador exclusivo global (GUID) do cluster do Amazon Redshift.

- Escolha Add Contas da AWS (Adicionar) à unidade de compartilhamento de dados. As Contas da AWS especificadas devem ter permissões de acesso à unidade de compartilhamento de dados.

SQL

Com o SQL, o administrador concede uso na unidade de compartilhamento de dados a um namespace específico na conta. Você pode encontrar o ID do namespace como parte do ARN na página de detalhes do cluster, na página de detalhes do namespace do Amazon Redshift sem servidor ou executando o comando `SELECT current_namespace;`. Para obter mais informações, consulte CURRENT_NAMESPACE.

```
GRANT USAGE ON DATASHARE my_datashare TO NAMESPACE '86b5169f-012a-234b-9fbb-
e2e24359e9a8';
```

A seguir está um exemplo de como conceder o uso de uma unidade de compartilhamento de dados a uma Conta da AWS.

```
GRANT USAGE ON DATASHARE salesshare TO ACCOUNT '123456789012';
```

Veja a seguir um exemplo de como conceder o uso de uma unidade de compartilhamento de dados a uma conta do Lake Formation.

```
GRANT USAGE ON DATASHARE salesshare TO ACCOUNT '123456789012' VIA DATA CATALOG;
```

Autorizar uma unidade de compartilhamento de dados no Amazon Redshift

Com o Amazon Redshift, é possível controlar o acesso das unidades de compartilhamento de dados autorizando consumidores específicos. Com unidades de compartilhamento de dados, é possível compartilhar dados em tempo real entre clusters do Amazon Redshift na mesma conta ou em contas da AWS diferentes, proporcionando uma maneira integrada de distribuir e consumir dados analíticos. As seções a seguir apresentam instruções detalhadas sobre como autorizar e revogar o acesso do consumidor às unidades de compartilhamento de dados no Amazon Redshift.

> **Note**
>
> Se você estiver adicionando um namespace como um consumidor de dados, não precisará executar a autorização. Para autorizar uma unidade de compartilhamento de dados, deve

haver pelo menos um consumidor de dados adicionado à unidade de compartilhamento de dados.

Console

Como administrador de produtor no console, você pode escolher os consumidores de dados aos quais conceder autorização para acessar as unidades de compartilhamento de dados ou remover a autorização. Consumidores autorizados de dados recebem notificações para tomar ações em datashares. Se você estiver adicionando um namespace como um consumidor de dados, não precisará executar a autorização.

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. No menu de navegação, escolha Datashares (Unidades de compartilhamento de dados). Aqui, você pode consultar uma lista chamada Consumidores de unidades de compartilhamento de dados. Escolha um ou mais clusters consumidores que deseja autorizar. Em seguida, escolha Authorize (Autorizar).
3. A caixa de diálogo Autorizar conta é exibida. Você pode escolher um dentre alguns tipos de autorização.

   - Somente leitura em [nome do cluster ou nome do grupo de trabalho]: isso significa que nenhuma permissão de gravação está disponível para o consumidor, mesmo que o criador da unidade de compartilhamento de dados tenha concedido permissões de gravação.
   - Leitura e gravação em [nome do cluster ou nome do grupo de trabalho]: isso significa que todas as permissões concedidas pelo criador, inclusive permissões de gravação, estão disponíveis no consumidor.
4. Escolha Salvar.

Você também pode autorizar AWS Data Exchange como consumidor.

1. Se você selecionou Publicar no AWS Glue Data Catalog ao criar a unidade de compartilhamento de dados, só poderá conceder autorização para a unidade de compartilhamento de dados a uma conta do Lake Formation.

   Para a unidade de compartilhamento de dados AWS Data Exchange, só é possível autorizar uma unidade de compartilhamento de dados de cada vez.

Ao autorizar uma unidade de compartilhamento de dados do AWS Data Exchange, você está compartilhando a unidade de compartilhamento de dados com o serviço AWS Data Exchange e permitindo que o AWS Data Exchange gerencie o acesso à unidade de compartilhamento de dados em seu nome. O AWS Data Exchange permite o acesso aos consumidores adicionando contas de consumidores como consumidores de dados à unidade de compartilhamento de dados do AWS Data Exchange quando eles assinam os produtos. O AWS Data Exchange não tem acesso de leitura à unidade de compartilhamento de dados.

2. Escolha Salvar.

Depois que os consumidores de dados são autorizados, eles podem acessar objetos de datashare e criar um banco de dados do consumidor para consultar os dados.

API

O administrador de segurança produtor determina o seguinte:

- Se a conta pode ter ou não acesso à unidade de compartilhamento de dados.
- Se uma conta tiver acesso à unidade de compartilhamento de dados, se essa conta tem ou não permissões de gravação.

As seguintes permissões IAM são necessárias para autorizar uma unidade de compartilhamento de dados.

redshift:AuthorizeDataShare

Você pode autorizar o uso e as gravações usando uma chamada à CLI ou com a API:

```
authorize-data-share
--data-share-arn <value>
--consumer-identifier <value>
[--allow-writes | --no-allow-writes]
```

Para obter mais informações sobre o comando, consulte authorize-data-share.

O identificador do consumidor pode ser:

- Um ID da conta da AWS de doze dígitos.
- O ARN identificador do namespace.

> **Note**
>
> As permissões de gravação não são concedidas na etapa de autorização. A autorização de uma unidade de compartilhamento de dados para gravações só permite que a conta tenha permissões de gravação concedidas pelo administrador da unidade de compartilhamento de dados. Se um administrador não permitir gravações, as únicas permissões disponíveis para o consumidor específico serão SELECT, USAGE e EXECUTE.

Você pode alterar a autorização de um consumidor da unidade de compartilhamento de dados chamando `authorize-data-share` novamente, mas com um valor diferente. A autorização anterior é substituída pela nova autorização. Por isso, se você originalmente autorizar e permitir gravações, mas reautorizar e especificar `no-allow-writes` ou simplesmente não especificar um valor, o consumidor terá as permissões revogadas.

Ações do consumidor para novas unidades de compartilhamento de dados no Amazon Redshift

Com o Amazon Redshift, é possível consumir unidades de compartilhamentos de dados de outras contas da AWS, o que permite o compartilhamento de dados e a colaboração entre contas. Com uma unidade de compartilhamento de dados, é possível compartilhar dados em tempo real de maneira segura entre clusters do Amazon Redshift, mesmo que estejam em diferentes contas da AWS. As seções a seguir apresentam etapas detalhadas para configurar o acesso, criar bancos de dados de unidades de compartilhamento de dados, conceder permissões em nível de objeto e consultar dados compartilhados.

Tópicos

- Associar uma unidade de compartilhamento de dados de uma Conta da AWS diferente no Amazon Redshift
- Criar um banco de dados de uma unidade de compartilhamento de dados no Amazon Redshift
- Conceder permissões em nível de objeto a usuários e perfis do consumidor no Amazon Redshift
- Consultar dados em uma unidade de compartilhamento de dados no Amazon Redshift

Associar uma unidade de compartilhamento de dados de uma Conta da AWS diferente no Amazon Redshift

Com o Amazon Redshift, é possível associar ou recusar unidades de compartilhamento de dados compartilhadas por outras Contas da AWS, permitindo o compartilhamento contínuo e seguro de dados entre limites organizacionais. As unidades de compartilhamento de dados são objetos de banco de dados compartilháveis que encapsulam dados de um ou mais bancos de dados do Amazon Redshift. As seções a seguir demonstram o processo de associação de unidades de compartilhamento de dados.

Console

Como administrador de consumidor, você pode associar uma ou mais unidades de compartilhamento de dados compartilhadas de outras contas com toda a sua conta da AWS ou namespaces específicos em sua conta.

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. No menu de navegação, escolha Datashares (Unidades de compartilhamento de dados). A página de lista de datashares é exibida. Selecione De outras contas.
3. Na seção Datashares from other accounts (Unidades de compartilhamento de dados de outras contas), escolha a unidade de compartilhamento de dados que deseja associar e escolha Associate (Associar). Quando a página Associar unidade de compartilhamento de dados for exibida, escolha um dos seguintes tipos de associação:
   - Selecione Conta toda da AWS para associar todos os namespaces existentes e futuros em diferentes regiões da AWS na conta da AWS à unidade de compartilhamento de dados.
   - Se a unidade de compartilhamento de dados for publicada no AWS Glue Data Catalog, você só poderá associá-la a toda a conta da AWS.
4. Aqui, você pode escolher Permissões permitidas. As opções são:
   - Somente leitura: se você escolher somente leitura, as permissões de gravação, como UPDATE ou INSERT, não estarão disponíveis para o consumidor, mesmo que essas permissões tenham sido concedidas e autorizadas no produtor.
   - Leitura e gravação: os usuários da unidade de compartilhamento de dados do consumidor terão todas as permissões, tanto de leitura quanto de gravação, concedidas e autorizadas pelo produtor.

5. Se preferir, escolha Regiões e namespaces específicos da AWS para associar uma ou mais regiões e namespaces específicos da AWS à unidade de compartilhamento de dados. Selecione Adicionar região para adicionar regiões e namespaces específicos da AWS à unidade de compartilhamento de dados. A página Adicionar região da AWS é exibida.
6. Escolha uma região da AWS.
7. Execute um destes procedimentos:
   - Selecione Adicionar todos os namespaces para adicionar todos os namespaces existentes e futuros nessa região à unidade de compartilhamento de dados.
   - Escolha Adicionar namespaces específicos para adicionar um ou mais namespaces específicos nessa região à unidade de compartilhamento de dados.
   - Selecione um ou mais namespaces e escolha Adicionar região da AWS.
8. Selecione Associar.

É possível para o produtor voltar e alterar configurações de uma autorização, o que pode afetar as configurações de associação em consumidores.

Se você estiver associando a unidade de compartilhamento de dados a uma conta do Lake Formation, acesse o console do Lake Formation para criar um banco de dados, depois defina as permissões sobre o banco de dados. Para obter mais informações, consulte Configurar permissões para unidades de compartilhamento de dados do Amazon Redshift no Guia do desenvolvedor do AWS Lake Formation. Depois que você cria um banco de dados do AWS Glue ou um banco de dados federado, é possível usar o editor de consultas v2 ou qualquer cliente SQL de sua preferência com o cluster consumidor para consultar os dados.

Depois que a unidade de compartilhamento de dados for associada, as unidades de compartilhamento de dados ficarão disponíveis.

> **Note**
>
> Você também pode alterar a associação de datashare a qualquer momento. Ao alterar a associação de regiões e namespaces específicos da AWS com toda a conta da AWS, o Amazon Redshift substitui as informações de região e namespaces específicos pelas informações da conta da AWS. Todos os namespaces e regiões da AWS na Conta da AWS então têm acesso à unidade de compartilhamento de dados.

API

> **Note**
>
> As etapas desta seção serão realizadas depois que o administrador do produtor conceder ações específicas nos objetos do banco de dados compartilhado e, se a unidade de compartilhamento de dados estiver sendo compartilhada com outra conta, o administrador de segurança do produtor autorizará o acesso.

O administrador de segurança consumidor determina o seguinte:

- Se todos os namespaces de uma conta, namespaces de regiões específicas da conta ou namespaces específicos terão acesso à unidade de compartilhamento de dados.
- Se os namespaces tiverem acesso à unidade de compartilhamento de dados, independentemente de esses namespaces terem ou não permissões de gravação.

O administrador de segurança do consumidor pode associar a unidade de compartilhamento de dados por meio do seguinte comando:

```
associate-data-share-consumer
--data-share-arn <value>
--consumer-identifier <value>
[--allow-writes | --no-allow-writes]
```

Para obter mais informações sobre o comando, consulte associate-data-share-consumer.

O administrador de segurança consumidor deve definir explicitamente `allow-writes` como verdadeiro ao associar uma unidade de compartilhamento de dados a um namespace, para permitir o uso de comandos INSERT e UPDATE. Do contrário, os usuários só poderão realizar operações de leitura, como privilégios SELECT, USAGE ou EXECUTE.

Você pode alterar a associação de um namespace da unidade de compartilhamento de dados chamando `associate-data-share-consumer` novamente, com um valor diferente. Como a associação anterior é substituída pela nova associação, se você associar e definir originalmente `allow-writes`, mas associar e especificar `no-allow-writes`, ou simplesmente não especificar um valor, o consumidor terá as permissões de gravação revogadas.

Criar um banco de dados de uma unidade de compartilhamento de dados no Amazon Redshift

Com o Amazon Redshift, é possível usar uma unidade de compartilhamento de dados para criar um banco de dados e, em seguida, consultar dados em unidades de compartilhamento de dados de clusters de produtor para acessar dados ativos com segurança sem os copiar ou transferir. As etapas a seguir abordam os detalhes da configuração de um banco de dados no ambiente do Amazon Redshift.

Console

Para poder consultar dados na unidade de compartilhamento de dados, primeiro você deve criar um banco de dados de uma unidade de compartilhamento de dados. Você pode criar apenas um banco de dados a partir de um datashare especificado.

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. No menu de navegação, escolha Clusters e o seu cluster. A página de detalhes do cluster é exibida.
3. Selecione Datashares. A lista de datashare é exibida.
4. Na seção Conjuntos de dados de outros clusters, selecione Conectar ao banco de dados. Para obter mais informações, consulte Conectar-se a um banco de dados.
5. Escolha um datashare do qual você deseja criar bancos de dados e escolha Criar banco de dados a partir do datashare. A página Criar banco de dados a partir do datashare é exibida.
6. Em Nome do banco de dados, especifique um nome do banco de dados. O nome do banco de dados deve conter de 1 a 64 caracteres alfanuméricos (somente minúsculas) e não pode ser uma palavra reservada.
7. Escolha Criar.

Depois que o banco de dados for criado, você poderá consultar dados no banco de dados ou realizar operações de gravação, se elas tiverem sido concedidas, autorizadas e associadas pelo administrador do consumidor.

API

Para compartilhar dados para fins de leitura como administrador do consumidor, siga as etapas abaixo.

1. Liste os datashares que são disponibilizados para você e visualize o conteúdo de datashares. Para obter mais informações, consulte DESC DATASHARE e SHOW DATASHARES.

   O exemplo a seguir exibe as informações de datashares de entrada de um namespace de produtor especificado. Ao executar o DESC DATASHARE como administrador do consumidor, você deve especificar a opção NAMESPACE para visualizar as unidades de compartilhamento de dados de entrada.

```
DESC DATASHARE salesshare OF NAMESPACE '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';


 producer_account  |          producer_namespace          | share_type |
 share_name | object_type |           object_name           |   include_new
-------------------+-------------------------------------+------------
+------------+-------------+---------------------------------+-----------------
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    |
 salesshare | table       | public.tickit_users_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    |
 salesshare | table       | public.tickit_venue_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    |
 salesshare | table       | public.tickit_category_redshift |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    |
 salesshare | table       | public.tickit_date_redshift     |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    |
 salesshare | table       | public.tickit_event_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    |
 salesshare | table       | public.tickit_listing_redshift  |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    |
 salesshare | table       | public.tickit_sales_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    |
 salesshare | schema      | public                          |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    |
 salesshare | view        | public.sales_data_summary_view  |
```

   Somente superusuários de cluster podem fazer isso. Você também pode usar SVV_DATASHARES para visualizar os datashares e SVV_DATAASHARE_OBJECTS para vsualizar os objetos dentro do datashare.

   O exemplo a seguir exibe os datashares de entrada em um cluster de consumidores.

```
SHOW DATASHARES LIKE 'sales%';


 share_name | share_owner | source_database | consumer_database | share_type
 | createdate | is_publicaccessible | share_acl | producer_account |
 producer_namespace
-----------+-------------+-----------------+-------------------+------------
+------------+---------------------+-----------+-----------------
+--------------------------------------
 salesshare |             |                 |                   | INBOUND
    |            |         t           |           |   123456789012   |
 13b8833d-17c6-4f16-8fe4-1a018f5ed00d
```

2. Como superusuário de banco de dados, você pode criar bancos de dados locais que façam referência às unidades de compartilhamento de dados. Para obter mais informações, consulte CREATE DATABASE.

```
CREATE DATABASE sales_db FROM DATASHARE salesshare OF NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Se você quiser um controle mais granular sobre o acesso aos objetos no banco de dados local, use a cláusula WITH PERMISSIONS ao criar o banco de dados. Com essa cláusula, é possível conceder permissões no nível do objeto para objetos no banco de dados na etapa 4.

```
CREATE DATABASE sales_db WITH PERMISSIONS FROM DATASHARE salesshare OF NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Você pode ver os bancos de dados criados a partir do datashare consultando a visualização SVV_REDSHIFT_DATABASES. Você pode se conectar diretamente a esses bancos de dados ou a um banco de dados local em seu cluster de consumidor e executar uma consulta entre os bancos de dados criados nas unidades de compartilhamento de dados.

> **Note**
>
> Você não pode criar um datashare em cima de objetos de banco de dados criados a partir de um datashare existente. No entanto, você pode copiar os dados em uma tabela separada no cluster de consumidores, executar qualquer processamento necessário e, em seguida, compartilhar os novos objetos que foram criados.

Também é possível usar o console do Amazon Redshift para criar bancos de dados a partir de datashares. Para obter mais informações, consulte Criar bancos de dados a partir de datashares.

Conceder permissões em nível de objeto a usuários e perfis do consumidor no Amazon Redshift

Como administrador de consumidor, você pode conceder permissões a usuários e perfis do consumidor em nível de objeto seguindo as etapas abaixo.

SQL

Se tiver criado o banco de dados sem WITH PERMISSIONS, você só poderá atribuir permissões em todo o banco de dados criado a partir da unidade de compartilhamento de dados para usuários e funções.

```
GRANT USAGE ON DATABASE sales_db TO Bob;
```

```
GRANT USAGE ON SCHEMA sales_schema TO ROLE Analyst_role;
```

Você também pode criar exibições de vinculação tardia sobre objetos compartilhados e usá-las para atribuir permissões detalhadas. Você também pode considerar que os clusters de produtores criem conjuntos de dados adicionais para você com os detalhes necessários.

Se você tiver criado o banco de dados com WITH PERMISSIONS, deverá atribuir permissões em nível de objeto para objetos no banco de dados compartilhado. Um usuário com apenas a permissão USAGE não poderá acessar nenhum objeto em um banco de dados criado com WITH PERMISSIONS até receber permissões adicionais no nível de objeto.

```
GRANT SELECT ON sales_db.public.tickit_sales_redshift to Bob;
```

Para obter mais informações sobre como conceder permissões com gravações em vários warehouses, consulte Gerenciar permissões para unidades de compartilhamento de dados no Amazon Redshift.

Consultar dados em uma unidade de compartilhamento de dados no Amazon Redshift

Com o Amazon Redshift, é possível consultar dados entre unidades de compartilhamento de dados de clusters do produtor para acessar com segurança dados em tempo real sem os copiar ou transferir. A seção a seguir aborda a consulta de unidades de compartilhamento de dados no ambiente Amazon Redshift.

Usuários e funções com permissões em bancos de dados consumidores e esquemas em clusters consumidores podem explorar e navegar nos metadados de quaisquer objetos compartilhados. Eles também podem explorar e navegar por objetos locais em um cluster de consumidores. Para fazer isso, eles usam drivers JDBC ou ODBC ou visualizações SVV_ALL e SVV_REFISH.

Os clusters de produtores podem ter muitos esquemas no banco de dados, tabelas e visualizações dentro de cada esquema. Os usuários do lado do consumidor podem ver apenas o subconjunto de objetos que são disponibilizados através do datashare. Esses usuários não podem ver todos os metadados do cluster de produtores. Essa abordagem ajuda a fornecer controle de segurança de metadados detalhados com compartilhamento de dados.

Você continua se conectando a bancos de dados de cluster locais. Mas agora, você também pode ler dos bancos de dados e esquemas que são criados a partir do datashare usando a notação database.schema.table de três partes. Você pode executar consultas que abrangem todos e todos os bancos de dados que são visíveis para você. Estes podem ser bancos de dados locais no cluster ou bancos de dados criados a partir dos datashares. Os clusters de consumidores não podem se conectar aos bancos de dados criados a partir dos datashares.

Você pode acessar os dados usando a qualificação completa. Para obter mais informações, consulte Exemplos de consulta entre bancos de dados.

SQL

```
SELECT * FROM sales_db.public.tickit_sales_redshift ORDER BY 1,2 LIMIT 5;

 salesid | listid | sellerid | buyerid | eventid | dateid | qtysold | pricepaid |
 commission |      saletime
---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------
       1 |      1 |    36861 |   21191 |    7872 |   1875 |       4 |    728.00 |
  109.20 | 2008-02-18 02:36:48
       2 |      4 |     8117 |   11498 |    4337 |   1983 |       2 |     76.00 |
   11.40 | 2008-06-06 05:00:16
```

```
     3 |      5 |     1616 |   17433 |    8647 |   1983 |       2 |    350.00 |
52.50 | 2008-06-06 08:26:17
     4 |      5 |     1616 |   19715 |    8647 |   1986 |       1 |    175.00 |
26.25 | 2008-06-09 08:38:52
     5 |      6 |    47402 |   14115 |    8240 |   2069 |       2 |    154.00 |
23.10 | 2008-08-31 09:17:02
```

Você só pode usar instruções SELECT em objetos compartilhados. No entanto, você pode criar tabelas no cluster de consumidores consultando os dados dos objetos compartilhados em um banco de dados local diferente.

Ações do produtor para unidades de compartilhamento de dados existentes no Amazon Redshift

Com o Amazon Redshift, é possível gerenciar unidades de compartilhamento de dados existentes para controlar o acesso aos dados em um cluster do Amazon Redshift. As seções a seguir fornecem orientações detalhadas sobre como modificar objetos, gerenciar permissões e atualizar propriedades de uma unidade de compartilhamento de dados para controlar e auditar eficazmente o acesso aos dados em seu ambiente do Amazon Redshift.

Tópicos

- Visualizar uma unidade de compartilhamento de dados no Amazon Redshift
- Editar unidades de compartilhamento de dados em sua conta no Amazon Redshift
- Remover a autorização de uma unidade de compartilhamento de dados no Amazon Redshift
- Remover objetos de unidade de compartilhamento de dados de uma unidade de compartilhamento de dados no Amazon Redshift
- Remover consumidores de dados de uma unidade de compartilhamento de dados no Amazon Redshift
- Excluir uma unidade de compartilhamento de dados criada em sua conta no Amazon Redshift

Visualizar uma unidade de compartilhamento de dados no Amazon Redshift

Você pode visualizar as unidades de compartilhamento de dados no console ou com SQL.

Console

Visualize as unidades de compartilhamento de dados na guia Unidades de compartilhamento de dados ou Clusters.

- Use a guia Unidades de compartilhamento de dados para listar as unidades de compartilhamento de dados em sua conta ou de outras contas.
  - Para visualizar os datashares criados em sua conta, escolha Na minha conta e escolha o datashare que deseja visualizar.
  - Para exibir os datashares compartilhados de outras contas, escolha De outras contas e escolha o datashare que deseja visualizar.
- Use a guia Clusters para listar as unidades de compartilhamento de dados no cluster ou de outros clusters.

  Primeiro, conecte-se a um banco de dados. Em seguida, escolha uma unidade compartilhamento de dados na seção Unidades de compartilhamento de dados de outros clusters ou Unidades de compartilhamento de dados criados no meu cluster para visualizar os detalhes.

SQL

Você pode listar as unidades de compartilhamento de dados criadas no cluster e verificar o respectivo conteúdo.

O exemplo a seguir exibe as informações de uma unidade de compartilhamento de dados chamada `salesshare`.

```
DESC DATASHARE salesshare;

 producer_account  |          producer_namespace          | share_type | share_name
 | object_type |           object_name          |   include_new
-------------------+--------------------------------------+------------+------------
+-------------+--------------------------------+-------------------
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_users_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_venue_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_category_redshift|
```

```
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_date_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_event_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_listing_redshift |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_sales_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | schema      | public                         |  t
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | view        | public.sales_data_summary_view |
```

O exemplo a seguir exibe os datashares de saída em um cluster de produtores.

```
SHOW DATASHARES LIKE 'sales%';
```

A saída será semelhante à seguinte.

```
share_name | share_owner  | source_database | consumer_database | share_type |
  createdate      | is_publicaccessible  | share_acl | producer_account |
 producer_namespace
-----------+--------------+-----------------+-------------------+------------
+---------------------+----------------------+-----------+-----------------
+--------------------------------------
salesshare |    100       | dev             |                   |  OUTBOUND
 | 2020-12-09 02:27:08 |          True        |           |   123456789012   |
 13b8833d-17c6-4f16-8fe4-1a018f5ed00d
```

Para obter mais informações, consulte DESC DATASHARE e SHOW DATASHARES.

Você também pode usar SVV_DATASHARES, SVV_DATASHARE_CONSUMERS e SVV_DATASHARE_OBJECTS para exibir os datashares, os objetos dentro do datashare e os consumidores de datashare.

Editar unidades de compartilhamento de dados em sua conta no Amazon Redshift

Você pode editar uma unidade de compartilhamento de dados criada em sua conta usando o console e com SQL.

Console

No console, siga as etapas abaixo para se conectar a um banco de dados primeiro e ver a lista de unidades de compartilhamento de dados criadas na sua conta.

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. No menu de navegação, escolha Clusters e o seu cluster. A página de detalhes do cluster é exibida.
3. Selecione Datashares.
4. Na seção Datashares criados na minha conta, escolha Conectar ao banco de dados.
5. Escolha o pipeline que você deseja editar e escolha Editar. A página de detalhes do datashare é exibida.
6. Efetue quaisquer alterações na seção Objetos do datashare ou Consumidores de dados.
7. Escolha Salvar alterações. O Amazon Redshift atualiza seu datashare com as alterações.

> **Note**
>
> Se você optar por publicar a unidade de compartilhamento de dados no AWS Glue Data Catalog, não poderá editar a configuração de publicação da unidade de compartilhamento de dados para outras contas do Amazon Redshift.

SQL

Use ALTER DATASHARE para remover objetos de datashares em qualquer ponto do datashare. Para remover um esquema, use o seguinte comando:

```
ALTER DATASHARE salesshare REMOVE SCHEMA PUBLIC;
```

Para remover uma tabela, use o seguinte comando:

```
ALTER DATASHARE salesshare REMOVE TABLE public.tickit_sales_redshift;
```

Use REVOKE USAGE ON para revogar permissões no datashare para determinados consumidores. Ele revoga as permissões USAGE em objetos dentro de uma unidade de

compartilhamento de dados e interrompe instantaneamente o acesso a todos os clusters de consumidores. Listar conjuntos de dados e as consultas de metadados, como listar bancos de dados e tabelas, não retorna os objetos compartilhados depois que o acesso é revogado. Revogue o acesso ao datashare a partir de namespaces se você não quiser mais compartilhar os dados com os consumidores.

```
REVOKE USAGE ON DATASHARE salesshare FROM NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Se você não quiser mais compartilhar os dados com os consumidores, revogue o acesso à unidade de compartilhamento de dados das Contas da AWS:

```
REVOKE USAGE ON DATASHARE salesshare FROM ACCOUNT '123456789012';
```

Remover a autorização de uma unidade de compartilhamento de dados no Amazon Redshift

Com o Amazon Redshift, você pode controlar o acesso às unidades de compartilhamento de dados revogando a autorização para consumidores específicos. Esta seção fornece instruções para revogar o acesso do consumidor às unidades de compartilhamento de dados no Amazon Redshift.

> **Note**
>
> Para remover a autorização da unidade de compartilhamento de dados, deve haver pelo menos um consumidor de dados adicionado à unidade de compartilhamento de dados.

Console

Escolha um ou mais clusters consumidores dos quais deseja remover a autorização. Em seguida, escolha Remover autorização.

Após a remoção da autorização, os consumidores de dados perdem o acesso à unidade de compartilhamento de dados imediatamente.

API

O administrador de segurança produtor determina o seguinte:

- Se a conta pode ter ou não acesso à unidade de compartilhamento de dados.

- Se uma conta tiver acesso à unidade de compartilhamento de dados, se essa conta tem ou não permissões de gravação.

As seguintes permissões do IAM são necessárias para autorizar uma unidade de compartilhamento de dados.

redshift:DeauthorizeDataShare

Você pode desautorizar o uso e as gravações usando uma chamada à CLI ou com a API:

```
deauthorize-data-share
--data-share-arn <value>
--consumer-identifier <value>
```

Para obter mais informações sobre o comando, consulte deauthorize-data-share.

Remover objetos de unidade de compartilhamento de dados de uma unidade de compartilhamento de dados no Amazon Redshift

Você pode remover um ou mais objetos de uma unidade de compartilhamento de dados usando o procedimento a seguir.

Console

Para remover um ou mais objetos de uma unidade de compartilhamento de dados no console, siga as etapas abaixo.

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. No menu de navegação, escolha Clusters e o seu cluster. A página de detalhes do cluster é exibida.
3. Selecione Datashares.
4. Na seção Datashares criados na minha conta, escolha Conectar ao banco de dados. Para obter mais informações, consulte Conectar-se a um banco de dados.
5. Escolha o pipeline que você deseja editar e escolha Editar. A página de detalhes do datashare é exibida.
6. Para remover um ou mais objetos do datashare para o datashare, siga um destes procedimentos:

- Para remover esquemas do datashare, escolha um ou mais esquemas. Em seguida, escolha Remover. O Amazon Redshift remove os esquemas especificados e todos os objetos dos esquemas especificados do datashare.
- Para remover tabelas e visualizações do datashare, escolha uma ou mais tabelas e visualizações. Em seguida, escolha Remover. Ou escolha Remover por esquema para remover todas as tabelas e visualizações nos esquemas especificados.
- Para remover funções definidas pelo usuário do datashare, escolha uma ou mais funções definidas pelo usuário. Em seguida, escolha Remover. Ou escolha Remover por esquema para remover todas as funções definidas pelo usuário nos esquemas especificados.

SQL

Use ALTER DATASHARE para remover objetos de datashares em qualquer ponto do datashare. Para remover um esquema, use o seguinte comando:

```
ALTER DATASHARE salesshare REMOVE SCHEMA PUBLIC;
```

Para remover uma tabela, use o seguinte comando:

```
ALTER DATASHARE salesshare REMOVE TABLE public.tickit_sales_redshift;
```

Remover consumidores de dados de uma unidade de compartilhamento de dados no Amazon Redshift

Você pode remover um ou mais consumidores de dados de um datashare. Os consumidores de dados podem ser namespaces que identificaram exclusivamente clusters do Amazon Redshift ou contas da AWS.

Console

Para remover um ou mais consumidores de dados de uma unidade de compartilhamento de dados no console, escolha um ou mais consumidores de dados dos IDs de namespace ou da conta da AWS. Em seguida, escolha Remover.

O Amazon Redshift remove os consumidores de dados especificados do datashare. Eles perderão imediatamente o acesso ao datashare.

SQL

Use REVOKE USAGE ON para revogar permissões no datashare para determinados consumidores. Ele revoga as permissões USAGE em objetos dentro de uma unidade de compartilhamento de dados e interrompe instantaneamente o acesso a todos os clusters de consumidores. Listar conjuntos de dados e as consultas de metadados, como listar bancos de dados e tabelas, não retorna os objetos compartilhados depois que o acesso é revogado. Revogue o acesso ao datashare a partir de namespaces se você não quiser mais compartilhar os dados com os consumidores.

```
REVOKE USAGE ON DATASHARE salesshare FROM NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Excluir uma unidade de compartilhamento de dados criada em sua conta no Amazon Redshift

Você pode excluir uma unidade de compartilhamento de dados criada na sua conta usando o console ou com SQL.

Console

Para excluir uma unidade de compartilhamento de dados criada na sua conta usando o console, primeiro conecte-se a um banco de dados para ver a lista de unidades de compartilhamento de dados criadas na conta.

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. No menu de navegação, escolha Clusters e o seu cluster. A página de detalhes do cluster é exibida.
3. Selecione Datashares. A lista de datashare é exibida.
4. Na seção Datashares criados na minha conta, escolha Conectar ao banco de dados.
5. Escolha um ou mais datashares que você deseja excluir e escolha Excluir. A página Excluir datashares será exibida.

   A exclusão de uma unidade de compartilhamento de dados compartilhada com o Lake Formation não remove automaticamente as permissões associadas no Lake Formation. Para removê-las, acesse o console do Lake Formation.
6. Digite Excluir para confirmar a exclusão dos datashares especificados.

7. Escolha Excluir.

Depois que os datashares são excluídos, os consumidores de datashare perdem acesso aos conjuntos de dados.

SQL

Você pode usar SQL para excluir os objetos de unidade de compartilhamento de dados a qualquer momento usando DROP DATASHARE. Superusuários de cluster e proprietários de datashare podem descartar datashares.

O exemplo a seguir descarta uma unidade de compartilhamento de dados chamada `salesshare`.

```
DROP DATASHARE salesshare;
```

Ações do consumidor para unidades de compartilhamento de dados existentes no Amazon Redshift

Com o Amazon Redshift, é possível gerenciar unidades de compartilhamento de dados existentes para controlar o acesso aos dados em um cluster do Amazon Redshift. As seções a seguir fornecem orientações passo a passo sobre como gerenciar unidades de compartilhamento de dados no ambiente do Amazon Redshift como administrador de consumidor.

Tópicos

- Gerenciar permissões para unidades de compartilhamento de dados no Amazon Redshift
- Remover a associação de uma unidade de compartilhamento de dados de consumidores de dados em uma Conta da AWS diferente no Amazon Redshift
- Recusar uma unidade de compartilhamento de dados de uma Conta da AWS diferente no Amazon Redshift

Gerenciar permissões para unidades de compartilhamento de dados no Amazon Redshift

Como administrador de produtor, você mantém o controle dos conjuntos de dados que está compartilhando. Você pode adicionar novos objetos ao datashare ou removê-los de lá. Você também pode conceder ou revogar acesso a unidades de compartilhamento de dados como um todo para os clusters de consumidores, contas da AWS ou regiões da AWS. Quando as permissões são

revogadas, os clusters de consumidor perdem imediatamente o acesso aos objetos compartilhados e deixam de visualizá-los na lista de unidades de compartilhamento de dados INBOUND no SVV_DATASHARES.

O exemplo a seguir cria a unidade de compartilhamento de dados `salesshare` e adiciona o esquema `public` e a tabela `public.tickit_sales_redshift` a `salesshare`. Ele também concede permissões de uso no `salesshare` para o namespace especificado.

```
CREATE DATASHARE salesshare;

ALTER DATASHARE salesshare ADD SCHEMA public;

ALTER DATASHARE salesshare ADD TABLE public.tickit_sales_redshift;

GRANT USAGE ON DATASHARE salesshare TO NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Para CREATE DATASHARE, superusuários e proprietários de bancos de dados podem criar conjuntos de dados. Para obter mais informações, consulte CREATE DATASHARE. Para ALTER DATASHARE, o proprietário do datashare com as permissões necessárias nos objetos de datashare a serem adicionados ou removidos pode alterar o datashare. Para mais informações, consulte ALTER DATASHARE.

Como administrador de produtores, quando você solta um datashare, ele deixa de ser listado em clusters de consumidores. Os bancos de dados e referências de esquema criados no cluster de consumidor a partir do datashare descartado continuam a existir sem objetos neles. O administrador do consumidor deve excluir esses bancos de dados manualmente.

Do lado do consumidor, um administrador de consumidor pode determinar quais usuários e grupos devem ter acesso aos dados compartilhados criando um banco de dados da unidade de compartilhamento de dados. Dependendo das opções escolhidas ao criar o banco de dados, você pode controlar o acesso a ele da maneira a seguir. Para obter mais informações sobre como criar um banco de dados a partir de uma unidade de compartilhamento de dados, consulte CREATE DATABASE.

Consulte mais informações sobre como configurar uma unidade de compartilhamento de dados e ler dados de um consumidor em Compartilhar o acesso de leitura aos dados em uma Conta da AWS.

Criação do bancos de dados sem a cláusula WITH PERMISSIONS

Um administrador pode controlar o acesso no nível do banco de dados ou do esquema. Para controlar o acesso no nível do esquema, o administrador deve criar um esquema externo a partir do banco de dados do Amazon Redshift criado com base na unidade de compartilhamento de dados.

O exemplo a seguir concede permissões para acessar uma tabela compartilhada no nível de banco de dados e no nível de esquema.

```
GRANT USAGE ON DATABASE sales_db TO Bob;

CREATE EXTERNAL SCHEMA sales_schema FROM REDSHIFT DATABASE sales_db SCHEMA 'public';

GRANT USAGE ON SCHEMA sales_schema TO ROLE Analyst_role;
```

Para restringir ainda mais o acesso, você pode criar exibições sobre objetos compartilhados, expondo apenas os dados necessários. Você pode acabar usando essas exibições para dar acesso aos usuários e às funções.

Depois que os usuários tiverem acesso ao banco de dados ou esquema, eles terão acesso a todos os objetos compartilhados nesse banco de dados ou esquema.

Criação do bancos de dados com a cláusula WITH PERMISSIONS

Depois de conceder direitos de uso no banco de dados ou no esquema, um administrador poderá controlar ainda mais o acesso usando o mesmo processo de concessão da permissão que usaria em um banco de dados ou esquema local. Sem permissões de objeto individual, os usuários não podem acessar nenhum objeto no banco de dados compartilhado ou no esquema, mesmo depois de receberem a permissão USAGE.

O exemplo a seguir concede permissões para ter acesso a uma tabela compartilhada no nível do banco de dados.

```
GRANT USAGE ON DATABASE sales_db TO Bob;
GRANT USAGE FOR SCHEMAS IN DATABASE sales_db TO Bob;
GRANT SELECT ON sales_db.public.tickit_sales_redshift TO Bob;
```

Depois que tiverem recebido acesso ao banco de dados ou ao esquema, os usuários continuarão precisando receber as permissões relevantes para qualquer objeto no banco de dados ou esquema que você queira que eles acessem.

Compartilhamento detalhado usando WITH PERMISSIONS

Você pode utilizar o compartilhamento detalhado usando WITH PERMISSIONS para permitir que clusters ou grupos de trabalho sem servidor consultem a unidade de compartilhamento de dados. Esse processo pressupõe que a unidade de compartilhamento de dados tenha origem em outro cluster ou no namespace do Amazon Redshift sem servidor em sua conta ou esteja vindo de outra conta e tenha sido associada ao namespace que você está usando.

1. O administrador do banco de dados consumidor pode criar um banco de dados a partir da unidade de compartilhamento de dados.

   ```
   CREATE DATABASE my_ds_db [WITH PERMISSIONS] FROM DATASHARE my_datashare OF
    NAMESPACE 'abc123def';
   ```

   Se criar um banco de dados WITH PERMISSIONS, você poderá conceder permissões detalhadas em objetos de unidade de compartilhamento de dados para usuários e perfis diferentes. Sem isso, todos os usuários e funções com permissão USAGE no banco de dados da unidade de compartilhamento de dados recebem todas as permissões em todos os objetos dentro do banco de dados da unidade de compartilhamento de dados.
2. Aqui está como conceder permissões a um usuário ou função do banco de dados do Redshift. Você deve ter uma conexão com um banco de dados local para executar essas instruções: Você não poderá executar essas instruções se executar um comando USE no banco de dados da unidade de compartilhamento de dados antes de executar as instruções de concessão.

   ```
   GRANT USAGE ON DATABASE my_ds_db TO ROLE data_eng;
   GRANT CREATE, USAGE ON SCHEMA my_ds_db.my_shared_schema TO ROLE data_eng;
   GRANT ALL ON ALL TABLES IN SCHEMA my_ds_db.my_shared_schema TO ROLE data_eng;

   GRANT USAGE ON DATABASE my_ds_db TO bi_user;
   GRANT USAGE ON SCHEMA my_ds_db.my_shared_schema TO bi_user;
   GRANT SELECT ON my_ds_db.my_shared_schema.table1 TO bi_user;
   ```

Remover a associação de uma unidade de compartilhamento de dados de consumidores de dados em uma Conta da AWS diferente no Amazon Redshift

Com o Amazon Redshift, é possível remover a associação de unidades de compartilhamento de dados compartilhadas por outras Contas da AWS. Unidades de compartilhamento de dados são objetos de banco de dados compartilháveis que encapsulam dados de um ou mais bancos de

dados do Redshift. As seções a seguir demonstram o processo de desassociação de unidades de compartilhamento de dados no ambiente do Redshift.

Console

Como administrador de consumidor, você pode remover a associação de unidades de compartilhamento de dados de consumidores de dados no console.

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. No menu de navegação, escolha Datashares (Unidades de compartilhamento de dados). A página de lista de datashares é exibida.
3. Selecione De outras contas.
4. Na seção Conjuntos de dados de outras contas, escolha o datashare para remover a associação dos consumidores de dados.
5. Na seção Consumidores de dados, escolha um ou mais consumidores de dados para remover associação. Em seguida, escolha Remover associação.
6. Quando a página Remover associação aparecer, escolha Remover associação.

Após a remoção da associação, os consumidores de dados perderão o acesso ao datashare. Você pode fazer alterações na associação de consumidores de dados a qualquer momento.

SQL

> **Note**
>
> As etapas desta seção serão realizadas depois que o administrador do produtor conceder ações específicas nos objetos do banco de dados compartilhado e, se a unidade de compartilhamento de dados estiver sendo compartilhada com outra conta, o administrador de segurança do produtor autorizará o acesso.

O administrador de segurança do consumidor pode desassociar a unidade de compartilhamento de dados com o seguinte comando:

```
disassociate-data-share-consumer
--data-share-arn <value>
```

Para obter mais informações sobre o comando, consulte disassociate-data-share-consumer.

Recusar uma unidade de compartilhamento de dados de uma Conta da AWS diferente no Amazon Redshift

Com o Amazon Redshift, você pode recusar unidades de compartilhamento de dados compartilhadas por outras contas AWS, permitindo o compartilhamento de dados seguro e integrado entre limites organizacionais. As unidades de compartilhamento de dados são objetos de banco de dados compartilháveis que encapsulam dados de um ou mais bancos de dados do Amazon Redshift.

Como administrador de consumidor, você pode rejeitar a unidade de compartilhamento de dados cujo estado esteja disponível ou ativo. Depois que você rejeitar uma unidade de compartilhamento de dados, os usuários do cluster consumidor perderão acesso a ela. O Amazon Redshift não retornará a unidade de compartilhamento de dados rejeitada se você chamar a operação de API `DescribeDataSharesForConsumer`. Se o administrador do produtor executar a operação de API `DescribeDataSharesForProducer`, ele verá que a unidade de compartilhamento de dados foi rejeitada. Depois que a unidade de compartilhamento de dados é rejeitada, o administrador do produtor pode autorizar a unidade de compartilhamento de dados para um cluster de consumidor novamente, e o administrador do consumidor pode escolher associar a conta da AWS à unidade de compartilhamento de dados ou rejeitá-la.

Se sua conta da AWS tiver uma associação a uma unidade de compartilhamento de dados e uma associação pendente a uma unidade de compartilhamento de dados gerenciada pelo Lake Formation, rejeitar a associação à unidade de compartilhamento de dados gerenciada pelo Lake Formation também rejeitará a unidade de compartilhamento de dados original. Para rejeitar uma associação específica, o administrador do produtor pode remover a autorização de uma unidade de compartilhamento de dados especificada. Essa ação não afeta outras unidades de compartilhamento de dados.

Para rejeitar uma unidade de compartilhamento de dados, use o console da AWS, a operação de API `RejectDataShare` ou `reject-datashare` na AWS CLI.

Console

Para rejeitar uma unidade de compartilhamento de dados usando o Console da AWS, siga as etapas abaixo.

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.

2. No menu de navegação, escolha Unidades de compartilhamento de dados.
3. Selecione De outras contas.
4. Na seção Datashares de outras contas, escolha o datashare que deseja recusar. Quando a página Decline datashare (Recusar unidade de compartilhamento de dados) for exibida, escolha Decline (Recusar).

Após recusar os datashares, você não pode reverter a alteração. O Amazon Redshift remove as unidades de compartilhamento de dados da lista. Para ver a unidade de compartilhamento de dados novamente, o administrador do produtor deve autorizar novamente.

CLI

Para rejeitar uma unidade de compartilhamento de dados, o administrador usa o seguinte comando:

```
reject-data-share
--data-share-arn <value>
```

Para obter mais informações sobre o comando, consulte reject-data-share.

Conceitos básicos do compartilhamento de dados por meio do AWS CloudFormation no Amazon Redshift

Você pode automatizar a configuração do compartilhamento de dados usando uma pilha do AWS CloudFormation, que provisiona recursos da AWS. A pilha do CloudFormation configura o compartilhamento de dados entre dois clusters do Amazon Redshift na mesma conta da AWS. Assim, é possível iniciar o compartilhamento de dados sem executar instruções SQL para provisionar seus recursos.

A pilha cria uma unidade de compartilhamento de dados no cluster designado por você. A unidade de compartilhamento de dados contém uma tabela e uma amostra de dados somente para leitura. Esses dados podem ser lidos por outro cluster do Amazon Redshift.

Se você quiser começar a compartilhar dados em uma conta da AWS executando instruções SQL para configurar uma unidade de compartilhamento de dados e conceder permissões, sem usar o CloudFormation, consulte Compartilhar o acesso de leitura aos dados em uma Conta da AWS.

Antes de executar a pilha do CloudFormation de compartilhamento de dados, é necessário estar conectado a um usuário que tenha permissão para criar uma função do IAM e uma função do Lambda. Você também precisa ter dois clusters do Amazon Redshift na mesma conta. Use um, o produtor, para compartilhar os dados de exemplo, e o outro, consumidor, para lê-los. O principal requisito desses clusters é que cada um use nós RG ou RA3. Para requisitos adicionais, consulte Considerações sobre compartilhamento de dados no Amazon Redshift.

Consulte mais informações sobre conceitos básicos da configuração de um cluster do Amazon Redshift em Clusters provisionados do Amazon Redshift. Para obter mais informações sobre a automação da configuração com o CloudFormation, consulte O que é o AWS CloudFormation?.

> **Important**
>
> Antes de iniciar a pilha do CloudFormation, é necessário ter dois clusters do Amazon Redshift na mesma conta e os clusters devem usar nós RG ou RA3. Cada cluster deve ter um banco de dados e um superusuário. Para obter mais informações, consulte CREATE DATABASE e superuser.

Para iniciar a pilha do CloudFormation para o compartilhamento de dados do Amazon Redshift:

1. Escolha Iniciar pilha do CFN, que direciona você ao serviço do CloudFormation no Console de gerenciamento da AWS.

   Se for solicitado, faça login.

   Inicia-se o processo de criação de pilha, fazendo referência a um arquivo de template do CloudFormation, que é armazenado no Amazon S3. O template do CloudFormation é um arquivo de texto em formato JSON que declara os recursos da AWS que compõem uma pilha. Para obter mais informações sobre modelos do CloudFormation, consulte Conheça o básico de modelos.
2. Selecione Next (Próximo) para inserir os detalhes da pilha.
3. Em Parameters (Parâmetros), para cada cluster, insira o seguinte:

   - O nome do cluster do Amazon Redshift, por exemplo **`ra3-consumer-cluster`**.
   - O nome do banco de dados; por exemplo **`dev`**
   - O nome de um usuário do banco de dados; por exemplo **`consumeruser`**

Recomendamos o uso de clusters de teste, pois a pilha cria vários objetos de banco de dados.

Escolha Próximo.

4. São exibidas as opções de pilha.

   Selecione Next (Próximo) para aceitar as configurações padrão.

5. Em Capabilities (Capacidades), escolha I acknowledge that AWS CloudFormation might create IAM resources (Estou ciente de que o pode criar recursos do IAM).
6. Selecione Criar pilha.

O CloudFormation leva cerca de 10 minutos para criar a pilha do Amazon Redshift usando o modelo e cria uma unidade de compartilhamento de dados chamada `myproducer_share`. A pilha cria a unidade de compartilhamento de dados no banco de dados especificado nos detalhes da pilha. Somente objetos desse banco de dados podem ser compartilhados.

Se ocorrer um erro durante a criação da pilha, faça o seguinte:

- Verifique se inseriu corretamente o nome do cluster, o nome do banco de dados e o nome do usuário do banco de dados de cada cluster do Redshift.
- Verifique se o cluster tem nós RG ou RA3.
- Verifique se você está conectado com um usuário que tem permissão para criar um perfil do IAM e uma função do Lambda. Para obter mais informações sobre a criação de perfis do IAM, consulte Criação de funções do IAM. Para obter mais informações sobre políticas para a criação da função Λ, consulte Desenvolvimento de função.

Consultar a unidade de compartilhamento de dados que você criou

Para usar o procedimento a seguir, verifique se você tem as permissões necessárias para executar consultas em cada cluster descrito.

Para consultar sua unidade de compartilhamento de dados:

1. Conecte-se ao cluster de produtor no banco de dados do Redshift inserido ao criar a pilha do CloudFormation usando uma ferramenta cliente, como o Editor de Consultas v2 do Amazon Redshift.
2. Consulte as unidades de compartilhamento de dados.

```
SHOW DATASHARES;

+-----------------+-------------+-----------------+------------------
+------------+------------+--------------------+----------+-----------------
+-------------------------------------+
|    share_name    | share_owner | source_database | consumer_database | share_type
 | createdate | is_publicaccessible | share_acl | producer_account |
 producer_namespace          |
+-----------------+-------------+-----------------+------------------
+------------+------------+--------------------+----------+-----------------
+-------------------------------------+
| myproducer_share | 100         | sample_data_dev | myconsumer_db     | INBOUND
  | NULL       | true               | NULL      |  producer-acct   |        your-
producer-namespace       |
+-----------------+-------------+-----------------+------------------
+------------+------------+--------------------+----------+-----------------
+-------------------------------------+
```

O comando anterior retorna o nome da unidade de compartilhamento de dados criada pela pilha, chamada `myproducer_share`. Também retorna o nome do banco de dados associado à unidade de compartilhamento de dados, ., `myconsumer_db`.

Copie o identificador de namespace do produtor para usar em uma etapa posterior.

3. Descreva os objetos na unidade de compartilhamento de dados.

```
DESC DATASHARE myproducer_share;

+-----------------+-------------------------------------+-----------
+-----------------+-------------+-------------------------------------
+-------------+
| producer_account |          producer_namespace          | share_type |
 share_name    | object_type |             object_name             | include_new |
+-----------------+-------------------------------------+-----------
+-----------------+-------------+-------------------------------------
+-------------+
|   producer-acct  |        your-producer-namespace       | OUTBOUND   |
 myproducer_share | schema      | myproducer_schema                   | true
 |
|   producer-acct  |        your-producer-namespace       | OUTBOUND   |
 myproducer_share | table       | myproducer_schema.tickit_sales      | NULL
 |
```

```
|   producer-acct  |        your-producer-namespace       | OUTBOUND    |
 myproducer_share | view        | myproducer_schema.ticket_sales_view | NULL
 |
+------------------+--------------------------------------+-----------
+------------------+-------------+-------------------------------------
+-------------+
```

Quando você descreve a unidade de compartilhamento de dados, ela retorna propriedades para tabelas e visualizações. A pilha adiciona ao banco de dados do produtor tabelas e visualizações com exemplos de dados, como `tickit_sales` e `tickit_sales_view`. Para obter mais informações sobre os bancos de dados de amostra TICKIT, consulte Banco de dados de exemplo.

Não é necessário delegar permissões na unidade de compartilhamento de dados para executar consultas. A pilha concede as permissões necessárias.

4. Conecte-se ao cluster de consumidor usando sua ferramenta cliente. Descreva a unidade de compartilhamento de dados, especificando o namespace do produtor.

```
DESC DATASHARE myproducer_share OF NAMESPACE '<namespace id>'; --specify the unique
 identifier for the producer namespace

+------------------+--------------------------------------+-----------
+------------------+-------------+-------------------------------------
+-------------+
| producer_account |          producer_namespace          | share_type |
 share_name    | object_type |             object_name             | include_new |
+------------------+--------------------------------------+-----------
+------------------+-------------+-------------------------------------
+-------------+
|   producer-acct  |        your-producer-namespace       | INBOUND    |
 myproducer_share | schema      | myproducer_schema                   | NULL
 |
|   producer-acct  |        your-producer-namespace       | INBOUND    |
 myproducer_share | table       | myproducer_schema.tickit_sales      | NULL
 |
|   producer-acct  |        your-producer-namespace       | INBOUND    |
 myproducer_share | view        | myproducer_schema.ticket_sales_view | NULL
 |
+------------------+--------------------------------------+-----------
+------------------+-------------+-------------------------------------
+-------------+
```

5. É possível consultar tabelas na unidade de compartilhamento de dados especificando o banco de dados e o esquema da unidade de compartilhamento de dados. Para obter mais informações, consulte Exemplos de consulta entre bancos de dados. As consultas a seguir retornam dados de vendas e vendedores da tabela SALES no banco de dados de amostra TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

```
SELECT * FROM myconsumer_db.myproducer_schema.tickit_sales_view;

+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
| salesid | listid | sellerid | buyerid | eventid | dateid | qtysold | pricepaid |
 commission |      saletime       |
+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
|       1 |      1 |    36861 |   21191 |    7872 |   1875 |       4 |       728 |
     109.2 | 2008-02-18 02:36:48 |
|       2 |      4 |     8117 |   11498 |    4337 |   1983 |       2 |        76 |
      11.4 | 2008-06-06 05:00:16 |
|       3 |      5 |     1616 |   17433 |    8647 |   1983 |       2 |       350 |
      52.5 | 2008-06-06 08:26:17 |
|       4 |      5 |     1616 |   19715 |    8647 |   1986 |       1 |       175 |
     26.25 | 2008-06-09 08:38:52 |
|       5 |      6 |    47402 |   14115 |    8240 |   2069 |       2 |       154 |
      23.1 | 2008-08-31 09:17:02 |
+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
```

> **Note**
>
> A consulta é executada conforme a visualização no esquema compartilhado. Você não pode se conectar diretamente a bancos de dados criados com base em unidades de compartilhamento de dados. Eles são somente para leitura.

6. Para executar uma consulta que inclua agregações, use o exemplo a seguir.

```
SELECT * FROM myconsumer_db.myproducer_schema.tickit_sales ORDER BY 1,2 LIMIT 5;

+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
| salesid | listid | sellerid | buyerid | eventid | dateid | qtysold | pricepaid |
 commission |      saletime       |
```

```
+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
|       1 |      1 |    36861 |   21191 |    7872 |   1875 |       4 |       728 |
     109.2 | 2008-02-18 02:36:48 |
|       2 |      4 |     8117 |   11498 |    4337 |   1983 |       2 |        76 |
      11.4 | 2008-06-06 05:00:16 |
|       3 |      5 |     1616 |   17433 |    8647 |   1983 |       2 |       350 |
      52.5 | 2008-06-06 08:26:17 |
|       4 |      5 |     1616 |   19715 |    8647 |   1986 |       1 |       175 |
     26.25 | 2008-06-09 08:38:52 |
|       5 |      6 |    47402 |   14115 |    8240 |   2069 |       2 |       154 |
      23.1 | 2008-08-31 09:17:02 |
+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
```

A consulta retorna dados de vendas e de vendedor da amostra de dados do TICKIT.

Para obter mais exemplos de consultas de unidade de compartilhamento de dados, acesse Compartilhar o acesso de leitura aos dados em uma Conta da AWS.

Tipos de unidade de compartilhamento de dados no Amazon Redshift

Um datashare é a unidade de compartilhamento de dados no Amazon Redshift. Use unidades de compartilhamento de dados para compartilhar dados na mesma Conta da AWS ou em Contas da AWS diferentes. Também é possível compartilhar dados para fins de leitura em diferentes clusters do Amazon Redshift.

Cada datashare está associado a um banco de dados específico em seu cluster do Amazon Redshift.

Um administrador de produtor pode criar unidades de compartilhamento de dados e adicionar objetos de unidade de compartilhamento de dados para compartilhar dados com outros clusters, chamados de compartilhamentos de saída. Um administrador de consumidor pode receber unidades de compartilhamento de dados de outros clusters, chamados de compartilhamentos de entrada. Para obter detalhes sobre produtores e consumidores, consulte Produtores e consumidores da unidade de compartilhamento de dados.

Objetos da unidade de compartilhamento de dados são objetos de bancos de dados específicos em um cluster que os administradores de produtor podem adicionar a unidades de compartilhamento

de dados para serem compartilhados com consumidores de dados. Os objetos do datashare são somente leitura para consumidores de dados. Exemplos de objetos do datashare são tabelas, exibições e funções definidas pelo usuário. Você pode adicionar objetos do datashare a conjuntos de dados enquanto cria conjuntos de dados ou edita um datashare a qualquer momento.

O compartilhamento de dados continua funcionando quando os clusters são redimensionados ou quando o cluster de produtor está pausado.

Existem diferentes tipos de unidade de compartilhamento de dados: unidades de compartilhamento de dados padrão, unidades de compartilhamento de dados do AWS Data Exchange e unidades de compartilhamento de dados gerenciadas pelo AWS Lake Formation. As páginas a seguir apresentam uma visão geral de cada um deles.

Unidades de compartilhamento de dados comuns

Com as unidades de compartilhamento de dados comuns, é possível compartilhar dados entre clusters provisionados, grupos de trabalho sem servidor, zonas de disponibilidade, Contas da AWS e Regiões da AWS. Você pode compartilhar entre tipos de cluster, bem como entre clusters provisionados e o Amazon Redshift sem servidor.

Para compartilhar dados, observe o seguinte cluster provisionado, namespace sem servidor e identificadores Conta da AWS:

- Namespaces provisionados são identificadores que distinguem clusters provisionados do Amazon Redshift. Um identificador exclusivo global (GUID) do namespace é criado automaticamente durante a criação do cluster provisionado e anexado ao cluster. O nome do recurso da Amazon (ARN) está no formato arn:{partition}:redshift:{region}:{account-id}:namespace:{namespace-guid}. Você pode ver o namespace de um cluster provisionado na página de detalhes do cluster no console do Amazon Redshift.

  No fluxo de trabalho de compartilhamento de dados, o valor do GUID do namespace e o ARN do namespace são usados para compartilhar dados com clusters na Conta da AWS. Também é possível encontrar o namespace para o cluster atual usando a função `current_namespace`.
- Namespaces sem servidor são identificadores que identificam o Amazon Redshift Serverless. Um identificador exclusivo global (GUID) do namespace é criado automaticamente durante a criação do Amazon Redshift Serverless e anexado à instância. O ARN do namespace sem servidor está no formato arn:{partition}:redshift-serverless:{region}:{account-id}:namespace/{namespace-guid}.
- As Contas da AWS podem ser consumidores de unidades de compartilhamento de dados, e cada uma é representada por um ID de Conta da AWS de 12 dígitos.

Para unidades de compartilhamento de dados padrão, considere o seguinte:

- Quando um cluster de produtores é excluído, o Amazon Redshift exclui os conjuntos de dados criados pelo cluster de produtores. Quando um cluster de produtor é feito backup e restaurado, os conjuntos de dados criados ainda persistem no cluster restaurado. Se a restauração for executada no mesmo namespace sem servidor, as permissões de unidade de compartilhamento de dados poderão ser preservadas. Consulte Considerações sobre o compartilhamento de dados na restauração do Amazon Redshift sem servidor para saber quais cenários são compatíveis. No entanto, as permissões de unidade de compartilhamento de dados concedidas a clusters de consumidor não serão mais válidas no cluster restaurado. Conceda novamente permissões de uso de conjuntos de dados aos clusters de consumidores desejados. O banco de dados do consumidor no cluster de consumidores aponta para o datashare do cluster original onde o snapshot é obtido. Para consultar os dados compartilhados do cluster restaurado, o administrador do consumidor cria um banco de dados diferente. Como alternativa, o administrador pode descartar e recriar um banco de dados de consumidor existente para usar a unidade de compartilhamento de dados do cluster recém-restaurado.
- Quando um cluster de consumidor é excluído e restaurado de um snapshot, o acesso anterior compartilhado a esse cluster não fica mais válido e visível. Se o acesso às unidades de compartilhamento de dados ainda for necessário no cluster de consumidor restaurado, o administrador do produtor deverá conceder novamente o uso de unidades de compartilhamento de dados ao cluster de consumidor restaurado. O administrador do consumidor deve remover os bancos de dados de consumidor obsoletos criados com base em unidades de compartilhamento de dados inativas. Em seguida, o administrador deve recriar o banco de dados consumidor com base na unidade de compartilhamento de dados, depois que o produtor concedeu novamente as permissões. Como o GUID do namespace em um cluster restaurado é diferente do GUID do cluster original, conceda novamente as permissões da unidade de compartilhamento de dados quando o cluster de consumidor ou produtor for restaurado do backup.

AWS Data ExchangeUnidades de compartilhamento de dados do

Você pode usar unidades de compartilhamento de dados do AWS Data Exchange para gerenciar o faturamento do compartilhamento de dados do Amazon Redshift.

Uma unidade de compartilhamento de dados do AWS Data Exchange é uma unidade de licenciamento para compartilhar dados por meio do AWS Data Exchange. A AWS gerencia todo o faturamento e os pagamentos associados a assinaturas do AWS Data Exchange e o uso do compartilhamento de dados do Amazon Redshift. Os provedores de dados aprovados podem

adicionar unidades de compartilhamento de dados do AWS Data Exchange a produtos AWS Data Exchange. Quando os clientes se inscrevem em um produto com unidades de compartilhamento de dados do AWS Data Exchange, eles têm acesso às unidades de compartilhamento de dados no produto.

O AWS Data Exchange for Amazon Redshift torna a licença de acesso a seus dados do Amazon Redshift conveniente por meio do AWS Data Exchange. Quando um cliente assina um produto com unidades de compartilhamento de dados do AWS Data Exchange, o AWS Data Exchange adiciona automaticamente o cliente como consumidor de dados em todas as unidades de compartilhamento de dados do AWS Data Exchange incluídas no produto. As faturas são geradas automaticamente, e os pagamentos são coletados de modo centralizado e desembolsados automaticamente pelo AWS Marketplace Entitlement Service.

Os provedores podem licenciar dados no Amazon Redshift em nível detalhado, como esquemas, tabelas, visualizações e funções definidas pelo usuário. Você pode usar a mesma unidade de compartilhamento de dados do AWS Data Exchange em vários produtos do AWS Data Exchange. Todos os objetos adicionados à unidade de compartilhamento de dados do AWS Data Exchange estão disponíveis para os consumidores. Os produtores podem visualizar todas as unidades de compartilhamento de dados do AWS Data Exchange gerenciadas pelo AWS Data Exchange em nome deles usando operações de API do Amazon Redshift, comandos SQL e o console do Amazon Redshift. Clientes que assinam uma unidade de compartilhamento de dados do AWS Data Exchange de um produto têm acesso somente para leitura aos objetos nas unidades de compartilhamento de dados.

Os clientes que desejam consumir dados de produtores de terceiros podem navegar no catálogo do AWS Data Exchange para descobrir e assinar conjuntos de dados no Amazon Redshift. Depois que a assinatura do AWS Data Exchange está ativa, eles podem criar um banco de dados com base na unidade de compartilhamento de dados em seu cluster e consultar os dados no Amazon Redshift.

Como funcionam as unidades de compartilhamento de dados do AWS Data Exchange

Gerenciar unidades de compartilhamento de dados do AWS Data Exchange como administrador produtor

Se você for produtor de dados (também conhecido como provedor no AWS Data Exchange), poderá criar unidades de compartilhamento de dados do AWS Data Exchange que se conectam aos bancos de dados do Amazon Redshift. Para adicionar unidades de compartilhamento de dados do AWS Data Exchange para produtos no AWS Data Exchange, é necessário ser um fornecedor do AWS Data Exchange registrado.

Para obter mais informações sobre como usar unidades de compartilhamento de dados do AWS Data Exchange, consulte Compartilhar dados licenciados do Amazon Redshift no AWS Data Exchange.

Usar unidades de compartilhamento de dados do AWS Data Exchange como consumidor com uma assinatura do AWS Data Exchange ativa

Se você for consumidor com uma assinatura do AWS Data Exchange ativa (também conhecido como assinante do AWS Data Exchange), poderá navegar no catálogo do AWS Data Exchange no console do AWS Data Exchange para descobrir produtos que contenham unidades de compartilhamento de dados do AWS Data Exchange.

Depois de assinar um produto que contenha unidades de compartilhamento de dados do AWS Data Exchange, crie um banco de dados com base na unidade de compartilhamento de dados dentro do cluster. Em seguida, você pode consultar os dados no Amazon Redshift diretamente sem extrair, transformar nem carregar os dados.

Para obter mais informações sobre como usar unidades de compartilhamento de dados do AWS Data Exchange, consulte Compartilhar dados licenciados do Amazon Redshift no AWS Data Exchange.

Para unidades de compartilhamento de dados do AWS Data Exchange, considere o seguinte:

- Quando um cluster de produtores é excluído, o Amazon Redshift exclui os conjuntos de dados criados pelo cluster de produtores. Quando um cluster de produtor é feito backup e restaurado, os conjuntos de dados criados ainda persistem no cluster restaurado. Para que os assinantes de dados possam continuar acessando os dados, crie as unidades de compartilhamento de dados do AWS Data Exchange novamente e publique-as nos conjuntos de dados do produto. O banco de dados do consumidor no cluster de consumidores aponta para o datashare do cluster original onde o snapshot é obtido. Para consultar os dados compartilhados do cluster restaurado, o administrador do consumidor cria um banco de dados diferente ou descarta e recria um banco de dados de consumidor existente para usar a unidade de compartilhamento de dados do AWS Data Exchange recém-criada do cluster recém-restaurado.
- Recomendamos não excluir seu cluster se tiver alguma unidade de compartilhamento de dados do AWS Data Exchange. Executar esse tipo de alteração pode violar os termos do produto de dados no AWS Data Exchange.

Produtores e consumidores da unidade de compartilhamento de dados

Produtores de dados (também conhecidos como produtores de unidade de compartilhamento de dados ou produtores de compartilhamento de dados) são clusters dos quais você deseja compartilhar dados. Administradores de produtor e proprietários de bancos de dados podem criar unidades de compartilhamento de dados usando o comando CREATE DATASHARE. Você pode adicionar objetos, como esquemas, tabelas, visualizações e funções definidas pelo usuário (UDFs) de SQL, de um banco de dados que você deseja que o cluster de produtor compartilhe com clusters de consumidores para fins de leitura.

Produtores de dados (também conhecidos como provedores no AWS Data Exchange) para unidades de compartilhamento de dados do AWS Data Exchange podem licenciar dados pelo AWS Data Exchange. Os provedores aprovados podem adicionar unidades de compartilhamento de dados AWS Data Exchange a produtos AWS Data Exchange.

Quando um cliente assina um produto com unidades de compartilhamento de dados AWS Data Exchange, o AWS Data Exchange adiciona automaticamente o cliente como consumidor de dados em todas as unidades de compartilhamento de dados AWS Data Exchange incluídas com o produto. O AWS Data Exchange também remove todos os clientes das unidades de compartilhamento de dados AWS Data Exchange quando a assinatura termina. O AWS Data Exchange também gerencia automaticamente o faturamento, a cobrança de pagamentos e a distribuição de pagamentos dos produtos pagos com unidades de compartilhamento de dados AWS Data Exchange. Para obter mais informações, consulte AWS Data ExchangeUnidades de compartilhamento de dados do. Para se registrar como um provedor de dados do AWS Data Exchange, consulte Primeiros passos como provedor.

Consumidores de dados (também conhecidos como consumidores de compartilhamento de dados ou consumidores de datashare) são clusters que recebem conjuntos de dados de clusters de produtores.

Os clusters do Amazon Redshift que compartilham dados podem estar na mesma conta ou em Contas da AWS diferentes ou em Regiões da AWS diferentes, para que você possa compartilhar dados entre organizações e colaborar com outras partes. Os administradores de consumidor recebem as unidades de compartilhamento de dados para as quais têm permissão de uso e revisam o conteúdo de cada unidade de compartilhamento de dados. Para consumir dados compartilhados, o administrador do consumidor cria um banco de dados do Amazon Redshift com base na unidade de compartilhamento de dados. O administrador acaba atribuindo permissões para o banco de dados a usuários e funções no cluster consumidor. Depois que as permissões são concedidas, os usuários e

as funções podem listar os objetos compartilhados como parte das consultas de metadados padrão, além dos dados locais no cluster consumidor. Eles podem começar a consultar imediatamente.

Se você for um consumidor com um assinatura do AWS Data Exchange ativa (também conhecido como assinante do AWS Data Exchange), poderá encontrar, assinar e consultar dados detalhados e atualizados no Amazon Redshift sem necessidade de extrair, transformar e carregar dados. Para obter mais informações, consulte AWS Data ExchangeUnidades de compartilhamento de dados do.

AWS Lake FormationUnidades de compartilhamento de dados gerenciadas pelo

Com o Amazon Redshift, é possível acessar e compartilhar dados em tempo real entre contas da AWS e clusters do Amazon Redshift por meio de unidades de compartilhamento de dados gerenciadas pelo AWS Lake Formation. As unidades de compartilhamento de dados do AWS Lake Formation permitem que os provedores de dados compartilhem dados em tempo real com segurança entre seu data lake do Amazon S3 e qualquer consumidor, inclusive com outras contas da AWS e clusters do Amazon Redshift.

Usando o AWS Lake Formation, você pode definir e aplicar centralmente as permissões de acesso por banco de dados, tabela, coluna e linha das unidades de compartilhamento de dados do Amazon Redshift e restringir o acesso dos usuários a objetos em uma unidade de compartilhamento de dados. Ao compartilhar dados pelo Lake Formation, você pode definir permissões no Lake Formation e aplicar essas permissões a qualquer unidade de compartilhamento de dados e seus objetos. Por exemplo, se você tiver uma tabela contendo informações de funcionários, poderá usar os filtros por coluna do Lake Formation para impedir que funcionários que não trabalham no departamento de RH vejam informações de identificação pessoal (PII), como um número de previdência social. Para obter mais informações sobre os filtros de dados, consulte Filtragem de dados e segurança por célula no Lake Formation no Guia do desenvolvedor do AWS Lake Formation.

Você também pode usar etiquetas no Lake Formation para configurar permissões nos recursos do Lake Formation. Para obter mais informações, consulte Controle de acesso baseado em etiqueta do Lake Formation.

Atualmente, o Amazon Redshift é compatível com o compartilhamento de dados por meio do Lake Formation ao compartilhar dados dentro da mesma conta ou entre contas. O compartilhamento entre regiões ainda não é compatível.

Veja a seguir uma visão geral de alto nível sobre como usar o Lake Formation para controlar permissões de unidade de compartilhamento de dados.

1. No Amazon Redshift, o administrador do cluster produtor ou do grupo de trabalho cria uma unidade de compartilhamento de dados no cluster produtor ou no grupo de trabalho e concede o uso a uma conta do Lake Formation.
2. O administrador do cluster produtor ou do grupo de trabalho autoriza a conta do Lake Formation a ter acesso à unidade de compartilhamento de dados.
3. O administrador do Lake Formation descobre e registra as unidades de compartilhamento de dados. Ele também deve descobrir os ARNs do AWS Glue aos quais tem acesso e associar as unidades de compartilhamento de dados a um ARN do AWS Glue Data Catalog. Se você estiver usando a AWS CLI, poderá descobrir e aceitar unidades de compartilhamento de dados com as operações `describe-data-shares` e `associate-data-share-consumer` da CLI do Redshift. Para registrar uma unidade de compartilhamento de dados, use a operação `register-resource` da CLI do Lake Formation.
4. O administrador do Lake Formation cria um banco de dados federado no AWS Glue Data Catalog e configura as permissões do Lake Formation para controlar o acesso dos usuários aos objetos dentro da unidade de compartilhamento de dados. Para obter mais informações sobre bancos de dados federados no AWS Glue, consulte Gerenciar permissões para dados em uma unidade de compartilhamento de dados do Amazon Redshift.
5. Um administrador do Lake Formation descobre os bancos de dados do AWS Glue aos quais tem acesso e associa a unidade de compartilhamento de dados a um ARN do AWS Glue Data Catalog.
6. O administrador do Redshift descobre os ARNs de bancos de dados do AWS Glue aos quais tem acesso, cria um banco de dados externo no cluster consumidor do Amazon Redshift usando um ARN de banco de dados do AWS Glue e concede o uso a usuários do banco de dados autenticados com credenciais do IAM para começar a consultar o banco de dados do Amazon Redshift.
7. Os usuários do banco de dados podem usar as visualizações SVV_EXTERNAL_TABLES e SVV_EXTERNAL_COLUMNS para encontrar todas as tabelas ou colunas do banco de dados do AWS Glue às quais têm acesso, depois podem consultar as tabelas do banco de dados do AWS Glue.
8. Quando o administrador do cluster de produtor ou do grupo de trabalho decide não compartilhar mais os dados com o cluster de consumidor, o administrador do produtor pode revogar o uso, desautorizar ou excluir a unidade de compartilhamento de dados do Redshift. As permissões e os objetos associados no Lake Formation não são excluídos automaticamente.

Para obter mais informações sobre como compartilhar uma unidade de compartilhamento de dados com o AWS Lake Formation como um administrador do cluster produtor ou do grupo de trabalho, consulte Trabalhar com unidades de compartilhamento de dados gerenciadas pelo Lake Formation como produtor. Para consumir os dados compartilhados do cluster produtor ou do grupo de trabalho, consulte Trabalhar com unidades de compartilhamento de dados gerenciadas pelo Lake Formation como consumidor.

Valores de status das unidades de compartilhamento de dados no Amazon Redshift

Com o Amazon Redshift, é possível compartilhar dados em tempo real com segurança entre clusters do Amazon Redshift sem a necessidade de copiar ou transferir dados. Com as unidades de compartilhamento de dados do Amazon Redshift, é possível compartilhar resultados de consultas em tempo real, incluindo atualizações nos dados de origem, com qualquer cluster do Amazon Redshift na mesma conta ou em contas da AWS diferentes e em regiões da AWS. Este tópico descreve os possíveis status que as unidades de compartilhamento de dados podem ter no Amazon Redshift.

Com os datashares entre contas, existem diferentes status de datashares que exigem suas ações. Uma unidade de compartilhamento de dados pode ter o status ativo, ação necessária ou inativo.

O seguinte exemplo descreve cada status do datashare e sua ação necessária:

- Quando um administrador de produtor cria uma unidade de compartilhamento de dados, o status da unidade de compartilhamento de dados no cluster de produtor é Autorização pendente. O administrador do produtor pode autorizar os consumidores de dados a acessar a unidade de compartilhamento de dados. Não há nenhuma ação para o administrador do consumidor.
- Quando um administrador de produtor autoriza a unidade de compartilhamento de dados, o status da unidade de compartilhamento de dados se torna Autorizado no cluster de produtor. Não há nenhuma ação para o administrador do produtor. Quando há pelo menos uma associação com um consumidor de dados para o datashare, o status do datashare muda de Autorizado para Ativo.

  O status do datashare torna-se então Disponível (Ação necessária no console do Amazon Redshift) no cluster de consumidores. O administrador do consumidor pode associar a unidade de compartilhamento de dados aos consumidores de dados ou rejeitar a unidade de compartilhamento de dados. O administrador do consumidor também pode usar o comando `describeDatashareforConsumer` da AWS CLI para visualizar o status das unidades de compartilhamento de dados. Ou o administrador pode usar o comando da CLI

`describeDatashare` e fornecer o nome do recurso da Amazon (ARN) do datashare para visualizar o status do datashare.

- Quando o administrador do consumidor associa uma unidade de compartilhamento de dados a consumidores de dados, o status da unidade de compartilhamento de dados se torna Ativo no cluster de produtor. Quando há pelo menos uma associação com um consumidor de dados para o datashare, o status do datashare muda de Autorizado para Ativo. Nenhuma ação é necessária por parte do administrador do produtor.

  O status do datashare torna-se Ativo no cluster de consumidores. Nenhuma ação é necessária por parte do administrador do consumidor.
- Quando o administrador do consumidor remove uma associação de consumidor da unidade de compartilhamento de dados, o status da unidade de compartilhamento de dados se torna Ativo ou Autorizado. Se torna Ativo quando existe pelo menos uma associação para o datashare com outro consumidor de dados. Se torna Autorizado quando não há nenhuma associação de consumidores com o datashare no cluster de produtores. Não há nenhuma ação para o administrador do produtor.

  O status do datashare torna-se Ação necessária no cluster de consumidores se todas as associações forem removidas. O administrador do consumidor pode associar novamente uma unidade de compartilhamento de dados com consumidores de dados quando a unidade de compartilhamento de dados estiver disponível para os consumidores.
- Quando um administrador de cluster de consumidor recusa uma unidade de compartilhamento de dados, o status da unidade de compartilhamento de dados no cluster de produtor se torna Ação necessária e Recusado no cluster de consumidor. O administrador do produtor pode reautorizar a unidade de compartilhamento de dados. Não há nenhuma ação para o administrador do consumidor.
- Quando o administrador do produtor remove a autorização de uma unidade de compartilhamento de dados, o status da unidade de compartilhamento de dados se torna Ação necessária no cluster de produtor. O administrador do produtor pode optar por reautorizar a unidade de compartilhamento de dados, se necessário. Nenhuma ação é necessária por parte do administrador do consumidor.

Gerenciar o acesso a operações de API de compartilhamento de dados com políticas do IAM

Para controlar o acesso às operações de API de compartilhamento de dados, use políticas baseadas em ações do IAM. Para obter mais informações sobre como gerenciar políticas do IAM, consulte Gerenciar políticas do IAM no Manual do usuário do IAM.

Para obter informações sobre as permissões necessárias para usar as operações de API de compartilhamento de dados, consulte "Permissões necessárias para usar as operações de API de compartilhamento de dados" no Guia de gerenciamento de clusters do Amazon Redshift.

Para tornar o compartilhamento de dados entre contas mais seguro, você pode usar uma chave condicional `ConsumerIdentifier` para as operações de API `AuthorizeDataShare` e `DeauthorizeDataShare`. Com isso, é possível controlar explicitamente quais Contas da AWS poderão fazer chamadas para as duas operações de API.

Você pode negar a autorização ou desautorização do compartilhamento de dados para qualquer consumidor que não seja sua própria conta. Para isso, especifique o número da Conta da AWS na política do IAM.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Deny",
            "Action": [
                "redshift:AuthorizeDataShare",
                "redshift:DeauthorizeDataShare"
            ],
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "redshift:ConsumerIdentifier": "555555555555"
                }
            }
        }
    ]
```

```
}
```

Você pode permitir que um produtor com um DataShareArn **`testshare2`** compartilhe explicitamente com um consumidor que tem uma Conta da AWS de 111122223333 na política do IAM.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "redshift:AuthorizeDataShare",
                "redshift:DeauthorizeDataShare"
            ],
            "Resource": "arn:aws:redshift:us-
east-1:666666666666:datashare:af06285e-8a45-4ee9-b598-648c218c8ff1/testshare2",
            "Condition": {
                "StringEquals": {
                    "redshift:ConsumerIdentifier": "111122223333"
                }
            }
        }
    ]
}
```

Conectar-se a um banco de dados de consumidor do Amazon Redshift

Com uma conexão direta com um banco de dados de unidade de compartilhamento de dados, você pode se conectar diretamente a um banco de dados criado em uma unidade de compartilhamento de dados da mesma forma que você pode se conectar a qualquer outro tipo de banco de dados do Amazon Redshift. Por exemplo, você pode se conectar a um banco de dados criado em uma unidade de compartilhamento de dados usando drivers JDBC ou ODBC, o Editor de Consultas V2 do Amazon Redshift ou qualquer outra ferramenta que possa se conectar a um banco de dados

do Amazon Redshift. Para ter mais informações, consulte Conectar-se a um data warehouse do Amazon Redshift usando ferramentas de cliente SQL.

Acesso a dados compartilhados

Quando você se conecta a uma banco de dados criado em uma unidade de compartilhamento de dados, é possível consultar os objetos compartilhados usando a notação de duas partes (*schema_name*.*table_name*). Se a tabela puder ser encontrada no caminho de pesquisa do banco de dados do consumidor, você também poderá usar a notação de uma parte (*table_name*).

Se você quiser realizar consultas entre bancos de dados, é possível usar a notação de três partes (*consumer_database_name*.*schema_name*.*table_name*). Essas consultas podem fazer referência a objetos compartilhados de outro banco de dados de consumidor no cluster ou a objetos locais de bancos de dados locais. Elas também podem fazer referência a bancos de dados locais e dados compartilhados de outros clusters dentro da mesma consulta.

> **Note**
>
> Um banco de dados criado em uma unidade de compartilhamento de dados não tem um catálogo local. Portanto, todas as consultas que acessam tabelas de catálogos locais, como `pg_class`, enviam de volta um resultado vazio.

Acessar metadados referentes a objetos compartilhados

Para ajudar os administradores de cluster a descobrirem objetos compartilhados no banco de dados do consumidor, o Amazon Redshift fornece um conjunto de visualizações de metadados e comandos SHOW que listam os metadados referentes a esses objetos. Quando você se conecta a um banco de dados de consumidor, essas visualizações de metadados e comandos não permitem a descoberta de metadados entre bancos de dados. Eles enviam de volta apenas metadados referentes a objetos compartilhados na unidade de compartilhamento de dados associada ao banco de dados conectado.

Use SHOW SCHEMAS para visualizar uma lista dos esquemas compartilhados na unidade de compartilhamento de dados associada ao banco de dados conectado. Para obter mais informações, consulte SHOW SCHEMAS.

Use SHOW TABLES para visualizar uma lista das tabelas em um esquema compartilhado na unidade de compartilhamento de dados associada ao banco de dados conectado. Para obter mais informações, consulte SHOW TABLES.

Use SHOW COLUMNS para visualizar uma lista das coluna em uma tabela compartilhada na unidade de compartilhamento de dados associada ao banco de dados conectado. Para obter mais informações, consulte SHOW COLUMNS.

Use SVV_ALL_SCHEMAS para visualizar uma lista dos esquemas compartilhados na unidade de compartilhamento de dados associada ao banco de dados conectado. Para obter mais informações, consulte SVV_ALL_SCHEMAS.

Use SVV_ALL_TABLES para visualizar uma lista das tabelas compartilhadas na unidade de compartilhamento de dados associada ao banco de dados conectado. Para obter mais informações, consulte SVV_ALL_TABLES.

Use SVV_ALL_COLUMNS para visualizar uma lista das colunas compartilhadas na unidade de compartilhamento de dados associada ao banco de dados conectado. Para obter mais informações, consulte SVV_ALL_COLUMNS.

Integração do compartilhamento de dados do Amazon Redshift com ferramentas de business intelligence

Para integrar o compartilhamento de dados com ferramentas de business intelligence (BI), recomendamos que você use os drivers JDBC ou ODBC do Amazon Redshift. Os drivers JDBC e ODBC do Amazon Redshift aceitam a operação de API `GetCatalogs` nos drivers. Essa operação envia de volta uma lista de todos os bancos de dados, incluindo aqueles criados em unidades de compartilhamento de dados.

Os drivers também aceitam operações subsequentes, como `GetSchemas` e GetTables, que enviam de volta dados de todos os bancos de dados que `GetCatalogs` envia de volta. Eles as aceitam mesmo quando você não especifica explicitamente o catálogo na chamada. Para ter mais informações sobre os drivers JDBC e ODBC, consulte Configurar conexões no Guia de gerenciamento do Amazon Redshift.

O Editor de Consultas V2 do Amazon Redshift inclui bancos de dados de consumidor em sua interface de comutação de conexão. Entretanto, a maioria das ferramentas exclui esses bancos de dados e inclui apenas bancos de dados de cluster locais como bancos de dados conectáveis.

> **Note**
>
> Um novo banco de dados de sistema, chamado `sys:internal`, foi adicionado para manutenção interna. Algumas ferramentas incluem esse banco de dados de sistema como

um banco de dados conectável. No entanto, não é possível estabelecer conexão com o banco de dados e executar consultas nos respectivos objetos.

Monitoramento e auditoria do compartilhamento de dados no Amazon Redshift

Com o Amazon Redshift, é possível monitorar e auditar as atividades de compartilhamento de dados para manter a conformidade e a segurança.

Ao auditar o compartilhamento de dados, os produtores podem acompanhar a evolução do datashare. Por exemplo, a auditoria ajuda a rastrear quando unidades de compartilhamento de dados são criadas, objetos são adicionados ou removidos e permissões são concedidas ou revogadas em clusters do Amazon Redshift, contas da AWS ou regiões da AWS.

Além da auditoria, produtores e consumidores rastreiam o uso de unidades de compartilhamento de dados em várias granularidades, como conta, cluster e objeto. Para obter mais informações sobre rastreamento de uso e auditoria de visualizações, consulte SVL_DATASHARE_CHANGE_LOG e SVL_DATASHARE_USAGE_PRODUCER.

É possível monitorar as unidades de compartilhamento de dados consultando as visualizações do sistema.

1. O administrador do produtor que deseja compartilhar dados cria uma unidade de compartilhamento de dados do Amazon Redshift. Em seguida, ele adiciona os objetos de banco de dados necessários. Estes poderão ser esquemas, tabelas e visualizações para a unidade de compartilhamento de dados, e especificam uma lista de consumidores com os quais os objetos deverão ser compartilhados.

   Use as seguintes visualizações do sistema para ver visualizações consolidadas e rastrear alterações e uso de unidades de compartilhamento de dados em clusters de produtores e/ou consumidores:

   - SYS_DATASHARE_CHANGE_LOG
   - SYS_DATASHARE_USAGE_CONSUMER
   - SYS_DATASHARE_USAGE_PRODUCER

   Use as seguintes exibições do sistema para ver objetos do datashare e informações do consumidor de dados para datashares de saída:

- SVV_DATASHARES
- SVV_DATASHARE_CONSUMERS
- SVV_DATASHARE_OBJECTS

2. Os administradores do consumidor analisam as unidades de compartilhamento de dados para as quais têm permissão de uso e revisam o conteúdo de cada uma visualizando as unidades de compartilhamento de dados de entrada que usam SVV_DATASHARES.

   Para consumir dados compartilhados, cada administrador de consumidor cria um banco de dados do Amazon Redshift com base na unidade de compartilhamento de dados. O administrador acaba atribuindo permissões a funções e usuários indicados no cluster consumidor. Os usuários e as funções podem listar os objetos compartilhados como parte das consultas de metadados padrão mostrando as exibições do sistema de metadados a seguir e começar a consultar imediatamente os dados.

   - SVV_REDSHIFT_COLUMNS
   - SVV_REDSHIFT_DATABASES
   - SVV_REDSHIFT_FUNCTIONS
   - SVV_REDSHIFT_SCHEMAS
   - SVV_REDSHIFT_TABLES

   Para exibir objetos de esquemas locais e compartilhados do Amazon Redshift e esquemas externos, use as visualizações do sistema de metadados a seguir para consultá-los.

   - SVV_ALL_COLUMNS
   - SVV_ALL_SCHEMAS
   - SVV_ALL_TABLES

Quando você se conecta a um banco de dados de consumidor, a descoberta entre bancos de dados é desabilitada. As visualizações do sistema de metadados enviam de volta apenas metadados referentes a objetos compartilhados na unidade de compartilhamento de dados associada ao banco de dados conectado.

Integrar o compartilhamento de dados do Amazon Redshift com o AWS CloudTrail

O compartilhamento de dados é integrado ao AWS CloudTrail. O CloudTrail é um serviço que fornece um registro das ações realizadas por um usuário, função ou um produto da AWS no Amazon

Redshift. O CloudTrail captura as chamadas de API para compartilhar dados como eventos. As chamadas capturadas incluem chamadas do console do AWS CloudTrail e chamadas de código para as operações de compartilhamento de dados. Para obter mais informações sobre a integração do Amazon Redshift com o AWS CloudTrail, consulte “Logging with CloudTrail” (Registro em log com o CloudTrail).

Para obter mais informações sobre o CloudTrail, consulte “Como o CloudTrail funciona”.

Dados semiestruturados no Amazon Redshift

Usando suporte a dados semiestruturados no Amazon Redshift, você pode ingerir e armazenar dados semiestruturados em seus data warehouses do Amazon Redshift. Usando o tipo de dados SUPER e a linguagem PartiQL, o Amazon Redshift expande a capacidade de data warehouse para integrar com fontes de dados SQL e NoSQL. Dessa forma, o Amazon Redshift permite análises eficientes em dados armazenados relacionais e semiestruturados, como JSON. Para ter mais informações sobre a integração do Amazon Redshift com PartiQL, consulte PartiQL: uma linguagem de consultas compatível com SQL para o Amazon Redshift.

O Amazon Redshift oferece duas formas de suporte a dados semiestruturados: o tipo de dados SUPER e o Amazon Redshift Spectrum.

Você pode consultar dados semiestruturados ingerindo-os no Amazon Redshift e armazenando-os no tipo de dados SUPER ou usar o Amazon Redshift Spectrum para consultar os dados armazenados no Amazon S3.

Práticas recomendadas para trabalhar com dados semiestruturados

Considere as seguintes práticas recomendadas ao trabalhar com dados semiestruturados.

- Recomendamos que você defina a opção de configuração `enable_case_sensitive_super_attribute` como verdadeira ao trabalhar com dados SUPER. Para obter mais informações, consulte enable_case_sensitive_super_attribute.
- Use o comando COPY para carregar dados de buckets do Amazon S3 nas colunas SUPER do Amazon Redshift.
- Use a digitação dinâmica PartiQL e semântica lax para executar consultas ad hoc em valores de dados SUPER sem a necessidade de impor um esquema antes da consulta. Para obter informações sobre digitação dinâmica, consulte Digitação dinâmica. Para obter mais informações sobre semântica lax, consulte Semântica lax.
- Destrua dados sem esquemas e semiestruturados em visualizações materializadas usando o PartiQL se você planeja consultar os dados com frequência. Quando você executa análises nos dados destruídos, a organização colunar das exibições materializadas do Amazon Redshift oferece melhor performance. Além disso, usuários e ferramentas de business intelligence (BI) que

exigem um esquema convencional para dados ingeridos podem usar visualizações (materializadas ou virtuais) como a apresentação convencional do esquema dos dados.

Depois que suas visões materializadas do PartiQL extraírem os dados encontrados no JSON ou SUPER em visões materializadas colunares convencionais, você pode consultar as visões materializadas. Para obter informações sobre como consultar visões materializadas, consulte Visões materializadas no Amazon Redshift. Para obter mais informações sobre como o tipo de dados SUPER funciona com visões materializadas, consulte Tipo de dados SUPER com visões materializadas.

Conceitos para uso do tipo de dados SUPER

A seguir, você pode encontrar alguns conceitos de tipo de dados SUPER do Amazon Redshift.

Entenda qual é o tipo de dados SUPER no Amazon Redshift — OSUPER é um tipo de dados do Amazon Redshift que permite o armazenamento de arrays sem esquema e estruturas que contêm escalares do Amazon Redshift e possivelmente arrays e estruturas aninhadas. O tipo de dados SUPER pode armazenar nativamente diferentes formatos de dados semiestruturados, como JSON ou dados provenientes de fontes orientadas a documentos. Você pode adicionar uma nova coluna SUPER para armazenar dados semiestruturados e escrever consultas que acessam a coluna SUPER, juntamente com as colunas escalares usuais. Para obter mais informações sobre tipos de dados SUPER, consulte Tipo SUPER.

Ingerir JSON sem esquema em SUPER — Com o tipo de dados SUPER semiestruturado flexível, o Amazon Redshift pode receber e ingerir JSON sem esquema em colunas com um tipo de dados SUPER. Por exemplo, você pode ingerir o valor JSON `[10.5, “first”]` em uma coluna de tipo de dados SUPER usando o comando COPY. A coluna teria um valor SUPER de `[10.5, ‘first’]`. Você também pode ingerir JSON usando Função JSON_PARSE. COPY e `json_parse` ingerem JSON usando semântica de análise estrita por padrão. Você também pode construir valores SUPER, incluindo arrays e estruturas, usando os próprios dados do banco de dados.

A coluna SUPER não requer modificações de esquema ao ingerir as estruturas flexíveis de JSON sem esquema. Por exemplo, ao analisar um fluxo de cliques, você inicialmente armazena na coluna SUPER estruturas “clique” com atributos “IP” e “time”. Você pode adicionar um atributo “id do cliente” sem alterar seu esquema para ingerir tais alterações.

O formato nativo usado para o tipo de dados SUPER é um formato binário que requer menos espaço do que o valor JSON em sua forma textual. Isso permite uma ingestão mais rápida e processamento de tempo de execução de valores SUPER na consulta.

Consultar dados SUPER com PartiQL: PartiQL é uma extensão compatível com versões anteriores do SQL-92 que muitos serviços da AWS atualmente usam. Com o uso do PartiQL, construções SQL familiares combinam perfeitamente o acesso aos dados SQL clássicos e tabulares e aos dados semiestruturados do SUPER. Você pode executar a navegação de objetos e arrays e desaninhar arrays. O PartiQL estende a linguagem SQL padrão para expressar declarativamente e processar dados aninhados e multivalorizados.

PartiQL é uma extensão do SQL onde os dados aninhados e sem esquema de colunas SUPER são cidadãos de primeira classe. O PartiQL não requer que todas as expressões de consulta sejam verificadas durante o tempo de compilação da consulta. Essa abordagem permite que expressões de consulta que contêm o tipo de dados SUPER sejam digitadas dinamicamente durante a execução da consulta quando os tipos reais de dados dentro das colunas SUPER são acessados. Além disso, o PartiQL opera em um modo lax onde as inconsistências de tipo não causam falhas, mas retornam null. A combinação de processamento de consultas sem esquema e lax torna o PartiQL ideal para aplicativos de extração, carregamento, transformação (ELT) em que sua consulta SQL avalia os dados JSON que são ingeridos nas colunas SUPER.

Para obter mais informações sobre PartiQL para Amazon Redshift, consulte PartiQL: uma linguagem de consultas compatível com SQL para o Amazon Redshift. Para obter informações sobre digitação dinâmica, consulte Digitação dinâmica. Para obter mais informações sobre o processamento de consulta lax, consulte Semântica lax.

Integrar ao Redshift Spectrum — O Amazon Redshift oferece suporte a vários aspectos do PartiQL ao executar consultas do Redshift Spectrum em JSON, Parquet e outros formatos que têm dados aninhados. O Redshift Spectrum suporta apenas dados aninhados que possuem esquemas. Por exemplo, com o Redshift Spectrum, você pode declarar que seus dados JSON têm o atributo `nested_schemaful_example` no esquema `ARRAY<STRUCT<a:INTEGER, b:DECIMAL(5,2)>>`. O esquema deste atributo determina que os dados sempre contém uma matriz, que contém uma estrutura com inteiro a e decimal b. Se os dados forem alterados para incluir mais atributos, o tipo também será alterado. Em contraste, o tipo de dados SUPER não requer esquema. Você pode armazenar arrays com elementos de estrutura que têm diferentes atributos ou tipos. Além disso, os valores podem ser armazenados fora de arrays.

Considerações sobre o uso de dados do tipo SUPER

Ao trabalhar com dados SUPER, considere o seguinte:

- Use o driver JDBC versão 2.x, o driver ODBC versão 2.x ou o driver do Amazon Redshift Python versão 2.0.872 ou posterior. A versão 1.x do driver ODBC não tem suporte.

  Para obter mais informações sobre drivers JDBC, consulte "Configurar uma conexão para a versão 2.x do driver JDBC para o Amazon Redshift no Guia de gerenciamento do Amazon Redshift.

  Para obter mais informações sobre drivers ODBC, consulte "Configurar uma conexão para a versão 2.x do driver ODBC para o Amazon Redshift no Guia de gerenciamento do Amazon Redshift.

  Para obter mais informações sobre drivers Python, consulte Conector do Amazon Redshift Python no Guia de gerenciamento do Amazon Redshift.

Para obter mais informações sobre as configurações SUPER, consulte Configurações SUPER.

PartiQL: uma linguagem de consultas compatível com SQL para o Amazon Redshift

O Amazon Redshift oferece suporte a PartiQL, uma linguagem de consultas compatível com SQL, para selecionar, inserir, atualizar e excluir dados no Amazon Redshift. Usando PartiQL, você pode interagir facilmente com tabelas do Amazon Redshift e executar consultas ad hoc usando o Console de gerenciamento da AWS, o SQL Workbench/J, a AWS Command Line Interface e as APIs de dados do Amazon Redshift para PartiQL.

As operações PartiQL fornecem as mesmas disponibilidade, latência e performance que as outras operações de plano de dados do Redshift.

As seções a seguir descrevem a implementação de PartiQL do Amazon Redshift.

O que é PartiQL?

A linguagem PartiQL garante acesso de consultas compatíveis com SQL em vários armazenamentos de dados que contêm dados estruturados, dados semiestruturados e dados aninhados. Ela é amplamente utilizada na Amazon e agora está disponível como parte de muitos serviços da AWS, incluindo o Amazon Redshift.

Para obter a especificação da PartiQL e um tutorial sobre a linguagem de consulta principal, consulte a Visão geral da PartiQL.

Para obter exemplos de uso de consultas PartiQL no Amazon Redshift com dados semiestruturados, consulte Exemplos de uso de dados semiestruturados no Amazon Redshift.

> **Note**
>
> - O Amazon Redshift oferece suporte a um subconjunto da linguagem de consultas PartiQL.
> - O Amazon Redshift não é compatível com o formato de dados Amazon Ion nem com liderais do Amazon Ion.

PartiQL no Amazon Redshift

Para executar consultas PartiQL no Amazon Redshift, você pode usar os seguintes métodos:

- O Console de gerenciamento da AWS
- SQL Workbench/J
- O AWS CLI
- APIs de dados do Amazon Redshift para PartiQL

Carregamento de dados semiestruturados no Amazon Redshift

Use o tipo de dados SUPER para analisar e consultar dados hierárquicos e genéricos no Amazon Redshift. O Amazon Redshift apresenta Função JSON_PARSE para analisar dados no formato JSON e convertê-los na representação SUPER. O Amazon Redshift também oferece suporte ao carregamento de colunas SUPER no comando COPY Os formatos de arquivo compatíveis são JSON, Avro, texto, formato CSV (valores separados por vírgula), Parquet e ORC.

Você pode usar JSON_PARSE para inserir e atualizar dados JSON em colunas SUPER, ou usar COPY para carregar dados JSON no Amazon Redshift de fontes externas, como buckets do Amazon S3.

Tópicos

- Usar JSON_PARSE para inserir dados em colunas SUPER

- Usar COPY para carregar dados em colunas SUPER

Usar JSON_PARSE para inserir dados em colunas SUPER

Você pode inserir ou atualizar dados JSON em uma coluna SUPER usando Função JSON_PARSE. A função analisa dados no formato JSON e converte no tipo de dados SUPER, que você pode usar em instruções INSERT ou UPDATE.

O exemplo a seguir insere dados JSON em uma coluna SUPER. Se a função JSON_PARSE estiver ausente na consulta, o Amazon Redshift trata o valor como uma única string em vez de uma string formatada em JSON que deve ser analisada.

```
--Drop the table if it exists.
DROP TABLE IF EXISTS test_json;

--Create the table.
CREATE TABLE test_json (all_data SUPER);

--Populate the table.
INSERT INTO test_json VALUES (JSON_PARSE('
{
    "name": {
        "first_name": "Jake",
        "last_name": "Smith"
    },
    "age": 30,
    "hobby": "Biking"
}'
) );

SELECT * FROM test_json;

 all_data
---------
{"name":{"first_name":"Jake","last_name":"Smith"},"age":30,"hobby":"Biking"}
```

Usar COPY para carregar dados em colunas SUPER

Nas seções a seguir, você pode aprender sobre diferentes maneiras de usar o comando COPY para carregar dados JSON no Amazon Redshift. Para obter informações sobre os parâmetros de formato

de dados que o Amazon Redshift usa para analisar JSON em comandos COPY, leia a descrição do parâmetro JSON format for COPY em Parâmetros de formato de dados.

Tópicos

- Copiar dados de JSON e Avro
- Copiar dados de texto e CSV
- Copiar dados de Parquet e ORC em formato de coluna

Copiar dados de JSON e Avro

O Amazon Redshift fornece os seguintes métodos para ingerir um documento JSON usando COPY, mesmo com uma estrutura JSON totalmente ou parcialmente desconhecida:

- Armazene os dados derivados de um documento JSON em uma única coluna de dados SUPER usando a opção `noshred`. Este método é útil quando o esquema não é conhecido ou é esperado para mudar. Assim, este método torna mais fácil armazenar toda a tupla em uma única coluna SUPER.
- Destruir o documento JSON em várias colunas do Amazon Redshift usando o método `auto` ou a opção `jsonpaths`. Os atributos podem ser escalares do Amazon Redshift ou valores SUPER.

Você pode usar essas opções com os formatos JSON ou Avro. Para obter mais informações sobre opções JSON como `noshred`, `auto`, e `jsonpaths`, consulte JSON format for COPY.

O tamanho máximo de um objeto JSON no Amazon Redshift é 4 MB, o que se aplica antes de qualquer fragmentação ou análise.

Método 1: copiar um documento JSON em uma única coluna de dados SUPER usando **`noshred`**

Você pode copiar documentos JSON inteiros em colunas de dados SUPER únicas usando a opção `noshred` no comando COPY. Considere o seguinte exemplo:

1. Crie uma tabela com uma única coluna de dados SUPER.

   ```
   CREATE TABLE region_nations_noshred (rdata SUPER);
   ```

2. Copie os dados do Amazon S3 na única coluna de dados SUPER. Para ingerir os dados de origem JSON em uma única coluna de dados SUPER, especifique a propriedade `noshred` na cláusula FORMAT JSON.

```
COPY region_nations_noshred FROM 's3://redshift-downloads/semistructured/tpch-
nested/data/json/region_nation'
REGION 'us-east-1' IAM_ROLE 'arn:aws:iam::xxxxxxxxxxxx:role/Redshift-S3'
FORMAT JSON 'noshred';
```

Depois que COPY ingeriu com êxito o JSON, sua tabela tem uma coluna `rdata` de dados SUPER que contém os dados de todo o objeto JSON. Os dados ingeridos mantêm todas as propriedades da hierarquia JSON. No entanto, as folhas são convertidas para tipos escalares do Amazon Redshift para processamento eficiente de consultas.

3. Use a consulta a seguir para recuperar a string JSON original.

```
SELECT rdata FROM region_nations_noshred;
```

Quando o Amazon Redshift gera uma coluna de dados SUPER, ela se torna acessível usando JDBC como uma string por meio da serialização JSON. Para obter mais informações, consulte Serializar JSON aninhado complexo.

Método 2: copiar um documento JSON em múltiplas colunas de dados SUPER

Você pode fragmentar um documento JSON em várias colunas que podem ser colunas de dados SUPER ou tipos escalares do Amazon Redshift. O Amazon Redshift espalha diferentes partes do objeto JSON em colunas diferentes. Considere o seguinte exemplo:

1. Crie uma tabela com várias colunas para armazenar o JSON fragmentado.

```
CREATE TABLE region_nations
(
 r_regionkey smallint
 ,r_name varchar
 ,r_comment varchar
 ,r_nations super
 );
```

2. Para copiar JSON para a tabela `region_nations`, especifique a opção AUTO na cláusula FORMAT JSON para dividir o valor JSON em várias colunas. COPY corresponde aos atributos JSON de nível superior com nomes de coluna e permite que valores aninhados sejam ingeridos como valores SUPER, como matrizes e objetos JSON.

```
COPY region_nations FROM 's3://redshift-downloads/semistructured/tpch-nested/data/
json/region_nation'
REGION 'us-east-1' IAM_ROLE 'arn:aws:iam::xxxxxxxxxxx:role/Redshift-S3'
FORMAT JSON 'auto';
```

Quando os nomes de atributo JSON estiverem em maiúsculas e minúsculas mistas, especifique a propriedade `auto ignorecase` na cláusula FORMAT JSON. Para obter mais informações sobre o uso do comando COPY, consulte Carregar de dados JSON usando a opção "auto ignorecase".

Em alguns casos, há uma incompatibilidade entre nomes de coluna e atributos JSON ou o atributo a ser carregado é aninhado mais do que um nível profundo. Em caso afirmativo, use um arquivo `jsonpaths` para mapear manualmente atributos JSON para colunas do Amazon Redshift. Considere o seguinte exemplo:

1. Crie uma tabela com várias colunas para armazenar o JSON fragmentado.

```
CREATE TABLE region_nations
(
 r_regionkey smallint
 ,r_name varchar
 ,r_comment varchar
 ,r_nations super
 );
```

2. Nesse caso, os nomes das colunas não correspondem aos atributos JSON. Você pode criar um arquivo `jsonpaths` que mapeia os caminhos dos atributos para as colunas da tabela por sua posição no array `jsonpaths` como o seguinte:

```
{"jsonpaths": [
       "$.r_regionkey",
       "$.r_name",
       "$.r_comment",
       "$.r_nations
    ]
}
```

3. Use o local do arquivo `jsonpaths` como argumento para a opção FORMAT JSON em COPY.

```
COPY nations FROM 's3://redshift-downloads/semistructured/tpch-nested/data/json/
region_nation'
REGION 'us-east-1' IAM_ROLE 'arn:aws:iam::xxxxxxxxxxx:role/Redshift-S3'
FORMAT JSON 's3://redshift-downloads/semistructured/tpch-nested/data/jsonpaths/
nations_jsonpaths.json';
```

4. Use a consulta a seguir para acessar a tabela que mostra a dispersão de dados para várias colunas. As colunas de dados SUPER são impressas usando o formato JSON.

```
SELECT r_regionkey,r_name,r_comment,r_nations[0].n_nationkey FROM region_nations
 ORDER BY 1,2,3 LIMIT 1;
```

Arquivos `jsonpaths` mapeiam campos no documento JSON para colunas de tabela. É possível extrair outras colunas, como chaves de distribuição e classificação, enquanto ainda carrega o documento completo como uma coluna SUPER. A consulta a seguir carrega o documento completo na coluna `nations`. A coluna `name` é a chave de classificação, e a coluna `regionkey` é a chave de distribuição. Considere o seguinte exemplo:

1. Crie uma tabela com várias colunas para armazenar o JSON fragmentado.

```
CREATE TABLE nations_sorted (
    regionkey smallint,
    name varchar,
    nations super
) DISTKEY(regionkey) SORTKEY(name);
```

2. Mapeie o jsonpath raiz “$” para a raiz do documento da seguinte maneira:

```
{"jsonpaths": [
       "$.r_regionkey",
       "$.r_name",
       "$"
    ]
}
```

3. Use o local do arquivo `jsonpaths` como argumento para a opção FORMAT JSON em COPY.

```
COPY nations_sorted FROM 's3://redshift-downloads/semistructured/tpch-nested/data/
json/region_nation'
```

```
REGION 'us-east-1' IAM_ROLE 'arn:aws:iam::xxxxxxxxxxx:role/Redshift-S3'
FORMAT JSON 's3://redshift-downloads/semistructured/tpch-nested/data/jsonpaths/
nations_sorted_jsonpaths.json';
```

Para obter mais informações sobre como usar o `jsonpaths`, consulte COPY no formato JSON.

Copiar dados de texto e CSV

O Amazon Redshift representa colunas SUPER em formatos de texto e CSV como JSON serializado. A formatação JSON válida é necessária para que as colunas SUPER sejam carregadas com as informações do tipo correto. Tire aspas de que objetos, matrizes, números, boolianos e valores nulos. Quebre valores de strings entre aspas duplas. As colunas SUPER usam regras de escape padrão para formatos de texto e CSV.

Ao copiar de CSV, os delimitadores usam caracteres de escape de acordo com o padrão CSV. Considere o seguinte exemplo:

```
CREATE TABLE region_nations
(
 r_regionkey smallint
 ,r_name varchar
 ,r_comment varchar
 ,r_nations super
 );

 COPY region_nations FROM 's3://redshift-downloads/semistructured/tpch-nested/data/csv/
region_nation'
REGION 'us-east-1' IAM_ROLE 'arn:aws:iam::xxxxxxxxxxx:role/Redshift-S3'
FORMAT CSV;
```

Ao copiar de texto, quando o delimitador escolhido também pode aparecer em um campo SUPER, use a opção ESCAPE durante COPY e UNLOAD. Considere o seguinte exemplo:

```
CREATE TABLE region_nations
(
 r_regionkey smallint
 ,r_name varchar
 ,r_comment varchar
 ,r_nations super
 );
```

```
COPY region_nations FROM 's3://redshift-downloads/semistructured/tpch-nested/data/text/
region_nation'
REGION 'us-east-1' IAM_ROLE 'arn:aws:iam::xxxxxxxxxxxx:role/Redshift-S3'
DELIMITER ','
ESCAPE;
```

Copiar dados de Parquet e ORC em formato de coluna

Se seus dados semiestruturados ou aninhados já estiverem disponíveis no formato Apache Parquet ou Apache ORC, você pode usar o comando COPY para ingerir dados no Amazon Redshift.

A estrutura da tabela do Amazon Redshift deve corresponder ao número de colunas e aos tipos de dados da coluna dos arquivos Parquet ou ORC. Especificando SERIALIZETOJSON no comando COPY, você pode carregar qualquer tipo de coluna no arquivo que se alinha com uma coluna SUPER na tabela como SUPER. Isso inclui tipos de estrutura e array.

O exemplo a seguir usa um formato Parquet.

```
COPY region_nations FROM 's3://redshift-downloads/semistructured/tpch-nested/data/
parquet/region_nation'
REGION 'us-east-1' IAM_ROLE 'arn:aws:iam::xxxxxxxxxxxx:role/Redshift-S3'
FORMAT PARQUET SERIALIZETOJSON;
```

O exemplo a seguir usa um formato ORC.

```
COPY region_nations FROM 's3://redshift-downloads/semistructured/tpch-nested/data/orc/
region_nation'
IAM_ROLE 'arn:aws:iam::xxxxxxxxxxxx:role/Redshift-S3'
FORMAT ORC SERIALIZETOJSON;
```

Quando os atributos dos tipos de dados de data ou hora estão no ORC, o Amazon Redshift os converte em varchar ao codificá-los em SUPER.

Descarregamento de dados semiestruturados

Com o Amazon Redshift, você pode exportar dados semiestruturados do cluster do Amazon Redshift para o Amazon S3 em vários formatos, incluindo texto, Apache Parquet, Apache ORC e Avro. As seções a seguir orientam você no processo de configuração e execução de operações de descarregamento de dados semiestruturados no Amazon Redshift.

CSV or text formats

É possível descarregar tabelas com colunas de dados SUPER no Amazon S3 em um formato CSV (valores separados por vírgula) ou em texto. Usando uma combinação de cláusulas de navegação e unnest, o Amazon Redshift descarrega dados hierárquicos no formato de dados SUPER para o Amazon S3 em formatos CSV ou de texto. Posteriormente, você pode criar tabelas externas contra dados descarregados e consultá-los usando o Redshift Spectrum. Para obter informações sobre como usar UNLOAD e as permissões necessárias do IAM, consulte UNLOAD.

O exemplo a seguir descarrega todos os dados de uma tabela do Amazon Redshift em um bucket do Amazon S3.

```
UNLOAD ('SELECT * FROM <redshift_table>')
TO '<S3_bucket>'
IAM_ROLE '<iam_role>'
DELIMITER AS '|'
GZIP
ALLOWOVERWRITE;
```

Ao contrário de outros tipos de dados em que uma string definida pelo usuário representa um valor nulo, o Amazon Redshift exporta as colunas de dados SUPER usando o formato JSON e a representa como nulo conforme determinado pelo formato JSON. Como resultado, as colunas de dados SUPER ignoram a opção NULL [AS] usada nos comandos UNLOAD.

Parquet format

É possível descarregar tabelas com colunas de dados SUPER para o Amazon S3 no formato Parquet. O Amazon Redshift representa as colunas SUPER em Parquet como o tipo de dados JSON. Isso permite que dados semiestruturados sejam representados em Parquet. É possível consultar essas colunas usando o Redshift Spectrum ou ingeri-las de volta ao Amazon Redshift usando o comando COPY. Para obter informações sobre como usar UNLOAD e as permissões necessárias do IAM, consulte UNLOAD.

O exemplo a seguir descarrega todos os dados de uma tabela do Amazon Redshift em um bucket do Amazon S3 no formato Parquet.

```
UNLOAD ('SELECT * FROM <Amazon Redshift_table>')
TO '<S3_bucket>'
IAM_ROLE '<iam_role>'
```

```
FORMAT PARQUET;
```

Consultar dados semiestruturados

Com o Amazon Redshift, é possível consultar e analisar dados semiestruturados, como JSON, Avro ou Ion, bem como dados estruturados. Dados semiestruturados referem-se a dados que têm um esquema flexível que permite estruturas hierárquicas ou agrupadas. As seções a seguir demonstram como consultar dados semiestruturados usando o suporte do Amazon Redshift a formatos de dados abertos, para que seja possível obter informações valiosas de estruturas de dados complexas.

O Amazon Redshift usa a linguagem PartiQL para oferecer acesso compatível com SQL a dados relacionais, semiestruturados e aninhados.

O PartiQL opera com tipos dinâmicos. Esse método permite filtragem intuitiva, junção e agregação na combinação de conjuntos de dados estruturados, semiestruturados e aninhados. A sintaxe do PartiQL usa notação pontilhada e array subscript para navegação de caminho ao acessar dados aninhados. Ele também permite que os itens da cláusula FROM para iterar sobre arrays e usar para operações unnest. As seções a seguir descrevem os diferentes padrões de consulta que combinam o uso do tipo de dados SUPER com a navegação de caminho e matriz, desfazer aninhamento, transformar colunas em linhas ou junções. Para obter mais informações sobre a PartiQL, consulte PartiQL: uma linguagem de consultas compatível com SQL para o Amazon Redshift.

Navegação

O Amazon Redshift usa o PartiQL para habilitar a navegação em arrays e estruturas usando o colchete [...] e a notação de pontos, respectivamente. Além disso, você pode misturar navegação em estruturas usando a notação de pontos e arrays usando a notação de colchetes. Por exemplo, a instrução a seguir seleciona somente o terceiro elemento em uma matriz aninhada a um nível de profundidade em um objeto SUPER:

```
SELECT super_object.array[2];

 array
---------------
 third_element
```

Você pode usar a notação de ponto e colchetes ao realizar operações de dados, como filtrar, unir e agregar. Você pode usar essas notações em qualquer lugar em uma consulta na qual normalmente

há referências de coluna. Por exemplo, a instrução a seguir seleciona o número de eventos com o tipo UPDATED.

```
SELECT COUNT(*)
FROM test_json
WHERE all_data.data.pnr.events[0]."eventType" = 'UPDATED';

 eventType | count
-----------+-------
 "UPDATED" | 1
```

Para obter exemplos mais detalhados do uso da navegação PartiQL, consulte Exemplos de uso de dados semiestruturados no Amazon Redshift.

Desaninhar consultas

Para consultas desaninhadas, o Amazon Redshift oferece duas maneiras de iterar em matrizes SUPER: a sintaxe PartiQL e a operação UNNEST na cláusula FROM. Ambos os métodos de desagrupamento geram a mesma saída. Para ter informações sobre a operação UNNEST, consulte Cláusula FROM. Para obter exemplos de uso da operação UNNEST, consulte Exemplos de UNNEST.

O Amazon Redshift pode percorrer matrizes SUPER usando a sintaxe PartiQL na cláusula FROM de uma consulta. Com o exemplo anterior, o exemplo a seguir itera sobre os valores do atributo para `c_orders`.

```
SELECT orders.*, o FROM customer_orders orders, orders.c_orders o;
```

A sintaxe PartiQL de desaninhamento usando o item da cláusula FROM `x (AS) y` significa que y itera sobre cada valor (SUPER) na expressão de matriz x. Nesse caso, x é uma expressão SUPER, e y é um alias para x.

O operando esquerdo também pode usar a notação de pontos e colchetes para navegação regular. No exemplo a seguir, `customer_orders_lineitem c` é a iteração sobre a tabela `customer_order_lineitem` de base, e `c.c_orders o` é a iteração sobre a matriz `c.c_orders`. Para iterar sobre o atributo `o_lineitems`, que é uma matriz dentro de uma matriz, você pode adicionar várias cláusulas, como:

```
SELECT c.*, o, l FROM customer_orders_lineitem c, c.c_orders o, o.o_lineitems l;
```

O Amazon Redshift também oferece suporte ao índice de matriz ao iterar sobre a matriz usando a palavra-chave AT. A cláusula `x AS y AT z` itera sobre a matriz `x` e gera o campo `z,`, que é o índice da matriz. O exemplo a seguir mostra como o índice da matriz funciona.

```
SELECT c_name,
       orders.o_orderkey AS orderkey,
       index AS orderkey_index
FROM customer_orders_lineitem c, c.c_orders AS orders AT index
ORDER BY orderkey_index;

c_name             | orderkey | orderkey_index
-------------------+----------+----------------
Customer#000008251 | 3020007  |        0
Customer#000009452 | 4043971  |        0
  (2 rows)
```

Veja a seguir um exemplo de iteração sobre uma matriz escalar.

```
CREATE TABLE bar AS SELECT json_parse('{"scalar_array": [1, 2.3, 45000000]}') AS data;

SELECT element, index FROM bar AS b, b.data.scalar_array AS element AT index;

 index | element
-------+----------
     0 | 1
     1 | 2.3
     2 | 45000000
(3 rows)
```

O exemplo a seguir itera sobre uma matriz de vários níveis. O exemplo usa várias cláusulas unnest para iterar nas matrizes mais internas. O AS da matriz `f.multi_level_array` itera sobre `multi_level_array`. O elemento AS da matriz é a iteração sobre as matrizes dentro de `multi_level_array`.

```
CREATE TABLE foo AS SELECT json_parse('[[1.1, 1.2], [2.1, 2.2], [3.1, 3.2]]') AS
 multi_level_array;

SELECT array, element FROM foo AS f, f.multi_level_array AS array, array AS element;

 element | array
---------+---------
```

```
 1.1     | [1.1,1.2]
 1.2     | [1.1,1.2]
 2.1     | [2.1,2.2]
 2.2     | [2.1,2.2]
 3.1     | [3.1,3.2]
 3.2     | [3.1,3.2]
(6 rows)
```

Para obter mais informações sobre a cláusula FROM, consulte Cláusula FROM. Para obter mais exemplos de desaninhamento de consultas SUPER, consulte Exemplos de uso de dados semiestruturados no Amazon Redshift.

Transformar colunas em linhas de objetos

Para transformar colunas em linhas de objetos, o Amazon Redshift usa a sintaxe PartiQL para iterar sobre objetos SUPER. Ele faz isso usando a cláusula FROM de uma consulta junto com a palavra-chave UNPIVOT. No exemplo a seguir, a expressão é o objeto `c.c_orders[0]`. A consulta de exemplo itera sobre cada atributo exibido pelo objeto.

```
SELECT attr as attribute_name, json_typeof(val) as value_type
FROM customer_orders_lineitem c, UNPIVOT c.c_orders[0] AS val AT attr
WHERE c_custkey = 9451;

 attribute_name  | value_type
-----------------+------------
 o_orderstatus   | string
 o_clerk         | string
 o_lineitems     | array
 o_orderdate     | string
 o_shippriority  | number
 o_totalprice    | number
 o_orderkey      | number
 o_comment       | string
 o_orderpriority | string
(9 rows)
```

Assim como no desaninhamento, a sintaxe da transformação de coluna em linhas é uma extensão da cláusula FROM. A diferença é que a sintaxe da transformação de colunas em linhas usa a palavra-chave UNPIVOT para indicar que está iterando sobre um objeto em vez de uma matriz. Ele usa AS `value_alias` para iteração sobre todos os valores dentro de um objeto e usa o AT `attribute_alias` para iterar sobre todos os atributos. Considere a seguinte sintaxe:

```
UNPIVOT expression AS value_alias [ AT attribute_alias ]
```

O Amazon Redshift aceita o uso de desagregação de objetos e desaninhamento de matriz em uma única cláusula FROM da seguinte forma:

```
SELECT attr as attribute_name, val as object_value
FROM customer_orders_lineitem c, c.c_orders AS o, UNPIVOT o AS val AT attr
WHERE c_custkey = 9451;
```

Ao usar a transformação de coluna em linha de objetos, o Amazon Redshift não oferece suporte a transformação de coluna em linha correlacionada. Especificamente, suponha que você tenha um caso em que há vários exemplos de transformação de coluna em linha em diferentes níveis de consulta e transformação de coluna em linha interna faz referência à externa. O Amazon Redshift não oferece suporte a este tipo de transformações múltiplas de coluna em linha.

Para obter mais informações sobre a cláusula FROM, consulte Cláusula FROM. Para obter exemplos de uso da rotação com o tipo SUPER, consulte Exemplos de uso de dados semiestruturados no Amazon Redshift.

Digitação dinâmica

A digitação dinâmica não requer fundição explícita de dados que são extraídos dos caminhos de ponto e colchetes. O Amazon Redshift usa a digitação dinâmica para processar dados SUPER sem esquema sem a necessidade de declarar os tipos de dados antes de usá-los em sua consulta. A digitação dinâmica usa os resultados da navegação em colunas de dados SUPER sem ter que convertê-las explicitamente nos tipos do Amazon Redshift. A digitação dinâmica é mais útil em junções e cláusulas GROUP BY. O exemplo a seguir usa uma instrução SELECT que não requer conversão explícita das expressões de ponto e colchetes para os tipos habituais do Amazon Redshift. Para obter informações sobre a compatibilidade e conversão de tipos, consulte Compatibilidade e conversão dos tipos.

Considere o exemplo a seguir, que procura linhas em que o status de um pedido é `shipped`:

```
SELECT c_orders[0].o_orderkey
FROM customer_orders_lineitem
WHERE c_orders[0].o_orderstatus = 'shipped';
```

O sinal de igualdade nesta consulta de exemplo é avaliado como `true` quando o valor de c_orders[0].o_orderstatus é a string ‘shipped’. Em todos os demais casos, o sinal de igualdade é

avaliado como `false`, incluindo os casos em que os argumentos da igualdade são diferentes tipos. Por exemplo, se o status do pedido for um número inteiro, sua linha não será selecionada.

Digitação dinâmica e estática

Sem usar a digitação dinâmica, você não pode determinar se c_orders [0] .o_orderstatus é uma string, um inteiro ou uma estrutura. Você só pode determinar que c_orders [0] .o_orderstatus é um tipo de dados SUPER, que pode ser um escalar, um array ou uma estrutura do Amazon Redshift. O tipo estático de c_orders [0] .o_orderstatus é um tipo de dados SUPER. Convencionalmente, um tipo é implicitamente um tipo estático no SQL.

O Amazon Redshift usa a digitação dinâmica para o processamento de dados sem esquema. Quando a consulta avalia os dados, c_orders [0] .o_orderstatus acaba por ser um tipo específico. Por exemplo, avaliar c_orders [0] .o_orderstatus no primeiro registro de customer_orders_lineitem pode resultar em um inteiro. A avaliação no segundo registro pode resultar em uma string. Estes são os tipos dinâmicos da expressão.

Ao usar um operador ou função SQL com expressões de ponto e colchetes que têm tipos dinâmicos, o Amazon Redshift produz resultados semelhantes ao uso do operador ou função SQL padrão com os respectivos tipos estáticos. Neste exemplo, quando o tipo dinâmico da expressão de caminho é uma string, a comparação com a string 'P' é significativa. Sempre que o tipo dinâmico de c_orders [0] .o_orderstatus é qualquer outro tipo de dados exceto ser uma string, a igualdade retorna false. Outras funções retornam null quando argumentos digitados incorretamente são usados.

O seguinte exemplo grava a consulta anterior com digitação estática:

```
SELECT c_custkey
FROM customer_orders_lineitem
WHERE CASE WHEN JSON_TYPEOF(c_orders[0].o_orderstatus) = 'string'
           THEN c_orders[0].o_orderstatus::VARCHAR = 'P'
           ELSE FALSE END;
```

Observe a seguinte distinção entre predicados de igualdade e predicados de comparação. No exemplo anterior, se você substituir o predicado de igualdade por um predicado menor que ou igual, a semântica produzirá null em vez de false.

```
SELECT c_orders[0]. o_orderkey
FROM customer_orders_lineitem
WHERE c_orders[0].o_orderstatus <= 'P';
```

Neste exemplo, se c_orders [0] .o_orderstatus for uma string, o Amazon Redshift retornará true se for alfabeticamente igual ou menor que 'P'. O Amazon Redshift retornará false se for alfabeticamente maior que 'P'. No entanto, se c_orders [0] .o_orderstatus não for uma string, o Amazon Redshift retornará null, pois o Amazon Redshift não pode comparar valores de tipos diferentes, conforme mostrado na consulta a seguir:

```
SELECT c_custkey
FROM customer_orders_lineitem
WHERE CASE WHEN JSON_TYPEOF(c_orders[0].o_orderstatus) = 'string'
           THEN c_orders[0].o_orderstatus::VARCHAR <= 'P'
           ELSE NULL END;
```

A digitação dinâmica não exclui de comparações de tipos que são minimamente comparáveis. Por exemplo, você pode converter os tipos escalares CHAR e VARCHAR do Amazon Redshift para SUPER. Eles são comparáveis como strings, inclusive ignorando caracteres de espaço em branco à direita semelhantes aos tipos CHAR e VARCHAR do Amazon Redshift. Da mesma forma, números inteiros, decimais e valores de ponto flutuante são comparáveis como valores SUPER. Especificamente para colunas decimais, cada valor também pode ter uma escala diferente. O Amazon Redshift ainda os considera como tipos dinâmicos.

O Amazon Redshift também oferece suporte à igualdade em objetos e arrays que são avaliados como profundos iguais, como avaliar profundamente em objetos ou arrays e comparar todos os atributos. Use profundo igual com cautela, porque o processo de realização de igual profundo pode ser demorado.

Usar a digitação dinâmica para junções

Para junções, a digitação dinâmica corresponde automaticamente aos valores com diferentes tipos dinâmicos sem executar uma longa análise CASE WHEN para descobrir quais tipos de dados podem aparecer. Por exemplo, suponha que sua organização alterou o formato que estava usando para chaves de peça ao longo do tempo.

As chaves iniciais de parte inteira emitidas são substituídas por chaves de partes de string, como 'A55', e posteriormente substituídas por chaves de partes de matriz, como ['X', 10] combinando uma string e um número. O Amazon Redshift não precisa executar uma longa análise de caso sobre chaves de peça e pode usar joins como mostrado no exemplo a seguir.

```
SELECT c.c_name
    ,l.l_extendedprice
    ,l.l_discount
```

```
FROM customer_orders_lineitem c
    ,c.c_orders o
    ,o.o_lineitems l
    ,supplier_partsupp s
    ,s.s_partsupps ps
WHERE l.l_partkey = ps.ps_partkey
AND c.c_nationkey = s.s_nationkey
ORDER BY c.c_name;
```

O exemplo a seguir mostra quão complexa e ineficiente a mesma consulta pode ser sem usar a digitação dinâmica:

```
SELECT c.c_name
    ,l.l_extendedprice
    ,l.l_discount
FROM customer_orders_lineitem c
    ,c.c_orders o
    ,o.o_lineitems l
    ,supplier_partsupp s
    ,s.s_partsupps ps
WHERE CASE WHEN IS_INTEGER(l.l_partkey) AND IS_INTEGER(ps.ps_partkey)
           THEN l.l_partkey::integer = ps.ps_partkey::integer
           WHEN IS_VARCHAR(l.l_partkey) AND IS_VARCHAR(ps.ps_partkey)
           THEN l.l_partkey::varchar = ps.ps_partkey::varchar
           WHEN IS_ARRAY(l.l_partkey) AND IS_ARRAY(ps.ps_partkey)
                AND IS_VARCHAR(l.l_partkey[0]) AND IS_VARCHAR(ps.ps_partkey[0])
                AND IS_INTEGER(l.l_partkey[1]) AND IS_INTEGER(ps.ps_partkey[1])
           THEN l.l_partkey[0]::varchar = ps.ps_partkey[0]::varchar
                AND l.l_partkey[1]::integer = ps.ps_partkey[1]::integer
           ELSE FALSE END
AND c.c_nationkey = s.s_nationkey
ORDER BY c.c_name;
```

Consulta sem distinção entre maiúsculas e minúsculas

Você pode realizar comparações de strings de caracteres sem distinção entre maiúsculas e minúsculas em dados SUPER usando a função COLLATE ou definindo o agrupamento em nível de coluna ou de banco de dados. Para acessar mais informações sobre como definir o agrupamento na criação da tabela, consulte CRIAR TABELA. Para acessar informações sobre o comportamento do agrupamento com operadores e funções de dados SUPER, consulte Comportamento do agrupamento.

O exemplo a seguir usa a função COLLATE em valores de string extraídos dos dados SUPER.

```
CREATE TABLE events (data SUPER);
INSERT INTO events VALUES (JSON_PARSE('{"status": "Active", "name": "Event1"}'));
INSERT INTO events VALUES (JSON_PARSE('{"status": "ACTIVE", "name": "Event2"}'));
INSERT INTO events VALUES (JSON_PARSE('{"status": "active", "name": "Event3"}'));

SELECT data.name FROM events
WHERE COLLATE(data.status::VARCHAR, 'case_insensitive') = 'active';

 name
----------
 "Event1"
 "Event2"
 "Event3"
(3 rows)
```

Você também pode definir uma coluna SUPER com agrupamento sem distinção entre maiúsculas e minúsculas na criação da tabela. Nesse caso, as comparações de strings na coluna não diferenciam maiúsculas de minúsculas.

```
CREATE TABLE events_ci (data SUPER COLLATE CASE_INSENSITIVE);
INSERT INTO events_ci VALUES (JSON_PARSE('{"status": "Active"}'));
INSERT INTO events_ci VALUES (JSON_PARSE('{"status": "ACTIVE"}'));

SELECT * FROM events_ci WHERE data.status::VARCHAR = 'active';

 data
-----------------------
 {"status":"Active"}
 {"status":"ACTIVE"}
(2 rows)
```

Semântica lax

Por padrão, as operações de navegação em valores SUPER retornam null em vez de retornar um erro quando a navegação é inválida. A navegação do objeto é inválida se o valor SUPER não for um objeto ou se o valor SUPER for um objeto, mas não contiver o nome do atributo usado na consulta. Por exemplo, a consulta a seguir acessa um nome de atributo inválido na coluna de dados SUPER cdata:

```
SELECT c.c_orders.something FROM customer_orders_lineitem c;
```

Navegação de array retorna null se o valor SUPER não é um array ou o índice de array está fora dos limites. A consulta a seguir retorna null porque c_orders [1] [1] está fora dos limites.

```
SELECT c.c_orders[1][1] FROM customer_orders_lineitem c;
```

A semântica lax é especialmente útil quando se usa a digitação dinâmica para lançar um valor SUPER. Transferir um valor SUPER para o tipo errado retorna null em vez de um erro se a conversão for inválida. Por exemplo, a consulta a seguir retorna null porque não pode converter o valor de sequência de caracteres 'Good' do atributo o_orderstatus de objeto para INTEGER. O Amazon Redshift retorna um erro para uma transmissão VARCHAR para INTEGER, mas não para uma transmissão SUPER.

```
SELECT c.c_orders.o_orderstatus::integer FROM customer_orders_lineitem c;
```

Order by (Ordenar por)

O Amazon Redshift não define comparações SUPER entre valores com diferentes tipos dinâmicos. Um valor SUPER que é uma cadeia não é menor nem maior do que um valor SUPER que é um número. Para usar as cláusulas ORDER BY com colunas SUPER, o Amazon Redshift define um pedido total entre diferentes tipos a serem observados quando o Amazon Redshift classifica os valores SUPER usando as cláusulas ORDER BY. A ordem entre tipos dinâmicos é booleano, número, string, array, objeto.

Para obter um exemplo de uso de GROUP BY e ORDER BY em uma consulta SUPER, consulte Filtrar dados semiestruturados.

Operadores e funções

Com o Amazon Redshift, é possível realizar operações de analytics avançada em grandes conjuntos de dados que usam dados SUPER por meio de operadores e funções. Os operadores e funções para dados SUPER são constructos de SQL que permitem análise e manipulação complexas de dados semiestruturados armazenados em tabelas do Amazon Redshift.

As seções a seguir abordarão a sintaxe, os exemplos e as práticas recomendadas para usar operadores e funções em dados SUPER no Amazon Redshift e aproveitar todo o potencial de seus dados semiestruturados.

Operadores aritméticos

Os valores SUPER dão suporte a todos os operadores aritméticos básicos +, -, *,/,% usando a digitação dinâmica. O tipo resultante da operação permanece como SUPER. Para todos os operadores, exceto para o operador binário +, os operandos de entrada devem ser números. Caso contrário, o Amazon Redshift retorna nulo. A distinção entre valores decimais e de ponto flutuante é mantida quando o Amazon Redshift executa esses operadores e o tipo dinâmico não é alterado. No entanto, alterações de escala decimal quando você usa multiplicações e divisões. Os estouros aritméticos ainda causam erros de consulta, eles não são alterados para nulo. Operador binário + executa adição se as entradas são números ou concatenação se as entradas são string. Se um operando é uma string e o outro operando é um número, o resultado é nulo. Operadores de prefixo unário + e - retorna null se o valor SUPER não for um número, como mostrado no seguinte exemplo:

```
SELECT (c_orders[0]. o_orderkey + 0.5) * c_orders[0]. o_orderkey / 10 AS math FROM
 customer_orders_lineitem;
            math
----------------------------
 1757958232200.1500
(1 row)
```

A digitação dinâmica permite que os valores decimais em SUPER tenham escalas diferentes. O Amazon Redshift trata valores decimais como se fossem tipos estáticos diferentes e permite todas as operações matemáticas. O Amazon Redshift calcula a escala resultante dinamicamente com base nas escalas dos operandos. Se um dos operandos for um número de ponto flutuante, o Amazon Redshift promoverá o outro operando para um número de ponto flutuante e gerará o resultado como um número de ponto flutuante.

Funções aritméticas

O Amazon Redshift oferece suporte às seguintes funções aritméticas para colunas SUPER. Eles retornam null se a entrada não for um número:

- FLOOR. Para obter mais informações, consulte Função FLOOR.
- CEIL e CEILING Para obter mais informações, consulte Função CEILING (ou CEIL).
- ROUND. Para obter mais informações, consulte Função ROUND.
- TRUNC. Para obter mais informações, consulte Função TRUNC.
- ABS. Para obter mais informações, consulte Função ABS.

O seguinte exemplo usa funções aritméticas para consultar dados:

```
SELECT x, FLOOR(x), CEIL(x), ROUND(x)
FROM (
    SELECT (c_orders[0]. o_orderkey + 0.5) * c_orders[0].o_orderkey / 10 AS x
    FROM customer_orders_lineitem
    );

         x          |     floor     |     ceil      |     round
--------------------+---------------+---------------+---------------
 1389636795898.0500  | 1389636795898  | 1389636795899  | 1389636795898
```

A função ABS mantém a escala da entrada decimal enquanto FLOOR, CEIL. O ROUND elimina a escala do decimal de entrada.

Funções de array

O Amazon Redshift dá suporte às seguintes funções utilitárias e de composição de matriz:

- ARRAY. Para obter mais informações, consulte Função ARRAY.
- ARRAY_CONCAT. Para obter mais informações, consulte Função ARRAY_CONCAT.
- ARRAY_CONTAINS. Para obter mais informações, consulte Função ARRAY_CONTAINS.
- ARRAY_DISTINCT. Para obter mais informações, consulte Função ARRAY_DISTINCT.
- ARRAY_EXCEPT. Para obter mais informações, consulte Função ARRAY_EXCEPT.
- ARRAY_FLATTEN. Para obter mais informações, consulte Função ARRAY_FLATTEN.
- ARRAY_INTERSECTION. Para obter mais informações, consulte Função ARRAY_INTERSECTION.
- ARRAY_POSITION. Para obter mais informações, consulte Função ARRAY_POSITION.
- ARRAY_POSITIONS. Para obter mais informações, consulte Função ARRAY_POSITIONS.
- ARRAY_SORT. Para obter mais informações, consulte Função ARRAY_SORT.
- ARRAY_UNION. Para obter mais informações, consulte Função ARRAY_UNION.
- ARRAYS_OVERLAP. Para obter mais informações, consulte Função ARRAYS_OVERLAP.
- GET_ARRAY_LENGTH. Para obter mais informações, consulte Função GET_ARRAY_LENGTH.
- SPLIT_TO_ARRAY. Para obter mais informações, consulte Função SPLIT_TO_ARRAY.
- SUBARRAY. Para obter mais informações, consulte Função SUBARRAY.

Você pode construir arrays SUPER a partir de valores nos tipos de dados do Amazon Redshift usando a função ARRAY, incluindo outros valores SUPER. O seguinte exemplo usa a função variádica ARRAY:

```
SELECT ARRAY(1, c.c_custkey, NULL, c.c_name, 'abc') FROM customer_orders_lineitem c;
                                array
 ------------------------------------------------------
[1,8401,null,""Customer#000008401"",""abc""]
[1,9452,null,""Customer#000009452"",""abc""]
[1,9451,null,""Customer#000009451"",""abc""]
[1,8251,null,""Customer#000008251"",""abc""]
[1,5851,null,""Customer#000005851"",""abc""]
(5 rows)
```

O seguinte exemplo usa concatenação de array com a função ARRAY_CONCAT:

```
SELECT ARRAY_CONCAT(JSON_PARSE('[10001,10002]'),JSON_PARSE('[10003,10004]'));

             array_concat
-------------------------------------
 [10001,10002,10003,10004]
(1 row)
```

O seguinte exemplo usa manipulação de array com a função SUBARRAY que retorna um subconjunto do array de entrada.

```
SELECT SUBARRAY(ARRAY('a', 'b', 'c', 'd', 'e', 'f'), 2, 3);

   subarray
---------------
 ["c","d","e"]
(1 row))
```

O seguinte exemplo mescla vários níveis de arrays em um único array usando ARRAY_FLATTEN:

```
SELECT x, ARRAY_FLATTEN(x) FROM (SELECT ARRAY(1, ARRAY(2, ARRAY(3, ARRAY())))  AS x);

     x           | array_flatten
 ----------------+---------------
 [1,[2,[3,[]]]]  | [1,2,3]
(1 row)
```

As funções de array ARRAY_CONCAT e ARRAY_FLATTEN usam regras de digitação dinâmica. Eles retornam um nulo em vez de um erro se a entrada não for um array. A função GET_ARRAY_LENGTH retorna o comprimento de um array SUPER dado um objeto ou caminho de array.

```
SELECT c_name
FROM customer_orders_lineitem
WHERE GET_ARRAY_LENGTH(c_orders) = (
    SELECT MAX(GET_ARRAY_LENGTH(c_orders))
    FROM customer_orders_lineitem
    );
```

O exemplo a seguir divide uma string em um array de strings usando SPLIT_TO_ARRAY. A função usa um delimitador como um parâmetro opcional. Se nenhum delimitador estiver ausente, o padrão será uma vírgula.

```
SELECT SPLIT_TO_ARRAY('12|345|6789', '|');

   split_to_array
---------------------
 ["12","345","6789"]
(1 row)
```

Comportamento do agrupamento

Com SUPER, você pode definir o agrupamento de uma coluna com CREATE TABLE, acessar o agrupamento padrão definido com CREATE DATABASE e definir o agrupamento de uma expressão com a função COLLATE.

A configuração de agrupamento se aplica a todos os operadores de comparação e valores de string armazenados em SUPER, sejam eles valores de string, strings dentro de matrizes SUPER ou valores de um objeto SUPER. Para objetos SUPER, o comportamento do agrupamento se aplica somente aos valores e não aos atributos. Por exemplo, uma comparação de valores que têm o agrupamento sem distinção entre maiúsculas e minúsculas de `{"attribute": "a"} = {"attribute": "A"}` exibe true, enquanto `{"attribute": "a"} = {"ATTRIBUTE": "a"}` exibe false.

Funções de informação

As colunas de dados SUPER suportam funções de inspeção que retornam o tipo dinâmico e outras informações de tipo sobre o valor SUPER. A função escalar JSON_TYPEOF retorna um VARCHAR com valores booleanos, number, string, object, array ou null, dependendo do tipo dinâmico do valor SUPER. O Amazon Redshift oferece suporte às seguintes funções booleanas para colunas de dados SUPER:

- Função IS_ARRAY
- Função IS_BIGINT
- Função IS_CHAR
- Função IS_DECIMAL
- Função IS_FLOAT
- Função IS_INTEGER
- Função IS_OBJECT
- Função IS_SCALAR
- Função IS_SMALLINT
- Função IS_VARCHAR

Para obter mais informações sobre funções de informações do tipo SUPER, consulte Funções de informação de tipo SUPER.

Funções de objetos

Veja a seguir as funções de objeto do SQL compatíveis com o Amazon Redshift para criar e operar em objetos do tipo SUPER:

- Função GET_NUMBER_ATTRIBUTES
- Função LOWER_ATTRIBUTE_NAMES
- Função OBJECT
- Função OBJECT_TRANSFORM
- Função UPPER_ATTRIBUTE_NAMES

Para ter mais informações sobre funções de objeto, consulte Funções de objetos.

Funções de string

Para usar funções de string com literais de string no tipo de dados SUPER, você deve converter o literal de string em tipo de string antes de aplicar as funções. As funções exibirão null se a entrada não for um literal de string.

Funções de string agora comportam até 16 milhões de bytes.

O exemplo a seguir usa SUBSTRING para extrair a versão prévia de uma string com 5 milhões de bytes de tamanho armazenada em um objeto SUPER JSON.

```
CREATE TABLE customer_data (
   customer_id INT,
   profile SUPER
);

INSERT INTO customer_data VALUES (
   1,
   JSON_PARSE('{"name": "John Doe", "description": "' || REPEAT('A', 5000000) || '"}')
);

SELECT
   customer_id,
   profile.name::VARCHAR AS name,
   SUBSTRING(profile.description::VARCHAR, 1, 50) AS description_preview
FROM customer_data;

 customer_id | name     | description_preview
-------------+----------+----------------------------------------------------
         1 | John Doe | AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

O seguinte exemplo demonstra as funções LEFT, RIGHT e CONCAT em literais de string de uma matriz SUPER:

```
CREATE TABLE documents (
   doc_id INT,
   chapters SUPER
);

INSERT INTO documents VALUES (
   1,
```

```
   JSON_PARSE('["' || REPEAT('hello', 400000) || '", "' || REPEAT('world', 600000) ||
 '"]')
);

SELECT
   doc_id,
   LEFT(chapters[0]::VARCHAR, 20) AS chapter1_start,
   RIGHT(chapters[1]::VARCHAR, 20) AS chapter2_end,
   LEN(CONCAT(chapters[0]::VARCHAR, chapters[1]::VARCHAR)) AS concat_size
FROM documents;

 doc_id |    chapter1_start    |     chapter2_end     | concat_size
--------+----------------------+----------------------+-------------
      1 | hellohellohellohello | worldworldworldworld |     5000000
```

O seguinte exemplo armazena uma string autônoma em SUPER:

```
CREATE TABLE text_storage (
   text_id INT,
   content SUPER
);

INSERT INTO text_storage VALUES
   (1, REPEAT('A', 8000000)),
   (2, REPEAT('B', 16000000));

SELECT
   text_id,
   LEN(content::VARCHAR) AS content_length
FROM text_storage;

 text_id | content_length
---------+----------------
      1 |        8000000
      2 |       16000000
```

Configurações SUPER

Você pode configurar seus dados SUPER para cenários específicos. As seções a seguir apresentam detalhes sobre como escolher e aplicar as configurações apropriadas de dados SUPER com base em seus requisitos de formato de dados.

Tópicos

- Modos lax e estrito para SUPER
- Acessar campos JSON com nomes ou atributos de campo em maiúsculas ou mistas
- Opções de análise para SUPER

Modos lax e estrito para SUPER

Quando consulta dados SUPER, a expressão de caminho pode não corresponder à estrutura de dados SUPER real. Se você tentar acessar um membro inexistente de um objeto ou elemento de uma matriz, o Amazon Redshift retornará um valor NULL se sua consulta for executada no modo lax padrão. Se você executar sua consulta no modo estrito, o Amazon Redshift retornará um erro. Os seguintes parâmetros de sessão podem ser definidos para ativar ou desativar o modo lax.

O exemplo a seguir usa parâmetros de sessão para habilitar o modo lax.

```
SET navigate_super_null_on_error=ON;  --default lax mode for navigation

SET cast_super_null_on_error=ON;  --default lax mode for casting

SET parse_super_null_on_error=OFF;  --default strict mode for ingestion
```

Acessar campos JSON com nomes ou atributos de campo em maiúsculas ou mistas

Quando os nomes de atributos JSON estão em letras maiúsculas ou mistas, você deve ser capaz de percorrer estruturas do tipo SUPER de forma a diferenciar maiúsculas e minúsculas. Para fazer isso, você pode configurar `enable_case_sensitive_super_attribute` como TRUE e usar nomes de atributos em letras maiúsculas ou mistas diretamente em suas consultas, sem colocá-los entre aspas duplas.

O exemplo a seguir mostra como definir `enable_case_sensitive_super_attribute` para consultar dados.

```
SET enable_case_sensitive_super_attribute to TRUE;

-- Accessing JSON attribute names with uppercase and mixed-case names
SELECT json_table.data.ITEMS.Name,
       json_table.data.price
FROM
  (SELECT json_parse('{"ITEMS":{"Name":"TV"}, "price": 345}') AS data) AS json_table;

 Name | price
------+-------
 "TV" | 345
(1 row)

RESET enable_case_sensitive_super_attribute;

-- After resetting the above configuration, the following query accessing JSON
 attribute names with uppercase and mixed-case names should return null (if in lax
 mode).
SELECT json_table.data.ITEMS.Name,
       json_table.data.price
FROM
  (SELECT json_parse('{"ITEMS":{"Name":"TV"}, "price": 345}') AS data) AS json_table;

 name | price
------+-------
      | 345
(1 row)
```

Como alternativa, você pode configurar `enable_case_sensitive_identifier` como TRUE e envolver os nomes de atributos em letras maiúsculas e minúsculas com aspas duplas. Para obter mais informações, consulte enable_case_sensitive_identifier.

Opções de análise para SUPER

Quando você usa a função JSON_PARSE para analisar strings JSON em valores SUPER, determinadas restrições se aplicam.

- O mesmo nome de atributo não pode aparecer no mesmo objeto, mas pode aparecer em um objeto aninhado. A opção de configuração `json_parse_dedup_attributes` permite que JSON_PARSE mantenha apenas a última ocorrência de atributos duplicados em vez de retornar um erro.

- Literais de string individuais no objeto SUPER têm um limite de tamanho de 16 milhões de bytes. A opção de configuração `json_parse_truncate_strings` permite que JSON_PARSE() trunque automaticamente strings maiores que esse limite sem retornar um erro. Esse comportamento afeta somente valores de string e não nomes de atributo.

Para obter mais informações sobre como usar a função JSON_PARSE, consulte Função JSON_PARSE.

O exemplo a seguir mostra como configurar a opção de configuração `json_parse_dedup_attributes` para o comportamento padrão de retorno de erro para atributos duplicados.

```
SET json_parse_dedup_attributes=OFF;  --default behavior of returning error instead of
 de-duplicating attributes
```

O exemplo a seguir mostra como configurar a opção de configuração `json_parse_truncate_strings` para o comportamento padrão de retorno de erro para strings maiores que esse limite.

```
SET json_parse_truncate_strings=OFF;  --default behavior of returning error instead of
 truncating strings
```

Limitações

Com o Amazon Redshift, é possível trabalhar com o tipo de dados SUPER para armazenar e consultar dados semiestruturados, como JSON, Avro ou Ion. As limitações do tipo de dados SUPER estão relacionadas aos limites e restrições existentes quando se usa esse tipo de dados no Amazon Redshift. As seções a seguir apresentam detalhes sobre as limitações específicas do tipo de dados SUPER, como tamanho máximo, níveis de aninhamento e tipos de dados compatíveis no conjunto de dados semiestruturados.

- Você não pode definir colunas SUPER como uma distribuição ou chave de classificação.
- Um objeto SUPER individual pode armazenar até 16 MB de dados.
- A profundidade máxima de aninhamento para matrizes e estruturas do tipo de dados SUPER é 1.000.
- Qualquer string literal armazenada em um único objeto SUPER é limitada a 16 milhões de bytes.

- Caso contrário, um valor individual dentro de um objeto SUPER é limitado ao comprimento máximo do tipo do Amazon Redshift correspondente.
- Ao usar o comando COPY para carregar dados em uma coluna SUPER a partir de um formato baseado em linhas (TEXT, CSV, JSON, AVRO), toda a linha deve caber no buffer do scanner do COPY, que é de 64 MB por padrão. O limite de 64 MB do scanner não se aplica a formatos colunares (PARQUET, ORC). Em todos os casos, cada objeto SUPER está limitado a 16 MB, independentemente do formato de origem.
- A opção TRUNCATECOLUMNS do comando COPY, quando compatível, aplica-se apenas a colunas VARCHAR e não trunca valores de cadeia de caracteres em colunas SUPER.
- Você não pode executar operações de atualização parcial ou transformação em colunas SUPER.
- Você não pode usar o tipo de dados SUPER e seu alias em junções à direita ou junções externas completas.
- O tipo de dados SUPER não suporta XML como formato de serialização de entrada ou saída.
- Na cláusula FROM de uma subconsulta (que está correlacionada ou não) que faz referência a uma variável de tabela para desaninhamento, a consulta só pode se referir à sua tabela pai e não a outras tabelas.
- Não é possível realizar o desaninhamento nem a remoção dinâmica de objetos em colunas SUPER produzidas por uma expressão de tabela comum (CTE) recursiva (`WITH RECURSIVE`). Para ter mais informações sobre CTEs recursivas, consulte Expressões de tabela comuns recursivas.
- Limitações de conversão

  Os valores SUPER podem ser convertidos de e para outros tipos de dados, com as seguintes exceções:

  - O Amazon Redshift não diferencia inteiros e decimais da escala 0.
  - Se a escala não for zero, o tipo de dados SUPER terá o mesmo comportamento de outros tipos de dados do Amazon Redshift, exceto que o Amazon Redshift converte erros relacionados ao Super-em nulo, conforme mostrado no exemplo a seguir.

```
SELECT 5::bool;
 bool
-------
 True
(1 row)

SELECT 5::decimal::bool;
```

```
ERROR:  cannot cast type numeric to boolean

SELECT 5::super::bool;
 bool
-------
 True
(1 row)

SELECT 5.0::bool;
ERROR:  cannot cast type numeric to boolean

SELECT 5.0::super::bool;
 bool
-------
(1 row)
```

- O Amazon Redshift não converte os tipos de data e hora para o tipo de dados SUPER. O Amazon Redshift só pode converter os tipos de dados de data e hora do tipo de dados SUPER, conforme mostrado no exemplo a seguir.

```
SELECT o.o_orderdate FROM customer_orders_lineitem c,c.c_orders o;
  order_date
------------------
 "2001-09-08"
(1 row)


SELECT JSON_TYPEOF(o.o_orderdate) FROM customer_orders_lineitem c,c.c_orders o;
 json_typeof
-----------------
 string
(1 row)


SELECT o.o_orderdate::date FROM customer_orders_lineitem c,c.c_orders o;
 order_date
----------------
 2001-09-08
(1 row)


--date/time cannot be cast to super
SELECT '2019-09-09'::date::super;
```

```
ERROR:  cannot cast type date to super
```

- Converter de valores não escalares (objeto e array) para string retorna NULL. Para serializar adequadamente esses valores não escalares, não os converta. Em seu lugar, use `json_serialize` para converter valores não escalares. A função `json_serialize` retorna um varchar. Normalmente, você não precisa converter valores não escalares para o varchar, pois o Amazon Redshift serializa implicitamente, conforme mostrado no primeiro exemplo a seguir.

```
SELECT r_nations FROM region_nations WHERE r_regionkey=300;
   r_nations
----------------
 [1,"abc",null]
(1 row)

SELECT r_nations::varchar FROM region_nations WHERE r_regionkey=300;
 r_nations
-------------
(1 row)

SELECT JSON_SERIALIZE(r_nations) FROM region_nations WHERE r_regionkey=300;
 json_serialize
-----------------
 [1,"abc",null]
(1 row)
```

- O Amazon Redshift não oferece suporte a funções voláteis, como RANDOF () ou TIMEOFDAY (), em subconsultas que unnest uma tabela externa ou um lado esquerdo (LHS) de funções IN com tais subconsultas.

Tipo de dados SUPER com visões materializadas

Com o Amazon Redshift, é possível usar visões materializadas para melhorar o desempenho e a flexibilidade de consultas executadas em tipo de dados SUPER. Com o tipo de dados SUPER, é possível armazenar um superconjunto das colunas das tabelas base em uma visualização materializada, permitindo consultar diretamente a visualização materializada sem fazer uma junção com as tabelas base. As seções a seguir mostram como criar e usar visões materializadas com o tipo de dados SUPER no Amazon Redshift.

O Amazon Redshift dá suporte a visões materializadas que incorporam colunas de tipo de dados SUPER e consultas PartiQL. As visões materializadas podem ser atualizadas incrementalmente, enquanto o Amazon Redshift atualiza somente os dados que foram alterados nas tabelas base desde a última operação de atualização. Essa abordagem de atualização seletiva torna o processo de atualização mais eficiente do que os recálculos completos. Para obter mais informações sobre visões materializadas, consulte Visões materializadas no Amazon Redshift.

Acelerar consultas do PartiQL

Você pode usar visões materializadas para acelerar consultas do PartiQL que navegam e/ou aninhem dados hierárquicos em colunas SUPER. Ao criar uma ou mais visões materializadas para destruir os valores SUPER em várias colunas e utilizar a organização colunar das consultas analíticas do Amazon Redshift, você pode essencialmente extrair e normalizar dados aninhados. O nível de normalização depende de quanto esforço você colocar para transformar os dados SUPER em dados colunares convencionais.

Os tópicos a seguir mostram exemplos de como dividir ou fragmentar dados complexos em colunas menores, bem como criar colunas escalares a partir de dados fragmentados para melhorar o desempenho.

Tópicos

- Fragmentar dados semiestruturados em colunas SUPER com visões materializadas
- Criar colunas escalares do Amazon Redshift a partir de dados fragmentados

Fragmentar dados semiestruturados em colunas SUPER com visões materializadas

Com o Amazon Redshift, é possível melhorar o desempenho das consultas fragmentando os dados em colunas SUPER por meio de visões materializadas. A fragmentação se refere ao processo de dividir tipos de dados complexos, como JSON ou XML semiestruturado, em colunas menores e mais niveladas. As colunas SUPER são uma forma especializada de armazenamento colunar otimizado para verificar rapidamente os dados fragmentados.

As seções a seguir descrevem as etapas e considerações para fragmentar dados em colunas SUPER usando visões materializadas no Amazon Redshift.

O exemplo a seguir mostra uma visão materializada que fragmenta os dados aninhados com as colunas resultantes ainda sendo o tipo de dados SUPER.

```
SELECT c.c_name, o.o_orderstatus
FROM customer_orders_lineitem c, c.c_orders o;
```

O exemplo a seguir mostra uma visão materializada que cria colunas escalares convencionais do Amazon Redshift com base nos dados fragmentados.

```
SELECT c.c_name, c.c_orders[0].o_totalprice
FROM customer_orders_lineitem c;
```

Você pode criar uma única visão materializada super_mv para acelerar ambas as consultas.

Para responder à primeira consulta, você deve materializar o atributo o_orderstatus. Você pode omitir o atributo c_name porque ele não envolve navegação aninhada nem desaninhamento. Você também deve incluir na visão materializada o atributo c_custkey de customer_orders_lineitem para poder unir a tabela-base com a visão materializada.

Para responder à segunda consulta, você também deve materializar o atributo o_totalprice e o índice de array o_idx de c_orders. Assim, você pode acessar o índice 0 de c_orders.

```
CREATE MATERIALIZED VIEW super_mv distkey(c_custkey) sortkey(c_custkey) AS (
  SELECT c_custkey, o.o_orderstatus, o.o_totalprice, o_idx
  FROM customer_orders_lineitem c, c.c_orders o AT o_idx
);
```

Os atributos o_orderstatus e o_totalprice da visão materializada super_mv são SUPER.

A visão materializada super_mv será atualizada incrementalmente após alterações na tabela base customer_orders_lineitem.

```
REFRESH MATERIALIZED VIEW super_mv;
INFO: Materialized view super_mv was incrementally updated successfully.
```

Para reescrever a primeira consulta PartiQL como uma consulta SQL regular, junte customer_orders_lineitem com super_mv da seguinte forma.

```
SELECT c.c_name, v.o_orderstatus
FROM customer_orders_lineitem c
JOIN super_mv v ON c.c_custkey = v.c_custkey;
```

Da mesma forma, você pode reescrever a segunda consulta PartiQL. O exemplo a seguir usa um filtro em o_idx = 0.

```
SELECT c.c_name, v.o_totalprice
FROM customer_orders_lineitem c
JOIN super_mv v ON c.c_custkey = v.c_custkey
WHERE v.o_idx = 0;
```

No comando CREATE MATERIALIZED VIEW, especifique c_custkey como chave de distribuição e chave de classificação para super_mv. O Amazon Redshift executa uma junção de mesclagem eficiente, supondo que c_custkey também seja a chave de distribuição e a chave de classificação de customer_orders_lineitem. Se esse não for o caso, você pode especificar c_custkey como a chave de classificação e a chave de distribuição de customer_orders_lineitem da forma a seguir.

```
ALTER TABLE customer_orders_lineitem
ALTER DISTKEY c_custkey, ALTER SORTKEY (c_custkey);
```

Use a instrução EXPLAIN para verificar se o Amazon Redshift realiza uma junção de mesclagem nas consultas reescritas.

```
EXPLAIN
      SELECT c.c_name, v.o_orderstatus
      FROM customer_orders_lineitem c JOIN super_mv v ON c.c_custkey = v.c_custkey;

      QUERY PLAN

 ------------------------------------------------------------------------------------------------
      XN Merge Join DS_DIST_NONE  (cost=0.00..34701.82 rows=1470776 width=27)
      Merge Cond: ("outer".c_custkey = "inner".c_custkey)
      ->  XN Seq Scan on mv_tbl__super_mv__0 derived_table2  (cost=0.00..14999.86
 rows=1499986 width=13)
      ->  XN Seq Scan on customer_orders_lineitem c  (cost=0.00..999.96 rows=99996
 width=30)
      (4 rows)
```

Criar colunas escalares do Amazon Redshift a partir de dados fragmentados

Dados sem esquema armazenados no SUPER podem afetar a performance do Amazon Redshift. Por exemplo, filtrar predicados ou condições de junção como varreduras restritas de intervalo não

podem usar efetivamente mapas de zona. Usuários e ferramentas de BI podem usar exibições materializadas como a apresentação convencional dos dados e aumentar a performance das consultas analíticas.

A consulta a seguir verifica a visão materializada `super_mv` e filtros no `o_orderstatus`.

```
SELECT c.c_name, v.o_totalprice
FROM customer_orders_lineitem c
JOIN super_mv v ON c.c_custkey = v.c_custkey
WHERE v.o_orderstatus = 'F';
```

Inspecione `stl_scan` para verificar se o Amazon Redshift não pode usar efetivamente mapas de zona na varredura restrita por intervalo em `o_orderstatus`.

```
SELECT slice, is_rrscan FROM stl_scan
WHERE query = pg_last_query_id() AND perm_table_name LIKE '%super_mv%';

 slice | is_rrscan
-------+-----------
     0 | f
     1 | f
     5 | f
     4 | f
     2 | f
     3 | f
(6 rows)
```

O exemplo a seguir adapta a visão materializada `super_mv` para criar colunas escalares fora dos dados fragmentados. Nesse caso, o Amazon Redshift converte `o_orderstatus` de SUPER para VARCHAR. Além disso, especifica `o_orderstatus` como a chave de classificação para o `super_mv`.

```
CREATE MATERIALIZED VIEW super_mv distkey(c_custkey) sortkey(c_custkey, o_orderstatus)
 AS (
  SELECT c_custkey, o.o_orderstatus::VARCHAR AS o_orderstatus, o.o_totalprice, o_idx
  FROM customer_orders_lineitem c, c.c_orders o AT o_idx
);
```

Depois de executar novamente a consulta, verifique se o Amazon Redshift agora pode usar mapas de zona.

```
SELECT v.o_totalprice
FROM super_mv v
WHERE v.o_orderstatus = 'F';
```

Você pode verificar se a varredura com restrição de intervalo agora usa mapas de zona da seguinte maneira.

```
SELECT slice, is_rrscan FROM stl_scan
WHERE query = pg_last_query_id() AND perm_table_name LIKE '%super_mv%';

 slice | is_rrscan
-------+-----------
     0 | t
     1 | t
     2 | t
     3 | t
     4 | t
     5 | t
(6 rows)
```

Exemplos de uso de dados semiestruturados no Amazon Redshift

Os exemplos a seguir demonstram como trabalhar com dados semiestruturados no Amazon Redshift usando a sintaxe PartiQL. Você criará uma tabela de amostra para carregar um conjunto de amostras de dados semiestruturados e, em seguida, consultará objetos de dados semiestruturados em vários casos de uso.

> **Note**
>
> Recomendamos que você defina a opção de configuração `enable_case_sensitive_super_attribute` como verdadeiro antes de trabalhar com o tipo de dados SUPER. Para obter mais informações, consulte enable_case_sensitive_super_attribute.

Carregamento de dados semiestruturados

As instruções a seguir criam uma tabela de amostra e carregam um objeto JSON de amostra na coluna `all_data` SUPER.

```
DROP TABLE IF EXISTS test_json;

SET enable_case_sensitive_super_attribute TO true;

CREATE TABLE test_json (all_data SUPER);

INSERT INTO test_json VALUES (JSON_PARSE('
{
   "data":{
      "pnr":{
         "type":"pnr",
         "pnrid":"123PQRS-2024-09-20",
         "bookingIdentifier":"123PQRS",
         "version":"5",
         "triggerType":"",
         "events":[
            {
               "eventType":"UPDATED",
               "type":"PART",
               "id":"123PQRS-2024-09-20-HO-1"
            },
            {
               "eventType":"CREATED",
               "type":"ABC",
               "id":"123PQRS-2024-09-20-OT-38"
            }
         ],
         "create":{
            "pnrCreateDate":"2024-09-20T16:56:00Z",
            "officeID":"OFFCID1234",
            "officeIDCategory":"Email"
         },
         "lastModification":{
            "dateTime":"2024-09-20T17:09:00Z"
         },
         "PARTDetails":[
            {
               "path":"1",
               "TrainPARTs":[
                  {
                     "PARTID":"123PQRS-2024-09-20-HO-1",
                     "departure":{
                        "departureStation":"XYZ",
```

```
                    "departureTimeLocal":"2024-10-03T06:30:00",
                    "departureTimeGMT":"2024-10-03T10:30:00Z"
                 },
                 "arrival":{
                    "arrivalStation":"ABC",
                    "arrivalTimeLocal":"2024-10-03T08:20:00",
                    "arrivalTimeGMT":"2024-10-03T15:20:00Z"
                 },
                 "marketing":{
                    "carrierCode":"XX",
                    "TrainNumber":"100"
                 },
                 "operating":{
                    "carrierCode":"YY",
                    "TrainNumber":"100-A"
                 },
                 "status":"ON",
                 "aircraft":{
                    "code":"222"
                 },
                 "class":"WC",
                 "first":"Y",
                 "seating":[

                 ]
              }
           ]
        }
     ],
     "commuterInformation":[
        {
           "commuterID":"2",
           "commuterPNR":"123PQRS-2024-09-20-RO-2",
           "commuterTypeCode":"DOM",
           "firstName":"JOHN",
           "lastName":"MILLER"
        }
     ],
     "contactDetail":[
        {
           "emailContacts":[
              {
                 "id":"123PQRS-2024-09-20-OT-4",
                 "contact":"JOHNMILLER@EXAMPLE.COM",
```

```
                    "purpose":[
                       "BUSINESS"
                    ],
                    "commuter":[
                       "123PQRS-2024-09-20-RO-2"
                    ],
                    "language":"EN"
                 },
                 {
                    "id":"123PQRS-2024-09-20-OT-5",
                    "contact":"HARVEYCORMIER@EXAMPLE.COM",
                    "purpose":[
                       "NOTIFICATION"
                    ],
                    "commuter":[
                       "123PQRS-2024-09-20-RO-2"
                    ],
                    "language":"EN"
                 }
              ]
           },
           {
              "phoneContacts":[
                 {
                    "id":"123PQRS-2024-09-20-OT-3",
                    "contact":"1234567890",
                    "purpose":[
                       "NOTIFICATION"
                    ],
                    "commuter":[
                       "123PQRS-2024-09-20-RO-2"
                    ],
                    "language":""
                 }
              ]
           },
           {
              "addressInfo":[
                 {
                    "id":"123PQRS-2024-09-20-OT-6",
                    "addressline":[
                       "112 PORT STREET"
                    ],
                    "provinceState":"CA",
```

```
                  "cityName":"SAN JOSE",
                  "postalCode":"12345",
                  "countryCode":"USA",
                  "purpose":[
                     "MAILING"
                  ],
                  "commuter":[
                     "123PQRS-2024-09-20-RO-2"
                  ]
               }
            ]
         }
      ],
      "PendingService":[
         {
            "id":"123PQRS-2024-09-20-OT-26",
            "code":"MONO",
            "status":"",
            "text":"Broken Seat at Coach-No XYZ123 Seat-No 567",
            "trainCode":"WC-1",
            "TrainsArray":[
               "123PQRS-2024-09-20-HO-1"
            ],
            "commuter":[
               "123PQRS-2024-09-20-RO-2"
            ]
         },
         {
            "id":"123PQRS-2024-09-20-OT-27",
            "code":"OTHS",
            "status":"",
            "text":"Broken Seat at Coach-No XYZ567 Seat-No 111",
            "trainCode":"WC-1",
            "TrainsArray":[
               "123PQRS-2024-09-20-HO-1"
            ],
            "commuter":[
               "123PQRS-2024-09-20-RO-2"
            ]
         },
         {
            "id":"123PQRS-2024-09-20-OT-28",
            "code":"OTHS",
            "status":"",
```

```
            "text":"Broken Seat at Coach-No XYZ890 Seat-No 123",
            "trainCode":"WC-1",
            "TrainsArray":[
               "123PQRS-2024-09-20-HO-1"
            ],
            "commuter":[
               "123PQRS-2024-09-20-RO-2"
            ]
         },
         {
            "id":"123PQRS-2024-09-20-OT-29",
            "code":"OTHS",
            "status":"",
            "text":"Broken Seat at Coach-No XYZ111 Seat-No 333",
            "trainCode":"WC-1",
            "TrainsArray":[
               "123PQRS-2024-09-20-HO-1"
            ],
            "commuter":[
               "123PQRS-2024-09-20-RO-2"
            ]
         }
      ],
      "parts": [
         {
            "partname": "prop",
            "manufacturer": "local parts co",
            "quality": 2,
            "price": 10.00
         },
         {
            "partname": "prop",
            "manufacturer": "big parts co",
            "quality": null,
            "price": 9.00
         },
         {
            "partname": "prop",
            "manufacturer": "small parts co",
            "quality": 1,
            "price": 12.00
         },
         {
            "partname": "rudder",
```

```
            "manufacturer": "local parts co",
            "quality": 1,
            "price": 2.50
         },
         {
            "partname": "rudder",
            "manufacturer": "big parts co",
            "quality": 2,
            "price": 3.75
         },
         {
            "partname": "rudder",
            "manufacturer": "small parts co",
            "quality": null,
            "price": 1.90
         },
         {
            "partname": "wing",
            "manufacturer": "local parts co",
            "quality": null,
            "price": 7.50
         },
         {
            "partname": "wing",
            "manufacturer": "big parts co",
            "quality": 1,
            "price": 15.20
         },
         {
            "partname": "wing",
            "manufacturer": "small parts co",
            "quality": null,
            "price": 11.80
         }
      ],
      "count_by_color": [
         {
            "quality": "high",
            "red": 15,
            "green": 20,
            "blue": 7
         },
         {
            "quality": "normal",
```

```
               "red": 35,
               "green": null,
               "blue": 40
            },
            {
               "quality": "low",
               "red": 10,
               "green": 23,
               "blue": null
            }
         ]
       }
   },
   "id":"abcdefgh-ijklmnop-qrstuvwxyz123",
   "mainIds":[
      {
         "ID":"pqrstuvwxyz-aabbcc123",
         "Source":"NYC"
      }
   ]
}
'));
```

Consultar dados semiestruturados aninhados

A instrução a seguir usa a notação de pontos do PartiQL para extrair o campo `pnrid`, que está aninhado em três níveis de profundidade dentro do objeto de nível superior `all_data`.

```
select all_data.data.pnr.pnrid::varchar from test_json;

 pnrid
--------------------
 123PQRS-2024-09-20
```

A declaração a seguir usa a notação de colchetes do PartiQL para especificar e extrair somente o primeiro elemento da matriz `events` aninhada dentro do objeto de nível superior.

```
SELECT
    all_data.data.pnr.events[0]
FROM test_json;

 events
```

```
---------------------------------
{
   "eventType":"UPDATED",
   "type":"PART",
   "id":"123PQRS-2024-09-20-HO-1"
}
```

A instrução a seguir extrai a propriedade `eventType` somente do elemento especificado da matriz `events`.

```
SELECT
    all_data.data.pnr.events[0].eventType
FROM test_json;

 eventtype
-----------
 "UPDATED"
```

As instruções a seguir

Uso de **`enable_case_sensitive_super_attribute`** com dados semiestruturados

Os exemplos a seguir mostram como a opção de configuração enable_case_sensitive_super_attribute afeta a consulta de dados semiestruturados. Para obter mais informações, consulte Acessar campos JSON com nomes ou atributos de campo em maiúsculas ou mistas.

Na declaração a seguir, redefinir a opção de configuração para o padrão false faz a consulta retornar NULL para nomes de atributos em letras mistas.

```
RESET enable_case_sensitive_super_attribute;

SELECT
    all_data.data.pnr.events[0].eventType
FROM test_json;

 eventtype
-----------
NULL
```

No exemplo a seguir, a consulta de amostra retorna o resultado desejado depois que você define `enable_case_sensitive_super_attribute` como verdadeiro. Não são necessárias aspas duplas nos nomes dos atributos.

```
SET enable_case_sensitive_super_attribute TO true;

SELECT
    all_data.data.pnr.events[0].eventType
FROM test_json;

 eventtype
-----------
 "UPDATED"
```

Filtrar dados semiestruturados

A instrução a seguir usa a sintaxe PartiQL na cláusula WHERE de uma instrução que conta eventos do tipo `UPDATED` para recuperar dados de um determinado atributo de dentro de uma matriz. Você pode usar essa sintaxe em qualquer parte da consulta em que normalmente referenciaria colunas.

```
SELECT COUNT(*)
FROM test_json
WHERE all_data.data.pnr.events[0].eventType = 'UPDATED';

 count
------
 1
```

O exemplo a seguir usa a sintaxe entre colchetes e pontos de PartiQL nas cláusulas GROUP BY e ORDER BY.

```
SELECT all_data.data.pnr.events[0].eventType::varchar,
       COUNT(*)
FROM test_json
WHERE all_data.data.pnr.events[0].eventType IS NOT NULL
GROUP BY all_data.data.pnr.events[0].eventType
ORDER BY all_data.data.pnr.events[0].eventType;

 eventtype | count
-----------+-------
 "UPDATED" | 1
```

Desaninhamento de dados semiestruturados

A declaração a seguir usa junções PartiQL para desaninhar a matriz `events`. Observe que essa junção funciona mesmo quando o número de índices da matriz não é estático.

Para obter exemplos de desaninhamento de dados semiestruturados usando UNNEST na cláusula FROM, consulte Exemplos de UNNEST.

```
SELECT
a.all_data.data.pnr.type::varchar type_info,
a.all_data.data.pnr.pnrid::varchar pnr_id ,
a.all_data.data.pnr.bookingIdentifier::varchar booking_id,
a.all_data.data.pnr.version::varchar version_info,
b.eventType::varchar event_type,
b.id::varchar event_id
FROM test_json a,
  a.all_data.data.pnr.events b;

 type_info | pnr_id              | booking_id | version_info | event_type | event_id
-----------+---------------------+------------+--------------+------------
+-------------------------
 pnr       | 123PQRS-2024-09-20  | 123PQRS    | 5            | UPDATED    |
 123PQRS-2024-09-20-HO-1
 pnr       | 123PQRS-2024-09-20  | 123PQRS    | 5            | CREATED    |
 123PQRS-2024-09-20-OT-38
```

Desaninhamento de matrizes aninhadas

A declaração a seguir usa junções PartiQL para desaninhar uma matriz que está aninhada dentro de outra matriz.

Para obter exemplos de desaninhamento de dados semiestruturados usando UNNEST na cláusula FROM, consulte Exemplos de UNNEST.

```
SELECT
a.all_data.data.pnr.type::varchar type_info,
a.all_data.data.pnr.pnrid::varchar pnr_id ,
a.all_data.data.pnr.bookingIdentifier::varchar booking_id,
a.all_data.data.pnr.version::varchar version_info,
d.id::varchar email_record_id,
d.contact::varchar email_contact,
```

```
e::varchar email_purpose,
f::varchar email_commuter
FROM test_json a,
  a.all_data.data.pnr.contactDetail c,
  c.emailContacts d,
  d.purpose e,
  d.commuter f;

 type_info | pnr_id              | booking_id | version_info | email_record_id
 | email_contact             | email_purpose | email_commuter
-----------+---------------------+------------+--------------+-------------------------
+--------------------------+---------------+-------------------------
 pnr       | 123PQRS-2024-09-20  | 123PQRS    | 5            | 123PQRS-2024-09-20-OT-4
 | JOHNMILLER@EXAMPLE.COM    | BUSINESS      | 123PQRS-2024-09-20-RO-2
 pnr       | 123PQRS-2024-09-20  | 123PQRS    | 5            | 123PQRS-2024-09-20-OT-5
 | HARVEYCORMIER@EXAMPLE.COM | NOTIFICATION  | 123PQRS-2024-09-20-RO-2
```

Usar dados semiestruturados em subconsultas

A instrução a seguir usa uma subconsulta na cláusula WHERE para retornar somente uma subseção dos resultados do exemplo anterior.

```
SELECT
a.all_data.data.pnr.type::varchar type_info,
a.all_data.data.pnr.pnrid::varchar pnr_id ,
a.all_data.data.pnr.bookingIdentifier::varchar booking_id,
a.all_data.data.pnr.version::varchar version_info,
d.id::varchar email_record_id,
d.contact::varchar email_contact
FROM test_json a,
a.all_data.data.pnr.contactDetail c,
c.emailContacts d
WHERE (SELECT COUNT(*) FROM d.purpose e WHERE e = 'BUSINESS') > 0;

 type_info | pnr_id              | booking_id | version_info | email_record_id
 | email_contact             | email_purpose | email_commuter
-----------+---------------------+------------+--------------+-------------------------
+--------------------------+---------------+-------------------------
 pnr       | 123PQRS-2024-09-20  | 123PQRS    | 5            | 123PQRS-2024-09-20-OT-4
 | JOHNMILLER@EXAMPLE.COM    | BUSINESS      | 123PQRS-2024-09-20-RO-2
```

Agregação de consultas usando dados semiestruturados

A instrução a seguir usa a função COUNT para agregar o número de elementos na matriz `PendingService`.

```
SELECT
a.all_data.data.pnr.type::varchar type_info,
a.all_data.data.pnr.pnrid::varchar pnr_id ,
a.all_data.data.pnr.bookingIdentifier::varchar booking_id,
a.all_data.data.pnr.version::varchar version_info,
COUNT(*) AS total_pending_service
FROM test_json a,
  a.all_data.data.pnr.PendingService c
GROUP BY 1,2,3,4;

 type_info | pnr_id             | booking_id | version_info | total_pending_service
-----------+--------------------+------------+--------------+----------------------
 pnr       | 123PQRS-2024-09-20 | 123PQRS    | 5            | 4
```

Usar dados semiestruturados em visões materializadas

A declaração a seguir usa a instrução do exemplo anterior para criar uma visão materializada. A visão materializada atualiza automaticamente o número de serviços pendentes quando a tabela base obtém novos dados.

```
CREATE MATERIALIZED VIEW mv_total_pending_service
AUTO REFRESH YES
AS
SELECT
a.all_data.data.pnr.type::varchar type_info,
a.all_data.data.pnr.pnrid::varchar pnr_id ,
a.all_data.data.pnr.bookingIdentifier::varchar booking_id,
a.all_data.data.pnr.version::varchar version_info,
COUNT(*) AS total_pending_service
FROM test_json a,
  a.all_data.data.pnr.PendingService c
GROUP BY 1,2,3,4;
```

Usar PIVOT e UNPIVOT com dados semiestruturados

A declaração a seguir usa PIVOT na coluna `partname` para retornar o preço médio de cada peça.

```
SELECT *
FROM
(
SELECT
c.partname::varchar, c.price
FROM test_json a,
  a.all_data.data.pnr.parts c)
PIVOT (AVG(price) for partname IN ('prop', 'rudder', 'wing'));


 prop       | rudder             |  wing
------------+--------------------+--------
 10.33      | 2.71               |  11.50
```

No exemplo anterior, os resultados são transformados em colunas. O exemplo a seguir mostra uma consulta GROUP BY que retorna os preços médios em linhas, em vez de em colunas.

```
SELECT partname, avg(price)
FROM (
SELECT
c.partname::varchar, c.price
FROM test_json a,
  a.all_data.data.pnr.parts c)
WHERE partname IN ('prop', 'rudder', 'wing')
GROUP BY partname;

 partname |  avg
----------+-------
 prop     | 10.33
 rudder   |  2.71
 wing     | 11.50
```

Veja a seguir um exemplo de PIVOT com `manufacturer` como uma coluna implícita.

```
SELECT *
FROM (
SELECT
c.quality, c.manufacturer::varchar
FROM test_json a,
  a.all_data.data.pnr.parts c) PIVOT (
count(*) FOR quality IN (1, 2, NULL)
);
```

```
 manufacturer      | 1  | 2  | null
-------------------+----+----+------
 local parts co    | 1  | 1  |  1
 big parts co      | 1  | 1  |  1
 small parts co    | 1  | 0  |  2
```

Veja a seguir um exemplo de UNPIVOT na coluna `quality`.

```
SELECT *
FROM
(
SELECT
c.quality as quality
FROM test_json a,
  a.all_data.data.pnr.parts c)
UNPIVOT (cnt FOR column_header IN (quality));

 column_header   | cnt
-----------------+----
 quality         | 2
 quality         | 1
 quality         | 1
 quality         | 2
 quality         | 1
```

Machine learning

O Amazon Redshift Machine Learning (Amazon Redshift ML) é um serviço robusto baseado em nuvem que ajuda analistas e cientistas de dados de todos os níveis de qualificação a usarem a tecnologia de Machine Learning. O Amazon Redshift ML usa um modelo para gerar resultados. É possível usar modelos das seguintes maneiras:

- Você fornece os dados desejados para treinar um modelo e metadados associados a entradas de dados ao Amazon Redshift. Em seguida, o Amazon Redshift ML cria modelos no Amazon SageMaker AI que capturam padrões nos dados de entrada. Ao usar seus próprios dados para o modelo, você pode usar o Amazon Redshift ML para identificar tendências nos dados, como previsão de rotatividade, valor de permanência do cliente ou previsão de receita. Você pode usar esses modelos para gerar previsões para novos dados de entrada sem incorrer em custos adicionais.
- É possível usar um dos modelos de base (FM) fornecidos pelo Amazon Bedrock, como Claude ou Amazon Titan. Usando o Amazon Bedrock, você pode combinar o poder dos grandes modelos de linguagem (LLMs) com data analytics no Amazon Redshift em apenas algumas etapas. Ao usar um grande modelo de linguagem (LLM) externo, você pode utilizar o Amazon Redshift para realizar processamento de linguagem natural (PLN) em seus dados. O PLN pode ser utilizado para aplicações como geração de texto, análise de sentimentos ou tradução. Para ter mais informações sobre como usar o Amazon Bedrock com o Amazon Redshift, consulte Integração do Amazon Redshift ML com o Amazon Bedrock.

> **Note**
>
> Optar por não usar seus dados para melhorar o serviço
> Se você estiver usando modelos do Amazon Bedrock, é recomendável ler as políticas da AWS sobre como o serviço Amazon Bedrock lida com seus dados. Você deve determinar se precisa usar uma política de cancelamento para evitar que o serviço use seus dados para melhorias no modelo ou no serviço, caso o Amazon Bedrock implemente essa funcionalidade no futuro. Para garantir que o serviço não use seus dados para essas finalidades, use a política geral de exclusão da AWS.
> Para saber mais, consulte:
>
> - Políticas de recusa de serviços de IA

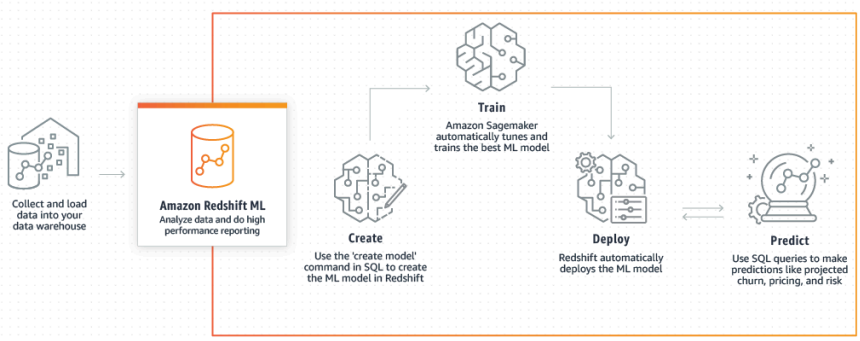
- Perguntas frequentes sobre o Amazon Bedrock

> **Note**
>
> Os LLMs podem gerar informações imprecisas ou incompletas. Recomendamos verificar as informações produzidas pelos LLMs para garantir que sejam precisas e completas.

Como o Amazon Redshift ML utiliza o Amazon SageMaker AI

O Amazon Redshift utiliza o Amazon SageMaker AI Autopilot para obter automaticamente o melhor modelo e disponibilizar a função de previsão no Amazon Redshift.

O diagrama a seguir ilustra como o Amazon Redshift ML funciona.



O fluxo de trabalho geral é o seguinte:

1. O Amazon Redshift exporta os dados de treinamento para o Amazon S3.
2. O Amazon SageMaker AI Autopilot pré-processa os dados de treinamento. O pré-processamento executa funções importantes, como a imputação de valores ausentes. Ele reconhece que certas colunas são categóricas (como o código postal), formata-as corretamente para treinamento e executa inúmeras outras tarefas. Escolher os melhores pré-processadores para aplicar no conjunto de dados de treinamento é um problema por si só, e o Amazon SageMaker AI Autopilot automatiza a respectiva solução.
3. O Amazon SageMaker AI Autopilot encontra o algoritmo e os hiperparâmetros do algoritmo que fornecem o modelo com as previsões mais precisas.
4. O Amazon Redshift registra a função de previsão como uma função SQL no cluster do Amazon Redshift.

5. Quando você executa instruções CREATE MODEL, o Amazon Redshift usa o Amazon SageMaker AI para treinamento. Portanto, há um custo associado para treinar seu modelo. Este é um item de linha separado para o Amazon SageMaker AI em seu faturamento da AWS. Você também paga pelo armazenamento usado no Amazon S3 para armazenar seus dados de treinamento. Não será cobrada a inferência que usa modelos criados com CREATE MODEL que podem ser compilados e executados no cluster do Amazon Redshift. Não há cobranças adicionais do Amazon Redshift para usar o Amazon Redshift ML.

Tópicos

- Visão geral do Machine Learning
- Machine Learning para iniciantes e especialistas
- Custos para usar o Amazon Redshift ML
- Conceitos básicos do Amazon Redshift ML
- Tutoriais para o Amazon Redshift ML
- Integração do Amazon Redshift ML com o Amazon Bedrock

Visão geral do Machine Learning

Com o Amazon Redshift, você pode aproveitar os recursos de machine learning para obter insights valiosos dos seus dados. Esta visão geral sobre machine learning (ML) mostra como explorar, visualizar e preparar seus dados para treinamento e implantação de um modelo de ML. As seções a seguir orientarão você pelo processo de utilização do Amazon Redshift ML para aproveitar o potencial de seus dados por meio de machine learning.

Usando o Amazon Redshift ML, é possível treinar modelos de machine learning usando instruções SQL e chamá-los em consultas SQL para previsão.

Para ajudá-lo a saber como usar o Amazon Redshift ML, assista ao seguinte vídeo: Amazon Redshift ML.

Para ter informações sobre os pré-requisitos para configurar clusters ou grupos de trabalho sem servidor, permissões e propriedade do Redshift para usar o Amazon Redshift ML, leia as seções a seguir. Essas seções também descrevem como o treinamento simples e as previsões funcionam no Amazon Redshift ML.

Como o machine learning pode resolver seu problema

Um modelo de Machine Learning gera previsões encontrando padrões em seus dados de treinamento e aplicando esses padrões a novos dados. No Machine Learning, você treina esses modelos aprendendo os padrões que melhor explicam seus dados. Em seguida, você usa os modelos para fazer previsões (também chamadas inference) sobre novos dados. Machine Learning normalmente é um processo iterativo no qual você pode continuar a melhorar a precisão das previsões alterando os parâmetros e melhorando os dados de treinamento. Se os dados forem alterados, o novo treinamento de novos modelos com o novo conjunto de dados acontece.

Para atender a vários objetivos de negócios, existem diferentes abordagens fundamentais de Machine Learning.

Aprendizado supervisionado no Amazon Redshift ML

O Amazon Redshift oferece suporte à aprendizagem supervisionada, que é a abordagem mais comum para análises empresariais avançadas. O aprendizado supervisionado é a abordagem preferida de Machine Learning quando você tem um conjunto estabelecido de dados e uma compreensão de como os dados de entrada específicos prevêem vários resultados de negócios. Esses resultados às vezes são chamados de rótulos. Em particular, seu conjunto de dados é uma tabela com atributos que compreendem feições (entradas) e destinos (saídas). Por exemplo, suponha que você tenha uma tabela que forneça a idade e o código postal para clientes passados e presentes. Suponha que você também tenha um campo “ativo” que seja true para clientes presentes e false para clientes que suspenderam sua associação. O objetivo do machine learning supervisionado é identificar os padrões de idade e código postal que levam à rotatividade de clientes, como representado por clientes cujos destinos são “False”. Você pode usar esse modelo para prever clientes com probabilidade de rotatividade, como suspender sua associação, e potencialmente oferecer incentivos de retenção.

O Amazon Redshift oferece suporte ao aprendizado supervisionado que inclui regressão, classificação binária e classificação multiclasse. Regressão refere-se ao problema da previsão de valores contínuos, como o gasto total dos clientes. Classificação binária refere-se ao problema de prever um de dois resultados, como prever se um cliente rotula ou não. Classificação multiclasse refere-se ao problema de prever um de muitos resultados, como prever o item que um cliente pode estar interessado. Analistas de dados e cientistas de dados podem usá-lo para executar o aprendizado supervisionado para lidar com problemas que vão desde previsão, personalização ou previsão de rotatividade de clientes. Também é possível usar o aprendizado supervisionado em

problemas como previsão de quais vendas serão fechadas, previsão de receita, detecção de fraude e previsão de valor de vida do cliente.

Aprendizado não supervisionado no Amazon Redshift ML

O aprendizado não supervisionado usa algoritmos de machine learning para analisar e agrupar dados de treinamento não rotulados. Os algoritmos detectam padrões ou agrupamentos ocultos. O objetivo é modelar a estrutura ou distribuição subjacente nos dados para saber mais sobre os dados.

O Amazon Redshift é compatível com o algoritmo de cluster K-Means para resolver um problema de aprendizado não supervisionado. Esse algoritmo resolve problemas de clusterização em que você deseja detectar agrupamentos nos dados. O algoritmo K-Means tenta encontrar agrupamentos discretos dentro dos dados. Os dados não classificados são agrupados e particionados de acordo com suas semelhanças e diferenças. Por meio do agrupamento, o algoritmo K-Means determina iterativamente os melhores centroides e atribui cada membro ao centroide mais próximo. Os membros mais próximos do mesmo centroide pertencem ao mesmo grupo. Os membros de um grupo são o mais semelhantes possível a outros membros do mesmo grupo e o mais diferentes possível de membros de outros grupos. Por exemplo, o algoritmo de clusterização K-Means pode ser usado para classificar cidades afetadas por uma pandemia ou classificar cidades de acordo com a popularidade dos produtos de consumo.

Ao usar o algoritmo K-Means, especifique uma entrada k que especifique o número de clusters a serem encontrados nos dados. A saída desse algoritmo é um conjunto de centroides k. Cada ponto de dados pertence a um dos clusters k mais próximos a ele. Cada cluster é descrito por seu centroide. Pode-se pensar no centroide como a média multidimensional do cluster. O algoritmo K-Means compara as distâncias para ver o quanto os clusters diferem uns dos outros. Uma distância maior geralmente indica uma diferença maior entre os clusters.

O pré-processamento dos dados é importante para o K-Means, pois garante que os recursos do modelo permaneçam na mesma escala e produzam resultados confiáveis. O Amazon Redshift é compatível com alguns pré-processadores K-Means para a instrução CREATE MODEL, como StandardScaler, MinMax e NumericPassThrough. Caso não queira aplicar nenhum pré-processamento para K-means, escolha NumericPassthrough explicitamente como um transformador. Para obter mais informações sobre como configurar parâmetros K-Means, consulte CREATE MODEL com parâmetros K-MEANS.

Para ajudá-lo a saber como realizar treinamento não supervisionado com o cluster K-Means, você pode assistir ao seguinte vídeo: Unsupervised training with K-Means clustering (Treinamento não supervisionado com clusters K-Means).

Termos e conceitos do Amazon Redshift ML

Os seguintes termos são usados para descrever alguns conceitos do Amazon Redshift ML:

- Machine Learning no Amazon Redshift treina um modelo com um comando SQL. O Amazon Redshift ML e o Amazon SageMaker AI gerenciam todas as conversões de dados, permissões, uso de recursos e descoberta do modelo adequado.
- Treinamento é a fase em que o Amazon Redshift cria um modelo de Machine Learning executando um subconjunto especificado de dados no modelo. O Amazon Redshift inicia automaticamente um trabalho de treinamento no Amazon SageMaker AI e gera um modelo.
- A previsão (também chamada de inferência) é o uso do modelo em consultas SQL do Amazon Redshift para prever resultados. No momento da inferência, o Amazon Redshift usa uma função de previsão baseada em modelo como parte de uma consulta maior para produzir previsões. As previsões são calculadas no local, no cluster do Redshift, proporcionando assim alta taxa de transferência, baixa latência e nenhum custo adicional.
- Com o traga seu próprio modelo (BYOM), você pode usar um modelo treinado fora do Amazon Redshift com o Amazon SageMaker AI para inferência no banco de dados localmente no Amazon Redshift. O Amazon Redshift ML oferece suporte ao uso de BYOM para inferência local.
- A inferência local é usada quando os modelos são pré-treinados no Amazon SageMaker AI, compilados pelo Amazon SageMaker AI Neo e localizados no Amazon Redshift ML. Para importar modelos compatíveis com inferência local para o Amazon Redshift, use o comando CREATE MODEL. O Amazon Redshift importa os modelos pré-treinados do SageMaker AI chamando o Amazon SageMaker AI Neo. Você compila o modelo lá e importa o modelo compilado para o Amazon Redshift. Use inferência local para uma velocidade mais rápida e custos mais baixos.
- A inferência remota é usada quando o Amazon Redshift invoca um endpoint de modelo implantado no SageMaker AI. Ela oferece a flexibilidade de invocar todos os tipos de modelo personalizado e de modelo de aprendizado profundo, como modelos TensorFlow que você criou e implantou no Amazon SageMaker AI.

Os seguintes fatores também são importantes:

- O Amazon SageMaker é um serviço de machine learning totalmente gerenciado. Com o Amazon SageMaker AI, cientistas de dados e desenvolvedores podem facilmente criar, treinar e implantar modelos diretamente em um ambiente hospedado pronto para produção. Para obter informações sobre o Amazon SageMaker, consulte What is Amazon SageMaker AI no Guia do desenvolvedor do Amazon SageMaker AI.

- O Amazon SageMaker AI Autopilot é um conjunto de recursos que treina e ajusta automaticamente os melhores modelos de machine learning para classificação ou regressão, com base em seus dados. Você mantém total controle e visibilidade. O Amazon SageMaker AI Autopilot aceita dados de entrada em formato tabular. O Amazon SageMaker AI Autopilot oferece limpeza e pré-processamento automáticos de dados, seleção automática de algoritmos para regressão linear, classificação binária e classificação multiclasse. Ele também comporta otimização automática de hiperparâmetros (HPO), treinamento distribuído, instância automática e seleção de tamanho de cluster. Para obter informações sobre o Amazon SageMaker AI Autopilot, consulte Automate model development with Amazon SageMaker AI Autopilot no Guia do desenvolvedor do Amazon SageMaker AI.
- O Amazon Bedrock é um serviço totalmente gerenciado que oferece uma seleção de modelos de base (FMs) de alto desempenho de empresas líderes de IA, como AI21 Labs, Anthropic, Cohere, Meta, Mistral AI, Stability AI e Amazon, por meio de uma única API, além de um amplo conjunto de recursos necessários para criar aplicações de IA generativa.

Machine Learning para iniciantes e especialistas

Com o Amazon Redshift, você pode aproveitar os recursos de machine learning (ML) para extrair insights dos seus dados, seja você iniciante ou especialista em ML. O Machine Learning é um recurso do Amazon Redshift que você usa para criar, treinar e implantar modelos de machine learning (ML) usando comandos SQL, sem a necessidade de conhecimentos aprofundados de ML ou de engenharia de dados complexa.

As seções a seguir fornecem orientações sobre o processo de utilização de machine learning, possibilitando que você aproveite o potencial de seus dados com o Amazon Redshift.

Com o Amazon Redshift ML, é possível treinar modelos com um único comando SQL CREATE MODEL. O comando CREATE MODEL cria um modelo que o Amazon Redshift usa para gerar previsões baseadas em modelos com construções SQL familiares.

O Amazon Redshift ML é especialmente útil quando você não tem experiência em Machine Learning, ferramentas, linguagens, algoritmos e APIs. Com o Amazon Redshift ML, você não precisa realizar o trabalho pesado que é necessário para integração com um serviço externo de Machine Learning. O Amazon Redshift poupa o tempo de formatar e mover dados, gerenciar controles de permissão ou criar integrações, fluxos de trabalho e scripts personalizados. Você pode usar facilmente algoritmos populares de Machine Learning e simplificar as necessidades de treinamento que exigem iteração frequente, desde o treinamento até a previsão. O Amazon Redshift descobre automaticamente o

melhor algoritmo e ajusta o melhor modelo para o seu problema. Você pode fazer previsões dentro do cluster do Amazon Redshift sem precisar transferir os dados para fora do Amazon Redshift nem interagir e pagar por outro serviço.

O Amazon Redshift ML oferece suporte a analistas de dados e cientistas de dados no uso de machine learning. Também permite que especialistas em machine learning usem seus conhecimentos para orientar a instrução CREATE MODEL a usar apenas os aspectos especificados. Com isso, você pode acelerar o tempo necessário para que CREATE MODEL encontre o melhor candidato, melhore a precisão do modelo ou ambas as coisas.

A instrução CREATE MODEL oferece flexibilidade em como você pode especificar os parâmetros para o trabalho de treinamento. Essa flexibilidade permite que tanto iniciantes como especialistas de machine learning escolham os pré-processadores, algoritmos, tipos de problemas ou hiperparâmetros de sua preferência. Por exemplo, um usuário interessado na rotatividade de clientes pode especificar na instrução CREATE MODEL que o tipo de problema é uma classificação binária, o que funciona bem para rotatividade de clientes. Em seguida, a instrução CREATE MODEL restringe sua busca pelo melhor modelo em modelos de classificação binária. Mesmo com a escolha do usuário do tipo de problema, ainda há muitas opções com as quais a instrução CREATE MODEL pode trabalhar. Por exemplo, o CREATE MODEL descobre e aplica as melhores transformações de pré-processamento e descobre as melhores configurações de hiperparâmetro.

O Amazon Redshift ML facilita o treinamento ao encontrar automaticamente o melhor modelo usando o Amazon SageMaker AI Autopilot. Nos bastidores, o Amazon SageMaker AI Autopilot treina e ajusta automaticamente o melhor modelo de machine learning com base nos dados fornecidos. O Amazon SageMaker AI Neo compila o modelo de treinamento e o disponibiliza para previsão no cluster do Redshift. Quando você executa uma consulta de inferência de machine learning usando um modelo treinado, a consulta pode usar todos os recursos de processamento paralelo em massa do Amazon Redshift. Ao mesmo tempo, a consulta pode usar a previsão baseada em machine learning.

- Como iniciante em machine learning com conhecimento geral de diferentes aspectos do machine learning, como pré-processadores, algoritmos e hiperparâmetros, use a instrução CREATE MODEL apenas para os aspectos especificados. Em seguida, você pode encurtar o tempo que CREATE MODEL precisa para encontrar o melhor candidato ou melhorar a precisão do modelo. Além disso, você pode aumentar o valor comercial das previsões introduzindo conhecimento de domínio adicional, como o tipo de problema ou o objetivo. Por exemplo, em um cenário de rotatividade de cliente, se o resultado “cliente não está ativo” for raro, o objetivo de F1 geralmente é preferido ao objetivo de Precisão. Como os modelos de alta precisão podem prever “o cliente está ativo” o tempo todo, isso resulta em alta precisão, mas pouco valor empresarial. Para obter

informações sobre o objetivo F1, consulte AutoMLJobObjective na Referência de API do Amazon SageMaker AI.

Para obter mais informações sobre as opções básicas para a instrução CREATE MODEL, consulte CREATE MODEL simples.

- Como um Praticante avançado de Machine Learning, você pode especificar o tipo de problema e os pré-processadores para certos recursos (mas não todos). Em seguida, a instrução CREATE MODEL segue suas sugestões sobre os aspectos especificados. Ao mesmo tempo, a instrução CREATE MODEL detecta automaticamente os melhores pré-processadores para os recursos restantes e os melhores hiperparâmetros. Para obter mais informações sobre como você pode restringir um ou mais aspectos do pipeline de treinamento, consulte CREATE MODEL com orientação do usuário.
- Como um especialista em Machine Learning, você pode assumir o controle total do treinamento e do ajuste de hiperparâmetros. Em seguida, a instrução CREATE MODEL não tenta detectar os pré-processadores, algoritmos e hiperparâmetros ideais porque você faz todas as escolhas. Para obter mais informações sobre como usar CREATE MODEL com AUTO OFF, consulte Modelos CREATE XGBoost com AUTO OFF.
- Os engenheiros de dados podem trazer um modelo XGBoost pré-treinado para o Amazon SageMaker AI e importá-lo para o Amazon Redshift para inferência local. Com o traga seu próprio modelo (BYOM), você pode usar um modelo treinado fora do Amazon Redshift com o Amazon SageMaker AI para inferência no banco de dados localmente no Amazon Redshift. O Amazon Redshift ML oferece suporte ao uso de BYOM para inferência local ou remota.

  Para obter mais informações sobre como usar a instrução CREATE MODEL para inferência local ou remota, consulte Traga seu próprio modelo (BYOM): inferência local.

Como usuário do Amazon Redshift ML, você pode escolher qualquer uma das opções a seguir para treinar e implantar seu modelo:

- Tipos de problema, consulte CREATE MODEL com orientação do usuário.
- Objetivos, consulte CREATE MODEL com orientação do usuário ou Modelos CREATE XGBoost com AUTO OFF.
- Tipos de modelo, consulte Modelos CREATE XGBoost com AUTO OFF.
- Pré-processadores, consulte CREATE MODEL com orientação do usuário.
- Hiperparâmetros, consulte Modelos CREATE XGBoost com AUTO OFF.
- Traga seu próprio modelo (BYOM), consulte Traga seu próprio modelo (BYOM): inferência local.

Custos para usar o Amazon Redshift ML

Com o Amazon Redshift, é possível aproveitar os recursos de machine learning para extrair insights de seus dados, sem a necessidade de uma ampla experiência em engenharia de dados ou machine learning. As seções a seguir descrevem os custos associados ao uso do Amazon Redshift ML e ajudam você a planejar e otimizar suas despesas, bem como a aproveitar essa avançada integração de machine learning.

Custos para usar o Amazon Redshift ML com o SageMaker AI

O Amazon Redshift ML para SageMaker AI usa seus recursos de cluster existentes para previsão, de modo que você possa evitar cobranças adicionais do Amazon Redshift. Não há cobrança adicional do Amazon Redshift para criar ou usar um modelo. A previsão acontece no local, no cluster do Redshift, então não é necessário pagar mais, a menos que você precise redimensionar o cluster. O Amazon Redshift ML usa o Amazon SageMaker AI para treinar seu modelo, o que tem um custo adicional associado.

Não há cobrança adicional para funções de previsão executadas no seu cluster do Amazon Redshift. A instrução CREATE MODEL usa o Amazon SageMaker AI e incorre em um custo adicional. O custo aumenta com o número de células em seus dados de treinamento. O número de células é o produto do número de registros (nos horários de consulta de treinamento ou tabela) vezes o número de colunas. Por exemplo, quando uma consulta SELECT da instrução CREATE MODEL cria 10.000 registros e 5 colunas, o número de células que ela cria é 50.000.

Em alguns casos, os dados de treinamento produzidos pela consulta SELECT da instrução CREATE MODEL excedem o limite MAX_CELLS fornecido (ou o padrão 1 milhão, se você não forneceu um limite). Nesses casos, a instrução CREATE MODEL escolhe de modo aleatório aproximadamente MAX_CELLS (ou seja, os registros de “número de colunas” do conjunto de dados de treinamento). A instrução CREATE MODEL então executa o treinamento usando essas tuplas escolhidas aleatoriamente. A amostragem aleatória garante que o conjunto de dados de treinamento reduzido não tenha qualquer viés. Assim, definindo o MAX_CELLS, você pode controlar seus custos de treinamento.

Ao usar a instrução CREATE MODEL, você pode usar as opções MAX_CELLS e MAX_RUNTIVE para controlar os custos, o tempo e a precisão do modelo potencial.

MAX_RUNTIME especifica o tempo máximo que o treinamento pode levar no SageMaker AI quando a opção AUTO ON ou OFF é usada. Os trabalhos de treinamento geralmente concluem antes do MAX_RUNTIVE, dependendo do tamanho do conjunto de dados. Depois que um modelo é treinado,

o Amazon Redshift faz um trabalho adicional em segundo plano para compilar e instalar seus modelos em seu cluster. Assim, CREATE MODEL pode demorar mais do que MAX_RUNTIVE. No entanto, MAX_RUNTIME limita a quantidade de computação e tempo usados no SageMaker AI para treinar seu modelo. Você pode verificar o status do seu modelo a qualquer momento usando SHOW MODEL.

Quando você executa CREATE MODEL com AUTO ON, o Amazon Redshift ML usa o SageMaker AI Autopilot para explorar de forma automática e inteligente diferentes modelos (ou candidatos) para encontrar o melhor. MAX_RUNTURE limita a quantidade de tempo e a computação gasta. Se MAX_RUNTIVE estiver definido muito baixo, talvez não haja tempo suficiente para explorar até mesmo um candidato. Se vir o erro “Candidato de piloto automático não tem modelos”, execute novamente o CREATE MODEL com um valor MAX_RUNTIVE maior. Para obter mais informações sobre esse parâmetro, consulte MaxAutoMLJobRuntimeInSeconds na Referência de API do Amazon SageMaker AI.

Quando você executa CREATE MODEL com AUTO OFF, MAX_RUNTIME corresponde a um limite de tempo de execução do trabalho de treinamento no SageMaker AI. Os trabalhos de treinamento geralmente são concluídos mais cedo, dependendo do tamanho do conjunto de dados e de outros parâmetros usados, como num_rounds em MODEL_TYPE XGBOOST.

Você também pode controlar custos ou reduzir o tempo de treinamento especificando um valor MAX_CELLS menor ao executar CREATE MODEL. A célula é uma entrada no banco de dados. Cada linha corresponde a tantas células quanto existem colunas, que podem ser de largura fixa ou variável. MAX_CELLS limita o número de células e, portanto, o número de exemplos de treinamento usados para treinar seu modelo. Por padrão, MAX_CELLS é definido como 1 milhão de células. Reduzir MAX_CELLS diminui o número de linhas do resultado da consulta SELECT no CREATE MODEL que o Amazon Redshift exporta e envia ao SageMaker para treinar um modelo. Reduzindo MAX_CELLS, assim, reduz o tamanho do conjunto de dados usado para treinar modelos com AUTO ON e AUTO OFF. Essa abordagem ajuda a reduzir os custos e o tempo para treinar modelos. Para ver informações sobre treinamento e tempos de cobrança de um trabalho de treinamento específico, selecione Trabalhos de treinamento no Amazon SageMaker AI.

Aumentar MAX_RUNTIME e MAX_CELLS geralmente melhora a qualidade do modelo, permitindo que o SageMaker AI explore mais candidatos. Dessa forma, o SageMaker AI pode ter mais tempo para treinar cada candidato e usar mais dados para treinar modelos melhores. Se você quiser uma iteração ou exploração mais rápida do seu conjunto de dados, use MAX_RUNTIVE e MAX_CELLS inferiores. Se você quiser uma precisão aprimorada dos modelos, use MAX_RUNTIVE e MAX_CELLS superiores.

Para obter mais informações sobre os custos associados a vários números de celular e detalhes sobre o teste gratuito, consulte Preços do Amazon Redshift.

Custos para usar o Amazon Redshift ML com o Amazon Bedrock

O uso do Amazon Redshift ML com o Amazon Bedrock gera custos adicionais. Para obter mais informações, consulte Preço do Amazon Bedrock.

Conceitos básicos do Amazon Redshift ML

O Amazon Redshift ML facilita a criação, o treinamento e a implantação de modelos de Machine Learning usando comandos SQL familiares. Com o Amazon Redshift ML, você pode usar os dados no cluster do Redshift para treinar modelos com o Amazon SageMaker AI. Depois, os modelos são localizados e as previsões podem ser feitas em um banco de dados do Amazon Redshift. No momento, o Amazon Redshift ML é compatível com os seguintes algoritmos de machine learning: XGBoost (AUTO ON e OFF) e perceptron multicamada (AUTO ON), K-Means (AUTO OFF) e Linear Learner.

Tópicos

- Definir cluster e configuração para administração do Amazon Redshift ML
- Usar a explicabilidade do modelo com o Amazon Redshift ML
- Métricas de probabilidade do Amazon Redshift ML

Definir cluster e configuração para administração do Amazon Redshift ML

Antes de trabalhar com o Amazon Redshift ML, conclua a configuração do cluster e configure as permissões para usar o Amazon Redshift ML.

Configuração de cluster para usar o Amazon Redshift ML

Antes de trabalhar com o Amazon Redshift ML, preencha os pré-requisitos a seguir.

Como administrador do Amazon Redshift, faça a configuração única a seguir para usar clusters provisionados pelo Amazon Redshift. Para usar o Amazon Redshift ML com o Amazon Redshift sem servidor, consulte Getting started with Amazon Redshift Serverless.

Para executar a configuração única do cluster para o Amazon Redshift ML

1. Crie um cluster do Amazon Redshift usando o Console de gerenciamento da AWS ou a AWS Command Line Interface (AWS CLI). Anexe a política do AWS Identity and Access Management (IAM) ao criar o cluster. Para ter mais informações sobre as permissões necessárias para usar o Amazon Redshift ML com o Amazon SageMaker, consulte Permissões necessárias para usar o machine learning (ML) do Amazon Redshift.
2. Crie o perfil do IAM necessário para usar o Amazon Redshift ML realizando um dos seguintes procedimentos:

   - Para usar o SageMaker AI com o Amazon Redshift ML, crie um perfil do IAM com as políticas `AmazonS3FullAccess` e `AmazonSageMakerFullAccess`. Se você também planeja criar modelos do Forecast, anexe igualmente a política `AmazonForecastFullAccess` ao seu perfil.
   - Para usar o Amazon Bedrock com o Amazon Redshift ML, crie um perfil do IAM com as políticas `AmazonS3FullAccess` e `AmazonBedrockFullAccess`.
   - Recomendamos criar uma função do IAM por meio do console do Amazon Redshift que tenha a política `AmazonRedshiftAllCommandsFullAccess` com permissões para executar comandos SQL, como CREATE MODEL. O Amazon Redshift usa um mecanismo contínuo baseado em API para criar funções do IAM de maneira programática em sua Conta da AWS em seu nome. O Amazon Redshift anexa automaticamente políticas gerenciadas pela AWS existentes para a função do IAM. Essa metodologia significa que você pode permanecer no console do Amazon Redshift e não precisa alternar para o console do IAM para criar a função. Para obter mais informações, consulte Criar uma função do IAM como padrão para o Amazon Redshift.

     Quando uma função do IAM for criada como padrão para o cluster, inclua `redshift` como parte do nome do recurso ou use uma etiqueta específica do RedShift para etiquetar esses recursos.

     Para usar os modelos de base do Amazon Bedrock, adicione a seguinte seção:

```
// Required section if you use Bedrock models.
{
   "Effect": "Allow",
   "Action": "bedrock:InvokeModel",
   "Resource": [
       "arn:aws:bedrock:<region>::foundation-model/*"
   ]
}
```

- Se você deseja criar uma função do IAM com uma política mais restritiva, use a política a seguir. Você também pode modificar essa política para atender às suas necessidades.

  O bucket do Amazon S3 `redshift-downloads/redshift-ml/` é o local onde os dados de amostra usados para outras etapas e exemplos são armazenados. Você pode removê-lo se não precisar carregar dados do Amazon S3. Ou substitua-o por outros buckets do Amazon S3 que você usa para carregar dados no Amazon Redshift.

  Os valores *`your-account-id`*, *`your-role`* e `amzn-s3-demo-bucket` são os que você especifica como parte do comando CREATE MODEL.

  (Opcional) Use as chaves AWS KMS da política de exemplo se você especificar uma chave AWS KMS ao usar o Amazon Redshift ML. O valor *`your-kms-key`* é a chave que você usa como parte do comando CREATE MODEL.

  JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:PutMetricData",
                "ecr:BatchCheckLayerAvailability",
                "ecr:BatchGetImage",
                "ecr:GetAuthorizationToken",
                "ecr:GetDownloadUrlForLayer",
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:DescribeLogStreams",
                "logs:PutLogEvents",
                "sagemaker:*Job*",
                "sagemaker:AddTags",
                "sagemaker:CreateModel",
                "sagemaker:CreateEndpoint",
                "sagemaker:CreateEndpointConfig",
                "sagemaker:DeleteEndpoint",
                "sagemaker:DeleteEndpointConfig",
                "sagemaker:DeleteModel"
```

```
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole",
                "s3:AbortMultipartUpload",
                "s3:GetObject",
                "s3:DeleteObject",
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:iam::111122223333:role/<your-role>",
                "arn:aws:s3:::amzn-s3-demo-bucket/*",
                "arn:aws:s3:::redshift-downloads/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::redshift-downloads"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:CreateGrant",
                "kms:Decrypt",
                "kms:DescribeKey",
                "kms:Encrypt",
                "kms:GenerateDataKey*"
             ],
             "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/<your-kms-key>"
             ]
        }
    ]
```

```
}
```

3. Para permitir que o Amazon Redshift e o SageMaker AI assumam a função de interagir com outros serviços, adicione a política de confiança a seguir ao perfil do IAM.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "redshift.amazonaws.com",
          "sagemaker.amazonaws.com",
          "forecast.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

4. (Opcional) Criar um bucket do Amazon S3 e uma chave AWS KMS. Eles são usados pelo Amazon Redshift para armazenar os dados de treinamento enviados ao Amazon SageMaker AI e receber o modelo treinado do Amazon SageMaker AI.
5. (Opcional) Crie diferentes combinações de funções do IAM e buckets do Amazon S3 para controlar o acesso a diferentes grupos de usuários.
6. Quando você ativa o roteamento de VPC aprimorado, o tráfego entre o Redshift ML e seu bucket do S3 passa pela sua VPC privada. Para obter mais informações sobre o roteamento da VPC, consulte Roteamento aprimorado da VPC no Amazon Redshift.

   Para obter mais informações sobre as permissões necessárias para especificar uma VPC privada para seu trabalho de ajuste de hiperparâmetros, consulte Permissions required to use Amazon Redshift ML with Amazon SageMaker AI.

> **Note**
>
> As chamadas de inferência feitas para modelos remotos do SageMaker IA não passam pela VPC.

Para obter informações sobre como usar a instrução CREATE MODEL para começar a criar modelos para diferentes casos de uso, consulte CREATE MODEL.

Gerenciar permissões e propriedade

Assim como em outros objetos de banco de dados, como tabelas ou funções, o Amazon Redshift vincula a criação e o uso de modelos ML para acessar mecanismos de controle. Há permissões separadas para a criação de um modelo que executa as funções de previsão.

Os exemplos a seguir usam dois grupos de usuários: `retention_analyst_grp` (criador do modelo) e `marketing_analyst_grp` (usuário do modelo), para ilustrar como o Amazon Redshift gerencia o controle de acesso. O analista de retenção cria modelos de machine learning que outros conjuntos de usuários podem usar com as permissões obtidas.

Um superusuário pode conceder permissão USER ou GROUP para criar modelos de Machine Learning usando a instrução a seguir.

```
GRANT CREATE MODEL TO GROUP retention_analyst_grp;
```

Usuários ou grupos com essa permissão podem criar um modelo em qualquer esquema no cluster se um usuário tiver a permissão usual de CREATE no SCHEMA. O modelo de Machine Learning faz parte da hierarquia de esquemas de forma semelhante a tabelas, exibições, procedimentos e funções definidas pelo usuário.

Supondo que já exista um esquema `demo_ml`, conceda aos dois grupos de usuários a permissão no esquema da seguinte forma.

```
GRANT CREATE, USAGE ON SCHEMA demo_ml TO GROUP retention_analyst_grp;
```

```
GRANT USAGE ON SCHEMA demo_ml TO GROUP marketing_analyst_grp;
```

Para permitir que outros usuários usem a função de inferência de machine learning, conceda a permissão EXECUTE. O exemplo a seguir usa a permissão EXECUTE para conceder ao GRUPO marketing_analyst_grp a permissão de usar o modelo.

```
GRANT EXECUTE ON MODEL demo_ml.customer_churn_auto_model TO GROUP
 marketing_analyst_grp;
```

Use a instrução REVOKE com CREATE MODEL e EXECUTE para revogar essas permissões de usuários ou grupos. Para obter mais informações sobre comandos de controle de permissão, consulte GRANT e REVOKE.

Usar a explicabilidade do modelo com o Amazon Redshift ML

Com a explicabilidade do modelo do Amazon Redshift ML, você usa valores relevantes de recursos para ajudar a entender como cada atributo dos dados de treinamento contribui para o resultado previsto.

A explicabilidade do modelo ajuda a melhorar os modelos de machine learning (ML) ao explicar as previsões feitas pelos modelos. A explicabilidade do modelo ajuda a explicar como esses modelos fazem previsões usando uma abordagem de atribuição de recursos.

O Amazon Redshift ML incorpora a explicabilidade do modelo para fornecer funcionalidade de explicação do modelo aos usuários do Amazon Redshift ML. Para obter mais informações sobre a explicabilidade do modelo, consulte O que é imparcialidade e explicabilidade do modelo para predições de machine learning? no Guia do desenvolvedor do Amazon SageMaker AI.

A explicabilidade do modelo também monitora as inferências que os modelos fazem na produção para oscilação de atribuição de recursos. Também fornece ferramentas para ajudar você a gerar relatórios de governança de modelos que podem ser usados para informar equipes de risco e conformidade e reguladores externos.

Quando você especifica a opção AUTO ON ou AUTO OFF ao usar a instrução CREATE MODEL, após a conclusão do trabalho de treinamento do modelo, o SageMaker AI cria a saída da explicação. É possível usar a função EXPLAIN_MODEL para consultar o relatório de explicabilidade em formato JSON. Para obter mais informações, consulte Funções de machine learning.

Métricas de probabilidade do Amazon Redshift ML

Em problemas de aprendizado supervisionado, os rótulos de classes são resultados de previsões que usam os dados de entrada. Por exemplo, se você está usando um modelo para prever se um

cliente se inscreveria novamente em um serviço de streaming, possíveis rótulos são “provável” e “pouco provável”. O Redshift ML oferece o recurso de métricas de probabilidade, que atribuem uma probabilidade a cada rótulo para indicar sua probabilidade. Isso ajuda você a tomar decisões mais informadas com base nos resultados previstos. No Amazon Redshift ML, as métricas de probabilidade estão disponíveis ao criar modelos AUTO ON com um tipo de problema de classificação binária ou classificação multiclasse. Se você omitir o parâmetro AUTO ON, o Redshift ML assumirá que o modelo deve ter AUTO ON.

Criar o modelo

Ao criar um modelo, o Amazon Redshift detecta automaticamente o tipo de modelo e o tipo de problema. Se for um problema de classificação, o Redshift criará automaticamente uma segunda função de inferência que você poderá usar para gerar probabilidades relativas a cada rótulo. O nome dessa segunda função de inferência é o nome da função de inferência especificada seguido pela string `_probabilities`. Por exemplo, se você nomear sua função de inferência como `customer_churn_predict`, o nome da segunda função de inferência será `customer_churn_predict_probabilities`. Depois, você pode consultar essa função para obter as probabilidades de cada rótulo.

```
CREATE MODEL customer_churn_model
FROM customer_activity
    PROBLEM_TYPE BINARY_CLASSIFICATION
TARGET churn
FUNCTION customer_churn_predict
IAM_ROLE {default}
AUTO ON
SETTINGS ( S3_BUCKET 'amzn-s3-demo-bucket'
```

Obter probabilidades

Depois que a função de probabilidade estiver pronta, a execução do comando retorna um tipo SUPER que contém matrizes das probabilidades retornadas e dos respectivos rótulos. Por exemplo, o resultado `"probabilities" : [0.7, 0.3], "labels" : ["False.", "True."]` significa que o rótulo False tem uma probabilidade de 0,7 e o rótulo True tem uma probabilidade de 0,3.

```
SELECT customer_churn_predict_probabilities(Account_length, Area_code,
            VMail_message, Day_mins, Day_calls, Day_charge,Eve_mins, Eve_calls,
            Eve_charge, Night_mins, Night_calls, Night_charge,Intl_mins, Intl_calls,
            Intl_charge, Cust_serv_calls)
```

```
FROM customer_activity;

customer_churn_predict_probabilities
 --------------------
 {"probabilities" : [0.7, 0.3], "labels" : ["False.", "True."]}
 {"probabilities" : [0.8, 0.2], "labels" : ["False.", "True."]}
 {"probabilities" : [0.75, 0.25], "labels" : ["True.", "False"]}
```

As matrizes de probabilidades e rótulos são sempre ordenadas em ordem decrescente de probabilidade. Você pode escrever uma consulta para retornar apenas o rótulo previsto com a maior probabilidade desaninhando os resultados retornados de SUPER da função de probabilidade.

```
SELECT prediction.labels[0], prediction.probabilities[0]
            FROM (SELECT customer_churn_predict_probabilities(Account_length,
 Area_code,
            VMail_message, Day_mins, Day_calls, Day_charge,Eve_mins, Eve_calls,
            Eve_charge, Night_mins, Night_calls, Night_charge,Intl_mins, Intl_calls,
            Intl_charge, Cust_serv_calls) AS prediction
FROM customer_activity);

  labels   | probabilities
-----------+--------------
 "False."  | 0.7
 "False."  | 0.8
 "True."   | 0.75
```

Para simplificar as consultas, você pode armazenar os resultados da função de previsão em uma tabela.

```
CREATE TABLE churn_auto_predict_probabilities AS
            (SELECT customer_churn_predict_probabilities(Account_length, Area_code,
            VMail_message, Day_mins, Day_calls, Day_charge,Eve_mins, Eve_calls,
            Eve_charge, Night_mins, Night_calls, Night_charge,Intl_mins,
            Intl_calls, Intl_charge, Cust_serv_calls) AS prediction
FROM customer_activity);
```

Você pode consultar a tabela com os resultados para retornar somente previsões que tenham uma probabilidade maior que 0,7.

```
SELECT prediction.labels[0], prediction.probabilities[0]
FROM churn_auto_predict_probabilities
```

```
WHERE prediction.probabilities[0] > 0.7;

  labels   | probabilities
-----------+--------------
 "False."  | 0.8
 "True."   | 0.75
```

Usando a notação de índices, você pode obter a probabilidade de um rótulo específico. O exemplo a seguir retorna as probabilidades de todos os rótulos `True.`

```
SELECT label, index, p.prediction.probabilities[index]
FROM churn_auto_predict_probabilities p, p.prediction.labels AS label AT index
WHERE label='True.';

  label  | index | probabilities
---------+-------+---------------
 "True." |     0 | 0.3
 "True." |     0 | 0.2
 "True." |     0 | 0.75
```

O exemplo a seguir retorna todas as linhas que têm um rótulo `True` com uma probabilidade maior que 0,7, indicando que é provável que o cliente retorne.

```
SELECT prediction.labels[0], prediction.probabilities[0]
FROM churn_auto_predict_probabilities
WHERE prediction.probabilities[0] > 0.7 AND prediction.labels[0] = "True.";

labels     | probabilities
-----------+--------------
 "True."   | 0.75
```

Tutoriais para o Amazon Redshift ML

Você pode usar o Amazon Redshift ML para treinar modelos de machine learning usando instruções SQL e chamá-los em consultas SQL para previsão. Machine Learning no Amazon Redshift treina um modelo com um comando SQL. O Amazon Redshift inicia automaticamente um trabalho de treinamento no Amazon SageMaker AI e gera um modelo. Depois de criar um modelo, você pode realizar previsões no Amazon Redshift usando a função de previsão do modelo.

Siga as etapas desses tutoriais para saber mais sobre os recursos do Amazon Redshift ML:

- Tutorial: Como criar modelos de rotatividade de clientes: neste tutorial, você usará o Amazon Redshift ML para criar um modelo de rotatividade de clientes com o comando CREATE MODEL e executar consultas de previsão para cenários de usuários. Em seguida, você implementa consultas usando a função SQL gerada pelo comando CREATE MODEL.
- Tutorial: Como construir modelos de clusterização K-means: neste tutorial, você usará o Amazon Redshift ML para criar, treinar e implantar um modelo de machine learning com base no algoritmo k-means.
- Tutorial: Como criar modelos de classificação multiclasse: neste tutorial, você usará o Amazon Redshift ML para criar um modelo de machine learning que solucione problemas de classificação multiclasse. O algoritmo de classificação multiclasse enquadra os pontos de dados em uma das três ou mais classes. Em seguida, você implementa consultas usando a função SQL gerada pelo comando CREATE MODEL.
- Tutorial: Como construir modelos XGBoost: neste tutorial, você criará um modelo com dados do Amazon S3 e executará consultas de previsão com o modelo usando o Amazon Redshift ML. O algoritmo XGBoost é uma implementação otimizada do algoritmo baseado em árvores com aumento de gradiente.
- Tutorial: Como criar modelos de regressão: neste tutorial, você usará o Amazon Redshift ML para criar um modelo de regressão de machine learning e executar consultas de previsão no modelo. Com modelos de regressão, é possível prever resultados numéricos, como o preço de uma casa ou quantas pessoas usarão o serviço de aluguel de bicicletas de uma cidade.
- Tutorial: Como construir modelos de regressão com o aprendizado linear: neste tutorial, você criará um modelo de aprendizado linear com dados do Amazon S3 e executará consultas de previsão com o modelo usando o Amazon Redshift ML. O algoritmo de aprendizagem linear do SageMaker AI resolve problemas de regressão ou de classificação multiclasse.
- Tutorial: Como criar modelos de classificação com aprendizado linear: neste tutorial, você criará um modelo de aprendizado linear com dados do Amazon S3 e executará consultas de previsão com o modelo usando o Amazon Redshift ML. O algoritmo de aprendizagem linear do SageMaker AI resolve problemas de regressão ou classificação.

Tutorial: Como criar modelos de rotatividade de clientes

Neste tutorial, você usará o Amazon Redshift ML para criar um modelo de rotatividade de clientes com o comando CREATE MODEL e executar consultas de previsão para cenários de usuários. Em seguida, você implementa consultas usando a função SQL gerada pelo comando CREATE MODEL.

Você pode usar um comando CREATE MODEL simples para exportar dados de treinamento, treinar um modelo, importar o modelo e preparar uma função de previsão do Amazon Redshift. Use a instrução CREATE MODEL para especificar dados de treinamento como uma tabela ou instrução SELECT.

Esse exemplo usa informações históricas para construir um modelo de machine learning de rotatividade de clientes de uma operadora de telefonia móvel. Primeiro, o SageMaker AI treina o modelo de machine learning e depois testa o modelo usando as informações de perfil de um cliente arbitrário. Após a validação do modelo, o Amazon SageMaker AI implanta o modelo e a função de previsão no Amazon Redshift. Você pode usar a função de previsão para prever se um cliente vai se desligar ou não.

Exemplos de casos de uso

Você pode resolver outros problemas de classificação binária usando o Amazon Redshift ML, como prever se é possível ou não fechar um lead de vendas. Você também pode prever se uma transação financeira é fraudulenta ou não.

Tarefas

- Pré-requisitos
- Etapa 1: Carregar dados do Amazon S3 para o Amazon Redshift
- Etapa 2: Criar o modelo de machine learning
- Etapa 3: Executar previsões com o modelo

Pré-requisitos

Para concluir as etapas neste tutorial, você precisa atender aos seguintes pré-requisitos:

- Você deve configurar um cluster do Amazon Redshift para o Amazon Redshift ML. Para isso, use a documentação Definir cluster e configuração para administração do Amazon Redshift ML.
- O cluster do Amazon Redshift que você usa para criar o modelo e o bucket do Amazon S3 usado para preparar os dados de treinamento e os artefatos do modelo devem estar na mesma região da AWS.
- Para baixar os comandos SQL e o conjunto de dados de exemplo usados nessa documentação, siga um destes procedimentos:
  - Baixe os Comandos SQL, o Arquivo de atividade do cliente e o Arquivo Abalone.

- Usando a AWS CLI para Amazon S3, execute o comando a seguir. Você pode usar seu próprio caminho de destino.

```
aws s3 cp s3://redshift-downloads/redshift-ml/tutorial-scripts/redshift-ml-
tutorial.sql </target/path>
aws s3 cp s3://redshift-downloads/redshift-ml/customer_activity/
customer_activity.csv </target/path>
aws s3 cp s3://redshift-downloads/redshift-ml/abalone_xgb/abalone_xgb.csv </target/
path>
```

Etapa 1: Carregar dados do Amazon S3 para o Amazon Redshift

Use o editor de consultas v2 do Amazon Redshift para editar e executar consultas e visualizar resultados.

A execução das consultas a seguir cria um perfil chamado `customer_activity` e ingere o conjunto de dados de exemplo do Amazon S3.

```
DROP TABLE IF EXISTS customer_activity;

CREATE TABLE customer_activity (
state varchar(2),
account_length int,
area_code int,
phone varchar(8),
intl_plan varchar(3),
vMail_plan varchar(3),
vMail_message int,
day_mins float,
day_calls int,
day_charge float,
total_charge float,
eve_mins float,
eve_calls int,
eve_charge float,
night_mins float,
night_calls int,
night_charge float,
intl_mins float,
intl_calls int,
intl_charge float,
```

```
cust_serv_calls int,
churn varchar(6),
record_date date
);

COPY customer_activity
FROM 's3://redshift-downloads/redshift-ml/customer_activity/'
REGION 'us-east-1' IAM_ROLE default
FORMAT AS CSV IGNOREHEADER 1;
```

Etapa 2: Criar o modelo de machine learning

A rotatividade é nossa entrada de destino nesse modelo. Todas as outras entradas para o modelo são atributos que ajudam a criar uma função para prever a rotatividade.

O exemplo a seguir usa a operação CREATE MODEL para fornecer um modelo que prevê se um cliente estará ativo, usando entradas como idade, código postal, gastos e casos do cliente. No exemplo a seguir, substitua amzn-s3-demo-bucket por seu próprio bucket do Amazon S3.

```
CREATE MODEL customer_churn_auto_model
FROM
    (
      SELECT state,
             account_length,
             area_code,
             total_charge/account_length AS average_daily_spend,
             cust_serv_calls/account_length AS average_daily_cases,
             churn
      FROM customer_activity
      WHERE  record_date < '2020-01-01'
     )
TARGET churn FUNCTION ml_fn_customer_churn_auto
IAM_ROLE default SETTINGS (
  S3_BUCKET 'amzn-s3-demo-bucket'
);
```

A consulta SELECT no exemplo anterior cria os dados de treinamento. A cláusula TARGET especifica qual coluna é o “rótulo” de machine learning que CREATE MODEL usa para aprender a prever. A coluna de destino “rotatividade” indica se o cliente ainda tem uma associação ativa ou suspendeu a associação. O campo S3_BUCKET é o nome do bucket do Amazon S3 que você criou anteriormente. O bucket do Amazon S3 é usado para compartilhar dados e artefatos de treinamento

entre o Amazon Redshift e o Amazon SageMaker AI. As colunas restantes são os recursos que são usados para a previsão.

Para obter um resumo da sintaxe e dos recursos de um caso de uso simples do comando CREATE MODEL, consulte Simple CREATE MODEL (CREATE MODEL simples).

Adicionar permissões para criptografia do lado do servidor (opcional)

O Amazon Redshift usa por padrão o Amazon SageMaker AI Autopilot para treinamento. Observe, especificamente, que o Amazon Redshift exporta os dados de treinamento com segurança para o bucket do Amazon S3 especificado pelo cliente. Se você não especificar um `KMS_KEY_ID`, os dados serão criptografados usando a criptografia do lado do servidor SSE-S3 por padrão.

Se sua entrada for criptografada usando a criptografia do lado do servidor com uma chave gerenciada pelo AWS KMS (SSE-KMS), adicione as seguintes permissões:

```
{
    "Effect": "Allow",
    "Action": [
    "kms:Encrypt"
    "kms:Decrypt"
    ]
}
```

Para obter mais informações sobre os perfis do Amazon SageMaker AI, consulte Como usar os perfis de execução do SageMaker no Guia do desenvolvedor do Amazon SageMaker AI.

Conferir o status do modelo de treinamento (opcional)

Você pode usar o comando SHOW MODEL para saber quando o modelo está pronto.

Use a operação a seguir para verificar o status do modelo.

```
SHOW MODEL customer_churn_auto_model;
```

Veja a seguir um exemplo de saída da operação anterior.

```
+--------------------------
+-----------------------------------------------------------------------------------------------
+
```

```
|           Key            |
                 Value
             |
+-------------------------
+-----------------------------------------------------------------------------------------------
+
|        Model Name        |
        customer_churn_auto_model
             |
|       Schema Name        |
                 public
             |
|          Owner           |
                 awsuser
             |
|      Creation Time       |
        Tue, 14.06.2022 17:15:52
             |
|       Model State        |
                TRAINING
             |
|                          |

             |
|      TRAINING DATA:      |

             |
|          Query           | SELECT STATE, ACCOUNT_LENGTH, AREA_CODE, TOTAL_CHARGE /
 ACCOUNT_LENGTH AS AVERAGE_DAILY_SPEND, CUST_SERV_CALLS / ACCOUNT_LENGTH AS
 AVERAGE_DAILY_CASES, CHURN |
|                          |
         FROM CUSTOMER_ACTIVITY
             |
|                          |
    WHERE RECORD_DATE < '2020-01-01'
             |
|      Target Column       |
                 CHURN
             |
|                          |

             |
```

```
|       PARAMETERS:        |

             |
|        Model Type        |
                  auto
             |
|       Problem Type       |

             |
|        Objective         |

             |
|     AutoML Job Name      |
     redshiftml-20220614171552640901
             |
|      Function Name       |
        ml_fn_customer_churn_auto
             |
|   Function Parameters    |                                                  state
 account_length area_code average_daily_spend average_daily_cases
                       |
| Function Parameter Types |
      varchar int4 int4 float8 int4
             |
|         IAM Role         |
          default-aws-iam-role
             |
|        S3 Bucket         |
             amzn-s3-demo-bucket
              |
|       Max Runtime        |
                  5400
             |
+--------------------------
+-----------------------------------------------------------------------------------------------------
+
```

Ao concluir o treinamento do modelo, a variável `model_state` se tornará `Model is Ready` e a função de previsão se tornará disponível.

Etapa 3: Executar previsões com o modelo

Você pode usar instruções SQL para exibir as previsões feitas pelo respectivo modelo. Neste exemplo, a função de previsão criada pela operação CREATE MODEL é denominada `ml_fn_customer_churn_auto`. Os argumentos de entrada para a função de previsão correspondem aos tipos de recurso, como varchar para `state` e integer para `account_length`. A saída da função de previsão é do mesmo tipo que a coluna TARGET da instrução CREATE MODEL.

1. Como você treinou o modelo em dados anteriores a 1/1/2020, agora você usa a função de previsão no conjunto de testes. A consulta a seguir exibe as previsões sobre se os clientes cadastrados após 1.º/1/2020 passarão por rotatividade ou não.

```
SELECT
    phone,
    ml_fn_customer_churn_auto(
        state,
        account_length,
        area_code,
        total_charge / account_length,
        cust_serv_calls / account_length
    ) AS active
FROM
    customer_activity
WHERE
    record_date > '2020-01-01';
```

2. O exemplo a seguir usa a mesma função de previsão para um caso de uso diferente. Neste caso, o Amazon Redshift prevê a proporção de rotatividade e não rotatividade entre clientes de diferentes estados onde a data de registro é posterior a 1.º/1/2020.

```
WITH predicted AS (
    SELECT
        state,
        ml_fn_customer_churn_auto(
            state,
            account_length,
            area_code,
            total_charge / account_length,
            cust_serv_calls / account_length
        ) :: varchar(6) AS active
    FROM
        customer_activity
```

```
    WHERE
        record_date > '2020-01-01'
)
SELECT
    state,
    SUM(
        CASE
            WHEN active = 'True.' THEN 1
            ELSE 0
        END
    ) AS churners,
    SUM(
        CASE
            WHEN active = 'False.' THEN 1
            ELSE 0
        END
    ) AS nonchurners,
    COUNT(*) AS total_per_state
FROM
    predicted
GROUP BY
    state
ORDER BY
    state;
```

3. O exemplo a seguir usa a função de previsão para o caso de uso de previsão da porcentagem de clientes que se desligam em um estado. Neste caso, o Amazon Redshift prevê a porcentagem de rotatividade quando a data de registro é posterior a 1/1/2020.

```
WITH predicted AS (
    SELECT
        state,
        ml_fn_customer_churn_auto(
            state,
            account_length,
            area_code,
            total_charge / account_length,
            cust_serv_calls / account_length
        ) :: varchar(6) AS active
    FROM
        customer_activity
    WHERE
        record_date > '2020-01-01'
```

```
)
SELECT
    state,
    CAST((CAST((SUM(
        CASE
            WHEN active = 'True.' THEN 1
            ELSE 0
        END
    )) AS FLOAT) / CAST(COUNT(*) AS FLOAT)) AS DECIMAL (3, 2)) AS pct_churn,
    COUNT(*) AS total_customers_per_state
FROM
    predicted
GROUP BY
    state
ORDER BY
    3 DESC;
```

Tópicos relacionados

Para obter mais informações sobre o Amazon Redshift ML, consulte a seguinte documentação:

- Custos para usar o Amazon Redshift ML
- Comando CREATE MODEL
- Função EXPLAIN_MODEL

Para obter mais informações sobre machine learning, consulte a seguinte documentação:

- Visão geral do Machine Learning
- Machine Learning para iniciantes e especialistas
- What Is Fairness and Model Explainability for Machine Learning Predictions? (O que é equidade e explicabilidade de modelo para previsões de machine learning?

Tutorial: Como construir modelos de clusterização K-means

Neste tutorial, você usará o Amazon Redshift ML para criar, treinar e implantar um modelo de machine learning com base no algoritmo K-means. Esse algoritmo resolve problemas de clusterização em que você deseja detectar agrupamentos nos dados. O K-means ajuda a agrupar

dados que ainda não foram rotulados. Para saber mais sobre o clustering do k-means, consulte Como funciona o clustering do k-means no Guia do desenvolvedor do Amazon SageMaker AI.

Você usará uma operação CREATE MODEL para criar um modelo K-means com base em um cluster do Amazon Redshift. Você pode usar um comando CREATE MODEL para exportar dados de treinamento, treinar um modelo, importar o modelo e preparar uma função de previsão do Amazon Redshift. Use a operação CREATE MODEL para especificar dados de treinamento como uma tabela ou instrução SELECT.

Neste tutorial, você usa o K-means no conjunto de dados Banco de dados global de eventos, linguagem e tom (GDELT), que monitora notícias mundiais em todo o planeta, e os dados são armazenados a cada segundo, diariamente. O K-means agrupará eventos que tenham tom, atores ou locais semelhantes. Os dados são armazenados como vários arquivos no Amazon Simple Storage Service, em duas pastas diferentes. As pastas são históricas, que cobrem os anos de 1979 a 2013, e atualizações diárias, que cobrem os anos de 2013 e posteriores. Esse exemplo usa o formato histórico e inclui dados de 1979.

Exemplos de casos de uso

Você pode resolver outros problemas de clusterização com o Amazon Redshift ML, como agrupar clientes com hábitos de visualização semelhantes em um serviço de streaming. Você também pode usar o Redshift ML para prever o número ideal de centros de expedição para um serviço de entrega.

Tarefas

- Pré-requisitos
- Etapa 1: Carregar dados do Amazon S3 para o Amazon Redshift
- Etapa 2: Criar o modelo de machine learning
- Etapa 3: Executar previsões com o modelo

Pré-requisitos

Para finalizar este tutorial, você deve concluir a configuração administrativa do Amazon Redshift ML.

Etapa 1: Carregar dados do Amazon S3 para o Amazon Redshift

1. Use o editor de consultas v2 do Amazon Redshift para executar a consulta a seguir. A consulta descarta a tabela `gdelt_data` no esquema público, caso ela exista, e cria uma tabela com o mesmo nome no esquema público.

```
DROP TABLE IF EXISTS gdelt_data CASCADE;

CREATE TABLE gdelt_data (
    GlobalEventId bigint,
    SqlDate bigint,
    MonthYear bigint,
    Year bigint,
    FractionDate double precision,
    Actor1Code varchar(256),
    Actor1Name varchar(256),
    Actor1CountryCode varchar(256),
    Actor1KnownGroupCode varchar(256),
    Actor1EthnicCode varchar(256),
    Actor1Religion1Code varchar(256),
    Actor1Religion2Code varchar(256),
    Actor1Type1Code varchar(256),
    Actor1Type2Code varchar(256),
    Actor1Type3Code varchar(256),
    Actor2Code varchar(256),
    Actor2Name varchar(256),
    Actor2CountryCode varchar(256),
    Actor2KnownGroupCode varchar(256),
    Actor2EthnicCode varchar(256),
    Actor2Religion1Code varchar(256),
    Actor2Religion2Code varchar(256),
    Actor2Type1Code varchar(256),
    Actor2Type2Code varchar(256),
    Actor2Type3Code varchar(256),
    IsRootEvent bigint,
    EventCode bigint,
    EventBaseCode bigint,
    EventRootCode bigint,
    QuadClass bigint,
    GoldsteinScale double precision,
    NumMentions bigint,
    NumSources bigint,
    NumArticles bigint,
    AvgTone double precision,
    Actor1Geo_Type bigint,
    Actor1Geo_FullName varchar(256),
    Actor1Geo_CountryCode varchar(256),
    Actor1Geo_ADM1Code varchar(256),
    Actor1Geo_Lat double precision,
```

```
    Actor1Geo_Long double precision,
    Actor1Geo_FeatureID bigint,
    Actor2Geo_Type bigint,
    Actor2Geo_FullName varchar(256),
    Actor2Geo_CountryCode varchar(256),
    Actor2Geo_ADM1Code varchar(256),
    Actor2Geo_Lat double precision,
    Actor2Geo_Long double precision,
    Actor2Geo_FeatureID bigint,
    ActionGeo_Type bigint,
    ActionGeo_FullName varchar(256),
    ActionGeo_CountryCode varchar(256),
    ActionGeo_ADM1Code varchar(256),
    ActionGeo_Lat double precision,
    ActionGeo_Long double precision,
    ActionGeo_FeatureID bigint,
    DATEADDED bigint
);
```

2. A consulta a seguir carrega os dados de exemplo na tabela `gdelt_data`.

```
COPY gdelt_data
FROM 's3://gdelt-open-data/events/1979.csv'
REGION 'us-east-1'
IAM_ROLE default
CSV
DELIMITER '\t';
```

Examinar os dados de treinamento (opcional)

Para ver em quais dados seu modelo será treinado, use a consulta a seguir.

```
SELECT
    AvgTone,
    EventCode,
    NumArticles,
    Actor1Geo_Lat,
    Actor1Geo_Long,
    Actor2Geo_Lat,
    Actor2Geo_Long
FROM
    gdelt_data LIMIT 100;
```

Etapa 2: Criar o modelo de machine learning

O exemplo a seguir usa o comando CREATE MODEL para criar um modelo que agrupa os dados em sete clusters. O valor K é o número de clusters nos quais os pontos de dados são divididos. O modelo classifica os pontos de dados em clusters nos quais os pontos de dados são mais semelhantes entre si. Ao agrupar os pontos de dados em vários grupos, o algoritmo K-means determina iterativamente o melhor centro de cluster. Em seguida, o algoritmo atribui cada ponto de dados ao centro de cluster mais próximo. Os membros mais próximos do mesmo centro de cluster pertencem ao mesmo grupo. Os membros de um grupo são o mais semelhantes possível a outros membros do mesmo grupo e o mais diferentes possível de membros de outros grupos. O valor de K é subjetivo e depende de métodos que medem as semelhanças entre os pontos de dados. Você pode alterar o valor de K para suavizar os tamanhos do cluster se os clusters estiverem distribuídos de modo desigual.

No exemplo a seguir, substitua amzn-s3-demo-bucket por seu próprio bucket do Amazon S3.

```
CREATE MODEL news_data_clusters
FROM
    (
        SELECT
            AvgTone,
            EventCode,
            NumArticles,
            Actor1Geo_Lat,
            Actor1Geo_Long,
            Actor2Geo_Lat,
            Actor2Geo_Long
        FROM
            gdelt_data
    ) FUNCTION news_monitoring_cluster
    IAM_ROLE default
    AUTO OFF
    MODEL_TYPE KMEANS
    PREPROCESSORS 'none'
    HYPERPARAMETERS DEFAULT
    EXCEPT
    (K '7')
    SETTINGS (S3_BUCKET 'amzn-s3-demo-bucket');
```

Conferir o status do modelo de treinamento (opcional)

Você pode usar o comando SHOW MODEL para saber quando o modelo está pronto.

Para verificar o status do modelo, use a operação SHOW MODEL a seguir e verifique se `Model State` é `Ready`.

```
SHOW MODEL NEWS_DATA_CLUSTERS;
```

Quando o modelo estiver pronto, a saída da operação anterior deverá mostrar que o `Model State` é `Ready`. Veja a seguir um exemplo de saída da operação anterior SHOW MODEL.

```
+--------------------------
+-----------------------------------------------------------------------------------------------
+
|        Model Name        |
 news_data_clusters                                    |
+--------------------------
+-----------------------------------------------------------------------------------------------
+
|       Schema Name        |                                                  public
                                        |
|          Owner           |                                                 awsuser
                                        |
|      Creation Time       |                                    Fri, 17.06.2022
 16:32:19                                |
|       Model State        |                                                  READY
                                        |
|        train:msd         |                                               2973.822754
                                        |
|      train:progress      |                                                 100.000000
                                        |
|     train:throughput     |                                              237114.875000
                                        |
|      Estimated Cost      |                                                 0.004983
                                        |
|                          |
                                        |
|      TRAINING DATA:      |
                                        |
|          Query           | SELECT AVGTONE, EVENTCODE, NUMARTICLES, ACTOR1GEO_LAT,
 ACTOR1GEO_LONG, ACTOR2GEO_LAT, ACTOR2GEO_LONG |
```

```
|                          |                                                    FROM GDELT_DATA
                                                |
|                          |
                                                |
|       PARAMETERS:        |
                                                |
|        Model Type        |                                                         kmeans
                                                |
|    Training Job Name     |
 redshiftml-20220617163219978978-kmeans                                  |
|      Function Name       |
 news_monitoring_cluster                                          |
|   Function Parameters    |       avgtone eventcode numarticles actor1geo_lat
 actor1geo_long actor2geo_lat actor2geo_long        |
| Function Parameter Types |                              float8 int8 int8 float8 float8
 float8 float8                              |
|         IAM Role         |                                        default-aws-iam-
role                                          |
|        S3 Bucket         |                                           amzn-s3-demo-
bucket                                        |
|       Max Runtime        |                                                5400
                                                |
|                          |
                                                |
|     HYPERPARAMETERS:     |
                                                |
|       feature_dim        |                                                 7
                                                |
|            k             |                                                 7
                                                |
+--------------------------
+-----------------------------------------------------------------------------------------------
+
```

Etapa 3: Executar previsões com o modelo

Identificar os clusters

Você pode encontrar agrupamentos discretos identificados nos dados pelo seu modelo, também conhecidos como clusters. Cluster é o conjunto de pontos de dados que está mais próximo do respectivo centro de cluster do que de qualquer outro centro de cluster. Como o valor K representa o número de clusters no modelo, ele também representa o número de centros de cluster. A consulta a seguir identifica os clusters mostrando o cluster associado a cada `globaleventid`.

```
SELECT
    globaleventid,
    news_monitoring_cluster (
        AvgTone,
        EventCode,
        NumArticles,
        Actor1Geo_Lat,
        Actor1Geo_Long,
        Actor2Geo_Lat,
        Actor2Geo_Long
    ) AS cluster
FROM
    gdelt_data;
```

Conferir a distribuição dos dados

Você pode verificar a distribuição de dados entre clusters para ver se o valor K escolhido possibilitou que os dados fossem distribuídos de maneira uniforme. Use a consulta a seguir para determinar se os dados são distribuídos uniformemente pelos clusters.

```
SELECT
    events_cluster,
    COUNT(*) AS nbr_events
FROM
    (
        SELECT
            globaleventid,
            news_monitoring_cluster(
                AvgTone,
                EventCode,
                NumArticles,
                Actor1Geo_Lat,
                Actor1Geo_Long,
                Actor2Geo_Lat,
                Actor2Geo_Long
            ) AS events_cluster
        FROM
            gdelt_data
    )
GROUP BY
    1;
```

Você pode alterar o valor de K para nivelar o tamanho dos agrupamentos se os clusters estiverem distribuídos de forma desigual.

Determinar os centros de cluster

Um conjunto de pontos de dados que está mais próximo do respectivo centro de cluster do que de qualquer outro centro de cluster. Por isso, encontrar os centros de cluster ajuda a definir os clusters.

Execute a consulta a seguir para determinar os centros dos clusters com base no número de artigos por código de evento.

```
SELECT
    news_monitoring_cluster (
        AvgTone,
        EventCode,
        NumArticles,
        Actor1Geo_Lat,
        Actor1Geo_Long,
        Actor2Geo_Lat,
        Actor2Geo_Long
    ) AS events_cluster,
    eventcode,
    SUM(numArticles) AS numArticles
FROM
    gdelt_data
GROUP BY
    1,
    2;
```

Mostrar informações sobre os pontos de dados em um cluster

Use a consulta a seguir para retornar os dados dos pontos atribuídos ao quinto cluster. Os artigos selecionados devem ter dois atores.

```
SELECT
    news_monitoring_cluster (
        AvgTone,
        EventCode,
        NumArticles,
        Actor1Geo_Lat,
        Actor1Geo_Long,
        Actor2Geo_Lat,
```

```
        Actor2Geo_Long
    ) AS events_cluster,
    eventcode,
    actor1name,
    actor2name,
    SUM(numarticles) AS totalarticles
FROM
    gdelt_data
WHERE
    events_cluster = 5
    AND actor1name <> ' '
    AND actor2name <> ' '
GROUP BY
    1,
    2,
    3,
    4
ORDER BY
    5 desc;
```

Mostrar dados sobre eventos com representantes do mesmo código étnico

A consulta a seguir conta o número de artigos escritos sobre eventos com um tom positivo. A consulta também exige que os dois atores tenham o mesmo código étnico e retorna a qual cluster cada evento está atribuído.

```
SELECT
    news_monitoring_cluster (
        AvgTone,
        EventCode,
        NumArticles,
        Actor1Geo_Lat,
        Actor1Geo_Long,
        Actor2Geo_Lat,
        Actor2Geo_Long
    ) AS events_cluster,
    SUM(numarticles) AS total_articles,
    eventcode AS event_code,
    Actor1EthnicCode AS ethnic_code
FROM
    gdelt_data
WHERE
    Actor1EthnicCode = Actor2EthnicCode
```

```
    AND Actor1EthnicCode <> ' '
    AND Actor2EthnicCode <> ' '
    AND AvgTone > 0
GROUP BY
    1,
    3,
    4
HAVING
    (total_articles) > 4
ORDER BY
    1,
    2 ASC;
```

Tópicos relacionados

Para obter mais informações sobre o Amazon Redshift ML, consulte a seguinte documentação:

- Custos para usar o ML do Amazon Redshift
- Operação CREATE MODEL
- Função EXPLAIN_MODEL

Para obter mais informações sobre machine learning, consulte a seguinte documentação:

- Visão geral do Machine Learning
- Machine Learning para iniciantes e especialistas
- What Is Fairness and Model Explainability for Machine Learning Predictions? (O que é equidade e explicabilidade de modelo para previsões de machine learning?

Tutorial: Como criar modelos de classificação multiclasse

Neste tutorial, você usa o Amazon Redshift ML para criar um modelo de machine learning que solucione problemas de classificação multiclasse. O algoritmo de classificação multiclasse enquadra os pontos de dados em uma das três ou mais classes. Em seguida, você implementa consultas usando a função SQL gerada pelo comando CREATE MODEL.

Você pode usar um comando CREATE MODEL para exportar dados de treinamento, treinar um modelo, importar o modelo e preparar uma função de previsão do Amazon Redshift. Use a operação CREATE MODEL para especificar dados de treinamento como uma tabela ou instrução SELECT.

Para acompanhar o tutorial, você usa o conjunto de dados público E-Commerce Sales Forecast (Previsão de vendas de comércio eletrônico), que inclui dados de vendas de um varejista on-line do Reino Unido. O modelo que você gerar terá como alvo os clientes mais ativos para um programa especial de fidelidade do cliente. Com a classificação multiclasse, você pode usar o modelo para prever por quantos meses um cliente ficará ativo em um período de 13 meses. A função de previsão designa clientes que devem estar ativos por sete ou mais meses para admissão no programa.

Exemplos de casos de uso

Você pode resolver outros problemas de classificação multiclasse com o Amazon Redshift ML, como prever o produto mais vendido de uma linha de produtos. Você também pode prever quais frutas uma imagem contém, como selecionar maçãs, peras ou laranjas.

Tarefas

- Pré-requisitos
- Etapa 1: Carregar dados do Amazon S3 para o Amazon Redshift
- Etapa 2: Criar o modelo de machine learning
- Etapa 3: Executar previsões com o modelo

Pré-requisitos

Para finalizar este tutorial, você deve concluir a configuração administrativa do Amazon Redshift ML.

Etapa 1: Carregar dados do Amazon S3 para o Amazon Redshift

Use o editor de consultas v2 do Amazon Redshift para executar as consultas a seguir. Estas consultas carregam os dados de exemplo no Amazon Redshift.

1. A consulta a seguir cria uma tabela chamada `ecommerce_sales`.

```
CREATE TABLE IF NOT EXISTS ecommerce_sales (
    invoiceno VARCHAR(30),
    stockcode VARCHAR(30),
    description VARCHAR(60),
    quantity DOUBLE PRECISION,
    invoicedate VARCHAR(30),
    unitprice DOUBLE PRECISION,
    customerid BIGINT,
    country VARCHAR(25)
```

```
);
```

2. A consulta a seguir copia os dados de exemplo do conjunto de dados E-Commerce Sales Forecast para a tabela `ecommerce_sales`.

```
COPY ecommerce_sales
FROM
    's3://redshift-ml-multiclass/ecommerce_data.txt'
IAM_ROLE default
DELIMITER '\t'
IGNOREHEADER 1
REGION 'us-east-1'
MAXERROR 100;
```

Dividir os dados

Quando você cria um modelo no Amazon Redshift ML, o SageMaker AI divide automaticamente os dados em conjuntos de treinamento e teste, para que o SageMaker AI possa determinar a precisão do modelo. Ao dividir manualmente os dados nesta etapa, você poderá verificar a precisão do modelo alocando um conjunto de previsões adicional.

Use a instrução SQL a seguir para dividir os dados em três conjuntos para treinamento, validação e previsão.

```
--creates table with all data
CREATE TABLE ecommerce_sales_data AS (
    SELECT
        t1.stockcode,
        t1.description,
        t1.invoicedate,
        t1.customerid,
        t1.country,
        t1.sales_amt,
        CAST(RANDOM() * 100 AS INT) AS data_group_id
    FROM
        (
            SELECT
                stockcode,
                description,
                invoicedate,
                customerid,
                country,
```

```
                SUM(quantity * unitprice) AS sales_amt
            FROM
                ecommerce_sales
            GROUP BY
                1,
                2,
                3,
                4,
                5
        ) t1
);

--creates training set
CREATE TABLE ecommerce_sales_training AS (
    SELECT
        a.customerid,
        a.country,
        a.stockcode,
        a.description,
        a.invoicedate,
        a.sales_amt,
        (b.nbr_months_active) AS nbr_months_active
    FROM
        ecommerce_sales_data a
        INNER JOIN (
            SELECT
                customerid,
                COUNT(
                    DISTINCT(
                        DATE_PART(y, CAST(invoicedate AS DATE)) || '-' || LPAD(
                            DATE_PART(mon, CAST(invoicedate AS DATE)),
                            2,
                            '00'
                        )
                    )
                ) AS nbr_months_active
            FROM
                ecommerce_sales_data
            GROUP BY
                1
        ) b ON a.customerid = b.customerid
    WHERE
        a.data_group_id < 80
);
```

```
--creates validation set
CREATE TABLE ecommerce_sales_validation AS (
    SELECT
        a.customerid,
        a.country,
        a.stockcode,
        a.description,
        a.invoicedate,
        a.sales_amt,
        (b.nbr_months_active) AS nbr_months_active
    FROM
        ecommerce_sales_data a
        INNER JOIN (
            SELECT
                customerid,
                COUNT(
                    DISTINCT(
                        DATE_PART(y, CAST(invoicedate AS DATE)) || '-' || LPAD(
                            DATE_PART(mon, CAST(invoicedate AS DATE)),
                            2,
                            '00'
                        )
                    )
                ) AS nbr_months_active
            FROM
                ecommerce_sales_data
            GROUP BY
                1
        ) b ON a.customerid = b.customerid
    WHERE
        a.data_group_id BETWEEN 80
        AND 90
);

--creates prediction set
CREATE TABLE ecommerce_sales_prediction AS (
    SELECT
        customerid,
        country,
        stockcode,
        description,
        invoicedate,
        sales_amt
```

```
    FROM
        ecommerce_sales_data
    WHERE
        data_group_id > 90);
```

Etapa 2: Criar o modelo de machine learning

Nesta etapa, você usa a instrução CREATE MODEL para criar seu modelo de machine learning usando a classificação multiclasse.

A consulta a seguir cria o modelo de classificação multiclasse com o conjunto de treinamento usando a operação CREATE MODEL. Substitua amzn-s3-demo-bucket por seu próprio bucket do Amazon S3.

```
CREATE MODEL ecommerce_customer_activity
FROM
    (
        SELECT
            customerid,
            country,
            stockcode,
            description,
            invoicedate,
            sales_amt,
            nbr_months_active
        FROM
            ecommerce_sales_training
    ) TARGET nbr_months_active FUNCTION predict_customer_activity IAM_ROLE default
 PROBLEM_TYPE MULTICLASS_CLASSIFICATION SETTINGS (
        S3_BUCKET 'amzn-s3-demo-bucket',
        S3_GARBAGE_COLLECT OFF
    );
```

Nessa consulta, você especifica o tipo de problema como `Multiclass_Classification`. O destino que você prevê para o modelo é `nbr_months_active`. Quando o SageMaker AI termina de treinar o modelo, ele cria a função `predict_customer_activity`, que você usará para fazer previsões no Amazon Redshift.

Mostrar o status do modelo de treinamento (opcional)

Você pode usar o comando SHOW MODEL para saber quando o modelo está pronto.

Use a consulta a seguir para retornar várias métricas do modelo, incluindo o estado e a precisão.

```
SHOW MODEL ecommerce_customer_activity;
```

Quando o modelo estiver pronto, a saída da operação anterior deverá mostrar que o `Model State` é `Ready`. Veja a seguir um exemplo de saída da operação anterior SHOW MODEL.

```
+--------------------------
+-----------------------------------------------------------------------------------------------
+
|        Model Name        |
 ecommerce_customer_activity                                |
+--------------------------
+-----------------------------------------------------------------------------------------------
+
|       Schema Name        |                                                  public
                                  |
|          Owner           |                                                  awsuser
                                  |
|      Creation Time       |                                   Fri, 17.06.2022 19:02:15
                                  |
|       Model State        |                                                   READY
                                  |
|   Training Job Status    |
 MaxAutoMLJobRuntimeReached                                 |
|   validation:accuracy    |                                           0.991280
                                  |
|      Estimated Cost      |                                           7.897689
                                  |
|                          |
                                  |
|      TRAINING DATA:      |
                                  |
|          Query           | SELECT CUSTOMERID, COUNTRY, STOCKCODE, DESCRIPTION,
 INVOICEDATE, SALES_AMT, NBR_MONTHS_ACTIVE |
|                          |                                      FROM
 ECOMMERCE_SALES_TRAINING                                 |
|      Target Column       |                                           NBR_MONTHS_ACTIVE
                                  |
|                          |
                                  |
|       PARAMETERS:        |
                                  |
```

```
|        Model Type        |                                        xgboost
                                  |
|       Problem Type       |                                MulticlassClassification
                                  |
|        Objective         |                                       Accuracy
                                  |
|     AutoML Job Name      |
 redshiftml-20220617190215268770                                |
|      Function Name       |
 predict_customer_activity                                |
|   Function Parameters    |               customerid country stockcode description
 invoicedate sales_amt               |
| Function Parameter Types |                         int8 varchar varchar varchar
 varchar float8                        |
|         IAM Role         |                                  default-aws-iam-role
                                  |
|        S3 Bucket         |                                      amzn-s3-demo-
bucket                                   |
|       Max Runtime        |                                         5400
                                  |
+--------------------------
+-------------------------------------------------------------------------------------------------
+
```

Etapa 3: Executar previsões com o modelo

A consulta a seguir mostra quais clientes se qualificam para seu programa de fidelidade do cliente. Se o modelo previr que o cliente ficará ativo por pelo menos sete meses, então ele selecionará o cliente para o programa de fidelidade.

```
SELECT
    customerid,
    predict_customer_activity(
        customerid,
        country,
        stockcode,
        description,
        invoicedate,
        sales_amt
    ) AS predicted_months_active
FROM
    ecommerce_sales_prediction
WHERE
```

```
    predicted_months_active >= 7
GROUP BY
    1,
    2
LIMIT
    10;
```

Executar consultas de previsão nos dados de validação (opcional)

Execute as consultas de previsão a seguir nos dados de validação para ver o nível de precisão do modelo.

```
SELECT
    CAST(SUM(t1.match) AS decimal(7, 2)) AS predicted_matches,
    CAST(SUM(t1.nonmatch) AS decimal(7, 2)) AS predicted_non_matches,
    CAST(SUM(t1.match + t1.nonmatch) AS decimal(7, 2)) AS total_predictions,
    predicted_matches / total_predictions AS pct_accuracy
FROM
    (
        SELECT
            customerid,
            country,
            stockcode,
            description,
            invoicedate,
            sales_amt,
            nbr_months_active,
            predict_customer_activity(
                customerid,
                country,
                stockcode,
                description,
                invoicedate,
                sales_amt
            ) AS predicted_months_active,
            CASE
                WHEN nbr_months_active = predicted_months_active THEN 1
                ELSE 0
            END AS match,
            CASE
                WHEN nbr_months_active <> predicted_months_active THEN 1
                ELSE 0
            END AS nonmatch
```

```
        FROM
            ecommerce_sales_validation
    )t1;
```

Prever quantos clientes perderam a entrada (opcional)

A consulta a seguir compara o número de clientes que, segundo a previsão, ficarão ativos por apenas cinco ou seis meses. O modelo prevê que esses clientes perderão a oportunidade de serem incluídos no programa de fidelidade. A consulta então compara o número de clientes que quase conseguiram entrar no programa com o número previsto dos que estarão qualificados para o programa de fidelidade. Essa consulta pode ser usada para fundamentar uma decisão sobre se o limite do programa de fidelidade deve ou não ser reduzido. Você também pode determinar se há uma quantidade significativa de clientes que quase vão conseguir entrar no programa. Você pode então incentivar esses clientes a aumentar suas atividades para obter uma associação ao programa de fidelidade.

```
SELECT
    predict_customer_activity(
        customerid,
        country,
        stockcode,
        description,
        invoicedate,
        sales_amt
    ) AS predicted_months_active,
    COUNT(customerid)
FROM
    ecommerce_sales_prediction
WHERE
    predicted_months_active BETWEEN 5 AND 6
GROUP BY
    1
ORDER BY
    1 ASC
LIMIT
    10)
UNION
(SELECT
      NULL AS predicted_months_active,
    COUNT (customerid)
FROM
    ecommerce_sales_prediction
```

```
WHERE
    predict_customer_activity(
        customerid,
        country,
        stockcode,
        description,
        invoicedate,
        sales_amt
    ) >=7);
```

Tópicos relacionados

Para obter mais informações sobre o Amazon Redshift ML, consulte a seguinte documentação:

- Custos para usar o ML do Amazon Redshift
- Operação CREATE MODEL
- Função EXPLAIN_MODEL

Para obter mais informações sobre machine learning, consulte a seguinte documentação:

- Visão geral do Machine Learning
- Machine Learning para iniciantes e especialistas
- What Is Fairness and Model Explainability for Machine Learning Predictions? (O que é equidade e explicabilidade de modelo para previsões de machine learning?

Tutorial: Como construir modelos XGBoost

Neste tutorial, você cria um modelo com dados do Amazon S3 e executa consultas de previsão com o modelo usando o Amazon Redshift ML. O algoritmo XGBoost é uma implementação otimizada do algoritmo baseado em árvores com aumento de gradiente. O XGBoost lida com mais tipos de dados, relacionamentos e distribuições do que outros algoritmos de árvores com aumento de gradiente. Você pode usar o XGBoost para regressão, classificação binária, classificação multiclasse e problemas de classificação. Para obter mais informações sobre o algoritmo XGBoost, consulte Algoritmo XGBoost no Guia do desenvolvedor do Amazon SageMaker AI.

A operação `CREATE MODEL` do Amazon Redshift ML com a opção `AUTO OFF` no momento aceita o XGBoost como `MODEL_TYPE`. Você pode fornecer informações relevantes, como o objetivo e os hiperparâmetros, como parte do `CREATE MODEL`, com base em seu caso de uso.

Neste tutorial, use o conjunto de dados banknote authentication (autenticação de cédulas), que é um problema de classificação binária para prever se determinada cédula é genuína ou falsificada.

Exemplos de casos de uso

Você pode resolver outros problemas de classificação binária usando o Amazon Redshift ML, como prever se um paciente está saudável ou tem alguma doença. Você também pode prever se um e-mail é ou não spam.

Tarefas

- Pré-requisitos
- Etapa 1: Carregar dados do Amazon S3 para o Amazon Redshift
- Etapa 2: Criar o modelo de machine learning
- Etapa 3: Executar previsões com o modelo

Pré-requisitos

Para finalizar este tutorial, você deve concluir a configuração administrativa do Amazon Redshift ML.

Etapa 1: Carregar dados do Amazon S3 para o Amazon Redshift

Use o editor de consultas v2 do Amazon Redshift para executar as consultas a seguir.

A consulta a seguir cria duas tabelas, carrega os dados do Amazon S3 e divide os dados em um conjunto de treinamento e um conjunto de testes. Você usará o conjunto de treinamento para treinar seu modelo e criar a função de previsão. Em seguida, você testará a função de previsão no conjunto de testes.

```
--create training set table
CREATE TABLE banknoteauthentication_train(
    variance FLOAT,
    skewness FLOAT,
    curtosis FLOAT,
    entropy FLOAT,
    class INT
);

--Load into training table
```

```
COPY banknoteauthentication_train
FROM
    's3://redshiftbucket-ml-sagemaker/banknote_authentication/train_data/' IAM_ROLE
 default REGION 'us-west-2' IGNOREHEADER 1 CSV;

--create testing set table
CREATE TABLE banknoteauthentication_test(
    variance FLOAT,
    skewness FLOAT,
    curtosis FLOAT,
    entropy FLOAT,
    class INT
);

--Load data into testing table
COPY banknoteauthentication_test
FROM
    's3://redshiftbucket-ml-sagemaker/banknote_authentication/test_data/'
    IAM_ROLE default
    REGION 'us-west-2'
    IGNOREHEADER 1
    CSV;
```

Etapa 2: Criar o modelo de machine learning

A consulta a seguir cria o modelo XGBoost no Amazon Redshift ML com base no conjunto de treinamento que você criou na etapa anterior. Substitua `amzn-s3-demo-bucket` por seu próprio `S3_BUCKET`, que armazenará os conjuntos de dados de entrada e outros artefatos do Redshift ML.

```
CREATE MODEL model_banknoteauthentication_xgboost_binary
FROM
    banknoteauthentication_train
    TARGET class
    FUNCTION func_model_banknoteauthentication_xgboost_binary
    IAM_ROLE default
    AUTO OFF
    MODEL_TYPE xgboost
    OBJECTIVE 'binary:logistic'
    PREPROCESSORS 'none'
    HYPERPARAMETERS DEFAULT
EXCEPT(NUM_ROUND '100')
SETTINGS(S3_BUCKET 'amzn-s3-demo-bucket');
```

Mostrar o status do modelo de treinamento (opcional)

Você pode usar o comando SHOW MODEL para saber quando o modelo está pronto.

Use a consulta a seguir para monitorar o andamento do treinamento do modelo.

```
SHOW MODEL model_banknoteauthentication_xgboost_binary;
```

Se o modelo for READY, a operação SHOW MODEL também fornecerá a métrica `train:error`, conforme mostrado no exemplo de saída a seguir. A métrica `train:error` é uma medida de precisão de seu modelo que mede até seis casas decimais. O valor 0 é o mais preciso e o 1 é o menos preciso.

```
+--------------------------+-------------------------------------------------+
|        Model Name        |   model_banknoteauthentication_xgboost_binary   |
+--------------------------+-------------------------------------------------+
| Schema Name              | public                                          |
| Owner                    | awsuser                                         |
| Creation Time            | Tue, 21.06.2022 19:07:35                        |
| Model State              | READY                                           |
| train:error              |                                        0.000000 |
| Estimated Cost           |                                        0.006197 |
|                          |                                                 |
| TRAINING DATA:           |                                                 |
| Query                    | SELECT *                                        |
|                          | FROM "BANKNOTEAUTHENTICATION_TRAIN"             |
| Target Column            | CLASS                                           |
|                          |                                                 |
| PARAMETERS:              |                                                 |
| Model Type               | xgboost                                         |
| Training Job Name        | redshiftml-20220621190735686935-xgboost         |
| Function Name            | func_model_banknoteauthentication_xgboost_binary |
| Function Parameters      | variance skewness curtosis entropy              |
| Function Parameter Types | float8 float8 float8 float8                     |
| IAM Role                 | default-aws-iam-role                            |
| S3 Bucket                | amzn-s3-demo-bucket                             |
| Max Runtime              |                                            5400 |
|                          |                                                 |
| HYPERPARAMETERS:         |                                                 |
| num_round                |                                             100 |
| objective                | binary:logistic                                 |
+--------------------------+-------------------------------------------------+
```

Etapa 3: Executar previsões com o modelo

Conferir a precisão do modelo

A consulta de previsão a seguir usa a função de previsão criada na etapa anterior para verificar a precisão do modelo. Execute essa consulta no conjunto de testes para verificar se o modelo não tem uma correspondência muito próxima do conjunto de treinamento. Essa correspondência próxima também é conhecida como sobreajuste, que pode levar o modelo a fazer previsões não confiáveis.

```
WITH predict_data AS (
    SELECT
        class AS label,
        func_model_banknoteauthentication_xgboost_binary (variance, skewness, curtosis,
 entropy) AS predicted,
        CASE
            WHEN label IS NULL THEN 0
            ELSE label
        END AS actual,
        CASE
            WHEN actual = predicted THEN 1 :: INT
            ELSE 0 :: INT
        END AS correct
    FROM
        banknoteauthentication_test
),
aggr_data AS (
    SELECT
        SUM(correct) AS num_correct,
        COUNT(*) AS total
    FROM
        predict_data
)
SELECT
    (num_correct :: FLOAT / total :: FLOAT) AS accuracy
FROM
    aggr_data;
```

Prever a quantidade de cédulas originais e falsas

A consulta de previsão a seguir retorna a quantidade prevista de cédulas originais e falsas no conjunto de teste.

```
WITH predict_data AS (
    SELECT
        func_model_banknoteauthentication_xgboost_binary(variance, skewness, curtosis,
 entropy) AS predicted
    FROM
        banknoteauthentication_test
)
SELECT
    CASE
        WHEN predicted = '0' THEN 'Original banknote'
        WHEN predicted = '1' THEN 'Counterfeit banknote'
        ELSE 'NA'
    END AS banknote_authentication,
    COUNT(1) AS count
FROM
    predict_data
GROUP BY
    1;
```

Encontrar a observação média de uma cédula original e uma falsificada

A consulta de previsão a seguir retorna o valor médio de cada recurso para cédulas que são previstas como originais e falsificadas no conjunto de teste.

```
WITH predict_data AS (
    SELECT
        func_model_banknoteauthentication_xgboost_binary(variance, skewness, curtosis,
 entropy) AS predicted,
          variance,
          skewness,
          curtosis,
          entropy
    FROM
        banknoteauthentication_test
)
SELECT
    CASE
        WHEN predicted = '0' THEN 'Original banknote'
```

```
        WHEN predicted = '1' THEN 'Counterfeit banknote'
        ELSE 'NA'
    END AS banknote_authentication,
    TRUNC(AVG(variance), 2) AS avg_variance,
    TRUNC(AVG(skewness), 2) AS avg_skewness,
    TRUNC(AVG(curtosis), 2) AS avg_curtosis,
    TRUNC(AVG(entropy), 2) AS avg_entropy
FROM
    predict_data
GROUP BY
    1
ORDER BY
    2;
```

Tópicos relacionados

Para obter mais informações sobre o Amazon Redshift ML, consulte a seguinte documentação:

- Custos para usar o ML do Amazon Redshift
- Operação CREATE MODEL
- Função EXPLAIN_MODEL

Para obter mais informações sobre machine learning, consulte a seguinte documentação:

- Visão geral do Machine Learning
- Machine Learning para iniciantes e especialistas
- What Is Fairness and Model Explainability for Machine Learning Predictions? (O que é equidade e explicabilidade de modelo para previsões de machine learning?

Tutorial: Como criar modelos de regressão

Neste tutorial, você usa o Amazon Redshift ML para criar um modelo de regressão de machine learning e executar consultas de previsão no modelo. Com modelos de regressão, é possível prever resultados numéricos, como o preço de uma casa ou quantas pessoas usarão o serviço de aluguel de bicicletas de uma cidade. Você usa o comando CREATE MODEL no Amazon Redshift com os dados de treinamento. Em seguida, o Amazon Redshift ML compila o modelo, importa o modelo treinado para o Redshift e prepara uma função de previsão SQL. Você pode usar a função de previsão em consultas SQL no Amazon Redshift.

Neste tutorial, você usará o Amazon Redshift ML para criar um modelo de regressão que prevê o número de pessoas que usam o serviço de compartilhamento de bicicletas da cidade de Toronto a qualquer hora do dia. As entradas para o modelo incluem feriados e condições climáticas. Você usará um modelo de regressão porque deseja um resultado numérico para esse problema.

Você pode usar o comando CREATE MODEL para exportar dados de treinamento, treinar e importar o modelo e disponibilizá-lo no Amazon Redshift como uma função SQL. Use a operação CREATE MODEL para especificar dados de treinamento como uma tabela ou instrução SELECT.

Exemplos de casos de uso

Você pode resolver outros problemas de regressão com o Amazon Redshift ML, como prever o valor da vida útil de um cliente. Você também pode usar o Redshift ML para prever o preço mais lucrativo e a receita resultante de um produto.

Tarefas

- Pré-requisitos
- Etapa 1: Carregar dados do Amazon S3 para o Amazon Redshift
- Etapa 2: Criar o modelo de machine learning
- Etapa 3: Validar o modelo

Pré-requisitos

Para finalizar este tutorial, você deve concluir a configuração administrativa do Amazon Redshift ML.

Etapa 1: Carregar dados do Amazon S3 para o Amazon Redshift

Use o editor de consultas v2 do Amazon Redshift para executar as consultas a seguir.

1. Você deve criar três tabelas para carregar os três conjuntos de dados públicos no Amazon Redshift. Os conjuntos de dados são Toronto Bike Ridership Data (Dados sobre usuários de bicicleta em Toronto), historical weather data (dados climáticos históricos) e historical holidays data (dados históricos de feriados). Execute a consulta a seguir no editor de consultas do Amazon Redshift para criar tabelas com os nomes `ridership`, `weather` e `holiday`.

```
CREATE TABLE IF NOT EXISTS ridership (
    trip_id INT,
    trip_duration_seconds INT,
    trip_start_time timestamp,
```

```
    trip_stop_time timestamp,
    from_station_name VARCHAR(50),
    to_station_name VARCHAR(50),
    from_station_id SMALLINT,
    to_station_id SMALLINT,
    user_type VARCHAR(20)
);

CREATE TABLE IF NOT EXISTS weather (
    longitude_x DECIMAL(5, 2),
    latitude_y DECIMAL(5, 2),
    station_name VARCHAR(20),
    climate_id BIGINT,
    datetime_utc TIMESTAMP,
    weather_year SMALLINT,
    weather_month SMALLINT,
    weather_day SMALLINT,
    time_utc VARCHAR(5),
    temp_c DECIMAL(5, 2),
    temp_flag VARCHAR(1),
    dew_point_temp_c DECIMAL(5, 2),
    dew_point_temp_flag VARCHAR(1),
    rel_hum SMALLINT,
    rel_hum_flag VARCHAR(1),
    precip_amount_mm DECIMAL(5, 2),
    precip_amount_flag VARCHAR(1),
    wind_dir_10s_deg VARCHAR(10),
    wind_dir_flag VARCHAR(1),
    wind_spd_kmh VARCHAR(10),
    wind_spd_flag VARCHAR(1),
    visibility_km VARCHAR(10),
    visibility_flag VARCHAR(1),
    stn_press_kpa DECIMAL(5, 2),
    stn_press_flag VARCHAR(1),
    hmdx SMALLINT,
    hmdx_flag VARCHAR(1),
    wind_chill VARCHAR(10),
    wind_chill_flag VARCHAR(1),
    weather VARCHAR(10)
);

CREATE TABLE IF NOT EXISTS holiday (holiday_date DATE, description VARCHAR(100));
```

2. A consulta a seguir carrega os dados de exemplo nas tabelas que você criou na etapa anterior.

```
COPY ridership
FROM
    's3://redshift-ml-bikesharing-data/bike-sharing-data/ridership/'
    IAM_ROLE default
    FORMAT CSV
    IGNOREHEADER 1
    DATEFORMAT 'auto'
    TIMEFORMAT 'auto'
    REGION 'us-west-2'
    gzip;

COPY weather
FROM
    's3://redshift-ml-bikesharing-data/bike-sharing-data/weather/'
    IAM_ROLE default
    FORMAT csv
    IGNOREHEADER 1
    DATEFORMAT 'auto'
    TIMEFORMAT 'auto'
    REGION 'us-west-2'
    gzip;

COPY holiday
FROM
    's3://redshift-ml-bikesharing-data/bike-sharing-data/holiday/'
    IAM_ROLE default
    FORMAT csv
    IGNOREHEADER 1
    DATEFORMAT 'auto'
    TIMEFORMAT 'auto'
    REGION 'us-west-2'
    gzip;
```

3. A consulta a seguir executa transformações nos conjuntos de dados `ridership` e `weather` para remover vieses ou anomalias. A remoção de vieses e anomalias resulta em melhor precisão do modelo. A consulta simplifica as tabelas criando duas novas visualizações chamadas `ridership_view` e `weather_view`.

```
CREATE
OR REPLACE VIEW ridership_view AS
SELECT
    trip_time,
```

```
    trip_count,
    TO_CHAR(trip_time, 'hh24') :: INT trip_hour,
    TO_CHAR(trip_time, 'dd') :: INT trip_day,
    TO_CHAR(trip_time, 'mm') :: INT trip_month,
    TO_CHAR(trip_time, 'yy') :: INT trip_year,
    TO_CHAR(trip_time, 'q') :: INT trip_quarter,
    TO_CHAR(trip_time, 'w') :: INT trip_month_week,
    TO_CHAR(trip_time, 'd') :: INT trip_week_day
FROM
    (
        SELECT
            CASE
                WHEN TRUNC(r.trip_start_time) < '2017-07-01' :: DATE THEN
 CONVERT_TIMEZONE(
                    'US/Eastern',
                    DATE_TRUNC('hour', r.trip_start_time)
                )
                ELSE DATE_TRUNC('hour', r.trip_start_time)
            END trip_time,
            COUNT(1) trip_count
        FROM
            ridership r
        WHERE
            r.trip_duration_seconds BETWEEN 60
            AND 60 * 60 * 24
        GROUP BY
            1
    );

CREATE
OR REPLACE VIEW weather_view AS
SELECT
    CONVERT_TIMEZONE(
        'US/Eastern',
        DATE_TRUNC('hour', datetime_utc)
    ) daytime,
    ROUND(AVG(temp_c)) temp_c,
    ROUND(AVG(precip_amount_mm)) precip_amount_mm
FROM
    weather
GROUP BY
    1;
```

4. A consulta a seguir cria uma tabela que combina todos os atributos de entrada relevantes de `ridership_view` e de `weather_view` na tabela `trip_data`.

```
CREATE TABLE trip_data AS
SELECT
    r.trip_time,
    r.trip_count,
    r.trip_hour,
    r.trip_day,
    r.trip_month,
    r.trip_year,
    r.trip_quarter,
    r.trip_month_week,
    r.trip_week_day,
    w.temp_c,
    w.precip_amount_mm,CASE
        WHEN h.holiday_date IS NOT NULL THEN 1
        WHEN TO_CHAR(r.trip_time, 'D') :: INT IN (1, 7) THEN 1
        ELSE 0
    END is_holiday,
    ROW_NUMBER() OVER (
        ORDER BY
            RANDOM()
    ) serial_number
FROM
    ridership_view r
    JOIN weather_view w ON (r.trip_time = w.daytime)
    LEFT OUTER JOIN holiday h ON (TRUNC(r.trip_time) = h.holiday_date);
```

Visualizar os dados de exemplo (opcional)

A consulta a seguir mostra as entradas da tabela. Você pode executar essa operação para garantir se a tabela foi feita corretamente.

```
SELECT *
FROM trip_data
LIMIT 5;
```

Veja a seguir um exemplo de saída da operação anterior.

```
+--------------------+-----------+----------+---------+-----------+----------
+-------------+----------------+--------------+--------+-----------------
+-----------+--------------+
|      trip_time      | trip_count | trip_hour | trip_day | trip_month | trip_year
 | trip_quarter | trip_month_week | trip_week_day | temp_c | precip_amount_mm |
 is_holiday | serial_number |
+--------------------+-----------+----------+---------+-----------+----------
+-------------+----------------+--------------+--------+-----------------
+-----------+--------------+
| 2017-03-21 22:00:00 |         47 |        22 |       21 |          3 |        17 |
         1 |               3 |             3 |      1 |               0 |          0 |
             1 |
| 2018-05-04 01:00:00 |         19 |         1 |        4 |          5 |        18 |
         2 |               1 |             6 |     12 |               0 |          0 |
             3 |
| 2018-01-11 10:00:00 |         93 |        10 |       11 |          1 |        18 |
         1 |               2 |             5 |      9 |               0 |          0 |
             5 |
| 2017-10-28 04:00:00 |         20 |         4 |       28 |         10 |        17 |
         4 |               4 |             7 |     11 |               0 |          1 |
             7 |
| 2017-12-31 21:00:00 |         11 |        21 |       31 |         12 |        17 |
         4 |               5 |             1 |    -15 |               0 |          1 |
             9 |
+--------------------+-----------+----------+---------+-----------+----------
+-------------+----------------+--------------+--------+-----------------
+-----------+--------------+
```

Mostrar a correlação entre atributos (opcional)

Determinar a correlação ajuda a medir a consistência da associação entre os atributos. O nível de associação pode ajudar você a determinar o que afeta o resultado pretendido. Neste tutorial, o resultado pretendido é `trip_count`.

A consulta a seguir cria ou substitui o procedimento `sp_correlation`. Use o procedimento armazenado, denominado `sp_correlation`, para mostrar a correlação entre um atributo e outros atributos em uma tabela no Amazon Redshift.

```
CREATE OR REPLACE PROCEDURE sp_correlation(source_schema_name in varchar(255),
 source_table_name in varchar(255), target_column_name in varchar(255),
 output_temp_table_name inout varchar(255)) AS $$
DECLARE
```

```
  v_sql varchar(max);
  v_generated_sql varchar(max);
  v_source_schema_name varchar(255)=lower(source_schema_name);
  v_source_table_name varchar(255)=lower(source_table_name);
  v_target_column_name varchar(255)=lower(target_column_name);
BEGIN
  EXECUTE 'DROP TABLE IF EXISTS ' || output_temp_table_name;
  v_sql = '
SELECT
  ''CREATE temp table '|| output_temp_table_name||' AS SELECT ''|| outer_calculation||
  '' FROM (SELECT COUNT(1) number_of_items, SUM('||v_target_column_name||')
 sum_target, SUM(POW('||v_target_column_name||',2)) sum_square_target, POW(SUM('||
v_target_column_name||'),2) square_sum_target,''||
  inner_calculation||
  '' FROM (SELECT ''||
  column_name||
  '' FROM '||v_source_table_name||''))''
FROM
  (
  SELECT
    DISTINCT
    LISTAGG(outer_calculation,'','') OVER () outer_calculation
    ,LISTAGG(inner_calculation,'','') OVER () inner_calculation
    ,LISTAGG(column_name,'','') OVER () column_name
  FROM
    (
    SELECT
      CASE WHEN atttypid=16 THEN ''DECODE(''||column_name||'',true,1,0)'' ELSE
 column_name END column_name
      ,atttypid
      ,''CAST(DECODE(number_of_items * sum_square_''||rn||'' - square_sum_''||
rn||'',0,null,(number_of_items*sum_target_''||rn||'' - sum_target * sum_''||rn||
        '')/SQRT((number_of_items * sum_square_target - square_sum_target) *
 (number_of_items * sum_square_''||rn||
        '' - square_sum_''||rn||''))) AS numeric(5,2)) ''||column_name
 outer_calculation
      ,''sum(''||column_name||'') sum_''||rn||'',''||
            ''SUM(trip_count*''||column_name||'') sum_target_''||rn||'',''||
            ''SUM(POW(''||column_name||'',2)) sum_square_''||rn||'',''||
            ''POW(SUM(''||column_name||''),2) square_sum_''||rn inner_calculation
    FROM
      (
      SELECT
        row_number() OVER (order by a.attnum) rn
```

```
        ,a.attname::VARCHAR column_name
        ,a.atttypid
      FROM pg_namespace AS n
        INNER JOIN pg_class AS c ON n.oid = c.relnamespace
        INNER JOIN pg_attribute AS a ON c.oid = a.attrelid
      WHERE a.attnum > 0
        AND n.nspname = '''||v_source_schema_name||'''
        AND c.relname = '''||v_source_table_name||'''
        AND a.atttypid IN (16,20,21,23,700,701,1700)
      )
    )
)';
  EXECUTE v_sql INTO v_generated_sql;
  EXECUTE  v_generated_sql;
END;
$$ LANGUAGE plpgsql;
```

A consulta a seguir mostra a correlação entre a coluna de destino, `trip_count`, e outros atributos numéricos em nosso conjunto de dados.

```
call sp_correlation(
    'public',
    'trip_data',
    'trip_count',
    'tmp_corr_table'
);

SELECT
    *
FROM
    tmp_corr_table;
```

Veja a seguir um exemplo de saída da operação anterior `sp_correlation`.

```
+------------+-----------+----------+------------+-----------+--------------
+-----------------+---------------+--------+------------------+------------
+---------------+
| trip_count | trip_hour | trip_day | trip_month | trip_year | trip_quarter
 | trip_month_week | trip_week_day | temp_c | precip_amount_mm | is_holiday |
 serial_number |
+------------+-----------+----------+------------+-----------+--------------
+-----------------+---------------+--------+------------------+------------
+---------------+
```

```
|          1 |      0.32 |     0.01 |       0.18 |      0.12 |         0.18 |
     0 |          0.02 |   0.53 |            -0.07 |      -0.13 |             0 |
+------------+-----------+----------+------------+-----------+--------------
+-----------------+---------------+--------+-----------------+------------
+---------------+
```

Etapa 2: Criar o modelo de machine learning

1. A consulta a seguir divide os dados em um conjunto de treinamento e um conjunto de validação, designando 80% do conjunto de dados para treinamento e 20% para validação. O conjunto de treinamento é a entrada para o modelo de ML que visa identificar o melhor algoritmo possível para o modelo. Depois que o modelo é criado, o conjunto de validação é usado para validar a precisão do modelo.

```
CREATE TABLE training_data AS
SELECT
    trip_count,
    trip_hour,
    trip_day,
    trip_month,
    trip_year,
    trip_quarter,
    trip_month_week,
    trip_week_day,
    temp_c,
    precip_amount_mm,
    is_holiday
FROM
    trip_data
WHERE
    serial_number > (
        SELECT
            COUNT(1) * 0.2
        FROM
            trip_data
    );

CREATE TABLE validation_data AS
SELECT
    trip_count,
    trip_hour,
    trip_day,
```

```
    trip_month,
    trip_year,
    trip_quarter,
    trip_month_week,
    trip_week_day,
    temp_c,
    precip_amount_mm,
    is_holiday,
    trip_time
FROM
    trip_data
WHERE
    serial_number <= (
        SELECT
            COUNT(1) * 0.2
        FROM
            trip_data
    );
```

2. A consulta a seguir cria um modelo de regressão para prever o valor `trip_count` referente a qualquer data e hora de entrada. No exemplo a seguir, substitua amzn-s3-demo-bucket por seu próprio bucket do S3.

```
CREATE MODEL predict_rental_count
FROM
    training_data TARGET trip_count FUNCTION predict_rental_count
    IAM_ROLE default
    PROBLEM_TYPE regression
    OBJECTIVE 'mse'
    SETTINGS (
        s3_bucket 'amzn-s3-demo-bucket',
        s3_garbage_collect off,
        max_runtime 5000
    );
```

Etapa 3: Validar o modelo

1. Use a consulta a seguir para gerar aspectos do modelo e encontrar a métrica de erro quadrático médio na saída. O erro quadrático médio é uma métrica de precisão típica para problemas de regressão.

```
show model predict_rental_count;
```

2. Execute as consultas de previsão a seguir em relação aos dados de validação para comparar a contagem prevista com a contagem real de viagens.

```
SELECT
    trip_time,
    actual_count,
    predicted_count,
    (actual_count - predicted_count) difference
FROM
    (
        SELECT
            trip_time,
            trip_count AS actual_count,
            PREDICT_RENTAL_COUNT (
                trip_hour,
                trip_day,
                trip_month,
                trip_year,
                trip_quarter,
                trip_month_week,
                trip_week_day,
                temp_c,
                precip_amount_mm,
                is_holiday
            ) predicted_count
        FROM
            validation_data
    )
LIMIT
    5;
```

3. A consulta a seguir calcula o erro quadrático médio e o erro quadrático médio da raiz com base nos dados de validação. O erro quadrático médio e o erro quadrático médio da raiz são usados para medir a distância entre o destino numérico previsto e a resposta numérica real. Um bom modelo tem uma pontuação baixa em ambas as métricas. A consulta a seguir retorna o valor de ambas as métricas.

```
SELECT
    ROUND(
```

```
        AVG(POWER((actual_count - predicted_count), 2)),
        2
    ) mse,
    ROUND(
        SQRT(AVG(POWER((actual_count - predicted_count), 2))),
        2
    ) rmse
FROM
    (
        SELECT
            trip_time,
            trip_count AS actual_count,
            PREDICT_RENTAL_COUNT (
                trip_hour,
                trip_day,
                trip_month,
                trip_year,
                trip_quarter,
                trip_month_week,
                trip_week_day,
                temp_c,
                precip_amount_mm,
                is_holiday
            ) predicted_count
        FROM
            validation_data
    );
```

4. A consulta a seguir calcula o erro percentual na contagem de viagens para cada tempo de viagem em 1.º/1/2017. A consulta ordena os tempos de viagem desde o momento com o menor erro percentual até o tempo com o erro percentual mais alto.

```
SELECT
    trip_time,
    CAST(ABS(((actual_count - predicted_count) / actual_count)) * 100 AS DECIMAL
 (7,2)) AS pct_error
FROM
    (
        SELECT
            trip_time,
            trip_count AS actual_count,
            PREDICT_RENTAL_COUNT (
                trip_hour,
```

```
                trip_day,
                trip_month,
                trip_year,
                trip_quarter,
                trip_month_week,
                trip_week_day,
                temp_c,
                precip_amount_mm,
                is_holiday
            ) predicted_count
        FROM
            validation_data
    )
WHERE
   trip_time LIKE '2017-01-01 %%:%%:%%'
ORDER BY
   2 ASC;
```

Tópicos relacionados

Para obter mais informações sobre o Amazon Redshift ML, consulte a seguinte documentação:

- Custos para usar o ML do Amazon Redshift
- Operação CREATE MODEL
- Função EXPLAIN_MODEL

Para obter mais informações sobre machine learning, consulte a seguinte documentação:

- Visão geral do Machine Learning
- Machine Learning para iniciantes e especialistas
- What Is Fairness and Model Explainability for Machine Learning Predictions? (O que é equidade e explicabilidade de modelo para previsões de machine learning?

Tutorial: Como construir modelos de regressão com o aprendizado linear

Neste tutorial, você cria um modelo de aprendizado linear com dados do Amazon S3 e executa consultas de previsão com o modelo usando o Amazon Redshift ML. O algoritmo de aprendizagem linear do SageMaker AI resolve problemas de regressão ou de classificação multiclasse. Para saber

mais sobre problemas de regressão e classificação multiclasse, consulte Tipos de problemas para os paradigmas básicos de machine learning no Guia do desenvolvedor do Amazon SageMaker AI. Neste tutorial, você resolve um problema de regressão. O algoritmo de aprendizado linear treina muitos modelos em paralelo e determina automaticamente o modelo mais otimizado. Use a operação CREATE MODEL no Amazon Redshift, que cria o modelo de aprendizado linear usando o SageMaker AI e envia uma função de previsão ao Amazon Redshift. Para obter mais informações sobre o algoritmo de aprendizagem linear, consulte Algoritmo de Aprendizagem linear no Guia do desenvolvedor do Amazon SageMaker AI.

Você pode usar um comando CREATE MODEL para exportar dados de treinamento, treinar um modelo, importar o modelo e preparar uma função de previsão do Amazon Redshift. Use a operação CREATE MODEL para especificar dados de treinamento como uma tabela ou instrução SELECT.

Os modelos de aprendizado linear otimizam objetivos contínuos ou objetivos discretos. Os objetivos contínuos são usados para regressão, enquanto as variáveis discretas são usadas para classificação. Alguns métodos fornecem uma solução apenas para objetivos contínuos, como o de regressão. O algoritmo de aprendizado linear possibilita um aumento de velocidade em relação às técnicas de otimização de hiperparâmetros nativas, como a técnica Naive Bayes. Uma técnica de otimização simples pressupõe que cada variável de entrada seja independente. Para usar o algoritmo de aprendizado linear, você deve fornecer colunas para representar as dimensões das entradas e linhas para representar as observações. Para obter mais informações sobre o algoritmo de aprendizagem linear, consulte Algoritmo de Aprendizagem linear no Guia do desenvolvedor do Amazon SageMaker AI.

Neste tutorial, você constrói um modelo de aprendizado linear que prevê a idade de abalones. Você usa o comando CREATE MODEL no conjunto de dados Abalone para determinar a relação entre as medidas físicas do abalone. Em seguida, você usa o modelo para determinar a idade de abalones.

Exemplos de casos de uso

Você pode resolver outros problemas de regressão com o aprendizado linear e o Amazon Redshift ML, como prever o preço de uma casa. Você também pode usar o Redshift ML para prever o número de pessoas que usarão o serviço de aluguel de bicicletas em uma cidade.

Tarefas

- Pré-requisitos
- Etapa 1: Carregar dados do Amazon S3 para o Amazon Redshift

- Etapa 2: Criar o modelo de machine learning
- Etapa 3: Validar o modelo

Pré-requisitos

Para finalizar este tutorial, você deve concluir a configuração administrativa do Amazon Redshift ML.

Etapa 1: Carregar dados do Amazon S3 para o Amazon Redshift

Use o editor de consultas v2 do Amazon Redshift para executar as consultas a seguir. Essas consultas carregam os dados de exemplo no Redshift e os dividem em um conjunto de treinamento e um conjunto de validação.

1. A consulta a seguir cria a tabela `abalone_dataset`.

```
CREATE TABLE abalone_dataset (
    id INT IDENTITY(1, 1),
    Sex CHAR(1),
    Length float,
    Diameter float,
    Height float,
    Whole float,
    Shucked float,
    Viscera float,
    Shell float,
    Rings integer
);
```

2. A consulta a seguir copia os dados de exemplo do conjunto de dados Abalone no Amazon S3 para a tabela `abalone_dataset` que você criou anteriormente no Amazon Redshift.

```
COPY abalone_dataset
FROM
    's3://redshift-ml-multiclass/abalone.csv' REGION 'us-east-1' IAM_ROLE default CSV
 IGNOREHEADER 1 NULL AS 'NULL';
```

3. Ao dividir manualmente os dados, você poderá verificar a precisão do modelo alocando um conjunto adicional de previsões. A consulta a seguir divide os dados em dois conjuntos. A tabela `abalone_training` destina-se ao treinamento e a tabela `abalone_validation`, à validação.

```
CREATE TABLE abalone_training as
```

```
SELECT
    *
FROM
    abalone_dataset
WHERE
    mod(id, 10) < 8;

CREATE TABLE abalone_validation as
SELECT
    *
FROM
    abalone_dataset
WHERE
    mod(id, 10) >= 8;
```

Etapa 2: Criar o modelo de machine learning

Nesta etapa, você usa a instrução CREATE MODEL para criar o modelo de machine learning com o algoritmo de aprendizado linear.

A consulta a seguir cria o modelo de aprendizado linear com a operação CREATE MODEL usando seu bucket do S3. Substitua amzn-s3-demo-bucket por seu próprio bucket do S3.

```
CREATE MODEL model_abalone_ring_prediction
FROM
    (
        SELECT
            Sex,
            Length,
            Diameter,
            Height,
            Whole,
            Shucked,
            Viscera,
            Shell,
            Rings AS target_label
        FROM
            abalone_training
    ) TARGET target_label FUNCTION f_abalone_ring_prediction IAM_ROLE default
 MODEL_TYPE LINEAR_LEARNER PROBLEM_TYPE REGRESSION OBJECTIVE 'MSE' SETTINGS (
        S3_BUCKET 'amzn-s3-demo-bucket',
        MAX_RUNTIME 15000
```

```
    );
```

Mostrar o status do modelo de treinamento (opcional)

Você pode usar o comando SHOW MODEL para saber quando o modelo está pronto.

Use a consulta a seguir para monitorar o andamento do treinamento do modelo.

```
SHOW MODEL model_abalone_ring_prediction;
```

Quando o modelo estiver pronto, a saída da operação anterior será semelhante à saída do exemplo a seguir. Observe que a saída fornece a métrica `validation:mse`, que é o erro quadrático médio. Você usará o erro quadrático médio para validar a precisão do modelo na próxima etapa.

```
+--------------------------
+-----------------------------------------------------------------------------------------------
+
|        Model Name        |
 model_abalone_ring_prediction                                    |
+--------------------------
+-----------------------------------------------------------------------------------------------
+
| Schema Name              | public
                                        |
| Owner                    | awsuser
                                        |
| Creation Time            | Thu, 30.06.2022 18:00:10
                                        |
| Model State              | READY
                                        |
| validation:mse           |
                                 4.168633 |
| Estimated Cost           |
                                 4.291608 |
|                          |
                                        |
| TRAINING DATA:           |
                                        |
| Query                    | SELECT SEX , LENGTH , DIAMETER , HEIGHT , WHOLE ,
 SHUCKED , VISCERA , SHELL, RINGS AS TARGET_LABEL |
|                          | FROM ABALONE_TRAINING
                                        |
```

```
| Target Column             | TARGET_LABEL
                                        |
|                           |
                                        |
| PARAMETERS:               |
                                        |
| Model Type                | linear_learner
                                        |
| Problem Type              | Regression
                                        |
| Objective                 | MSE
                                        |
| AutoML Job Name           | redshiftml-20220630180010947843
                                        |
| Function Name             | f_abalone_ring_prediction
                                        |
| Function Parameters       | sex length diameter height whole shucked viscera shell
                                        |
| Function Parameter Types | bpchar float8 float8 float8 float8 float8 float8 float8
                                        |
| IAM Role                  | default-aws-iam-role
                                        |
| S3 Bucket                 | amzn-s3-demo-bucket
                                        |
| Max Runtime               |
                                  15000 |
+--------------------------
+-----------------------------------------------------------------------------------------------
+
```

Etapa 3: Validar o modelo

1. A consulta de previsão a seguir valida a precisão do modelo no conjunto de dados `abalone_validation` calculando o erro quadrático médio e o erro quadrático médio da raiz.

```
SELECT
    ROUND(AVG(POWER((tgt_label - predicted), 2)), 2) mse,
    ROUND(SQRT(AVG(POWER((tgt_label - predicted), 2))), 2) rmse
FROM
    (
        SELECT
            Sex,
            Length,
```

```
            Diameter,
            Height,
            Whole,
            Shucked,
            Viscera,
            Shell,
            Rings AS tgt_label,
            f_abalone_ring_prediction(
                Sex,
                Length,
                Diameter,
                Height,
                Whole,
                Shucked,
                Viscera,
                Shell
            ) AS predicted,
            CASE
                WHEN tgt_label = predicted then 1
                ELSE 0
            END AS match,
            CASE
                WHEN tgt_label <> predicted then 1
                ELSE 0
            END AS nonmatch
        FROM
            abalone_validation
    ) t1;
```

A saída do comando será semelhante à do exemplo a seguir. O valor da métrica de erro quadrático médio deve ser semelhante ao da métrica `validation:mse` mostrada pela saída da operação SHOW MODEL.

```
+-----+--------------------+
| mse |        rmse        |
+-----+--------------------+
| 5.1 | 2.2600000000000002 |
+-----+--------------------+
```

2. Use a consulta a seguir para executar a operação EXPLAIN_MODEL na função de previsão. A operação retornará um relatório de explicabilidade do modelo. Para obter mais informações sobre

a operação EXPLAIN_MODEL, consulte Função EXPLAIN_MODEL no Guia do desenvolvedor de bancos de dados do Amazon Redshift.

```
SELECT
    EXPLAIN_MODEL ('model_abalone_ring_prediction');
```

As informações a seguir são um exemplo de relatório de explicabilidade do modelo produzido pela operação EXPLAIN_MODEL anterior. Os valores para cada uma das entradas são Shapley. Os valores Shapley representam o efeito que cada entrada tem na previsão do modelo. As entradas de valor mais alto têm mais impacto na previsão. Neste exemplo, as entradas de valor mais alto têm maior impacto na previsão de idade de abalones.

```
{
    "explanations": {
        "kernel_shap": {
            "label0": {
                "expected_value" :10.290688514709473,
                "global_shap_values": {
                    "diameter" :0.6856910187882492,
                    "height" :0.4415323937124035,
                    "length" :0.21507476107609084,
                    "sex" :0.448611774505744,
                    "shell" :1.70426496893776,
                    "shucked" :2.1181392924386994,
                    "viscera" :0.342220754059912,
                    "whole" :0.6711906974084011
                }
            }
        }
    },
    "version" :"1.0"
};
```

3. Use a consulta a seguir para calcular a porcentagem de previsões corretas realizadas pelo modelo sobre abalones que ainda não são adultos. Os abalones ainda prematuros têm dez anéis ou menos, e uma previsão correta do número real de anéis é um anel.

```
SELECT
    TRUNC(
        SUM(
            CASE
```

```
                WHEN ROUND(
                    f_abalone_ring_prediction(
                        Sex,
                        Length,
                        Diameter,
                        Height,
                        Whole,
                        Shucked,
                        Viscera,
                        Shell
                    ),
                    0
                ) BETWEEN Rings - 1
                AND Rings + 1 THEN 1
                ELSE 0
            END
        ) / CAST(COUNT(SHELL) AS FLOAT),
        4
    ) AS prediction_pct
FROM
    abalone_validation
WHERE
    Rings <= 10;
```

Tópicos relacionados

Para obter mais informações sobre o Amazon Redshift ML, consulte a seguinte documentação:

- Custos para usar o ML do Amazon Redshift
- Operação CREATE MODEL
- Função EXPLAIN_MODEL

Para obter mais informações sobre machine learning, consulte a seguinte documentação:

- Visão geral do Machine Learning
- Machine Learning para iniciantes e especialistas
- What Is Fairness and Model Explainability for Machine Learning Predictions? (O que é equidade e explicabilidade de modelo para previsões de machine learning?

Tutorial: Como criar modelos de classificação com aprendizado linear

Neste tutorial, você cria um modelo de aprendizado linear com dados do Amazon S3 e executa consultas de previsão com o modelo usando o Amazon Redshift ML. O algoritmo de aprendizagem linear do SageMaker AI resolve problemas de regressão ou classificação. Para saber mais sobre problemas de regressão e classificação multiclasse, consulte Tipos de problemas para os paradigmas básicos de machine learning no Guia do desenvolvedor do Amazon SageMaker AI. Neste tutorial, você resolve um problema de classificação multiclasse. O algoritmo de aprendizagem linear treina muitos modelos em paralelo e determina automaticamente o modelo mais otimizado. Use a operação CREATE MODEL no Amazon Redshift, que cria o modelo de aprendizagem linear usando o SageMaker AI e envia a função de previsão para o Amazon Redshift. Para obter mais informações sobre o algoritmo de aprendizagem linear, consulte Algoritmo de Aprendizagem linear no Guia do desenvolvedor do Amazon SageMaker AI.

Você pode usar um comando CREATE MODEL para exportar dados de treinamento, treinar um modelo, importar o modelo e preparar uma função de previsão do Amazon Redshift. Use a operação CREATE MODEL para especificar dados de treinamento como uma tabela ou instrução SELECT.

Os modelos de aprendizado linear otimizam objetivos contínuos ou objetivos discretos. Os objetivos contínuos são usados para regressão, enquanto as variáveis discretas são usadas para classificação. Alguns métodos fornecem uma solução apenas para objetivos contínuos, como o de regressão. O algoritmo de aprendizado linear possibilita um aumento de velocidade em relação às técnicas de otimização de hiperparâmetros nativas, como a técnica Naive Bayes. Uma técnica de otimização simples pressupõe que cada variável de entrada seja independente. O algoritmo de aprendizado linear treina muitos modelos em paralelo e seleciona o modelo mais otimizado. Um algoritmo semelhante é o XGBoost, que combina estimativas de um conjunto de modelos mais simples e mais fracos para fazer previsões. Para saber mais sobre o XGBoost, consulte Algoritmo XGBoost no Guia do desenvolvedor do Amazon SageMaker AI.

Para usar o algoritmo de aprendizado linear, você deve fornecer colunas para representar as dimensões das entradas e linhas para representar as observações. Para obter mais informações sobre o algoritmo de aprendizagem linear, consulte Algoritmo de Aprendizagem linear no Guia do desenvolvedor do Amazon SageMaker AI.

Neste tutorial, você cria um modelo de aprendizado linear que prevê os tipos de cobertura florestal de determinada área. Use o comando CREATE MODEL no conjunto de dados Covertype do UCI Machine Learning Repository. Em seguida, você usa a função de previsão criada pelo comando para determinar os tipos de cobertura de uma área selvagem. Um tipo de cobertura florestal geralmente é

um tipo de árvore. As entradas que o Redshift ML usará para criar o modelo incluem o tipo de solo, a distância das estradas e a designação da área selvagem. Para obter mais informações sobre o conjunto de dados, consulte Covertype Dataset (Conjunto de dados sobre tipo de cobertura) no UCI Machine Learning Repository.

Exemplos de casos de uso

Você pode resolver outros problemas de classificação multiclasse com aprendizado linear por meio do Amazon Redshift ML, como prever a espécie de uma planta com base em uma imagem. Você também pode prever a quantidade de um produto que um cliente comprará.

Tarefas

- Pré-requisitos
- Etapa 1: Carregar dados do Amazon S3 para o Amazon Redshift
- Etapa 2: Criar o modelo de machine learning
- Etapa 3: Validar o modelo

Pré-requisitos

Para finalizar este tutorial, você deve concluir a configuração administrativa do Amazon Redshift ML.

Etapa 1: Carregar dados do Amazon S3 para o Amazon Redshift

Use o editor de consultas v2 do Amazon Redshift para executar as consultas a seguir. Essas consultas carregam os dados de exemplo no Redshift e os dividem em um conjunto de treinamento e um conjunto de validação.

1. A consulta a seguir cria a tabela `covertype_data`.

```
CREATE TABLE public.covertype_data (
    elevation bigint ENCODE az64,
    aspect bigint ENCODE az64,
    slope bigint ENCODE az64,
    horizontal_distance_to_hydrology bigint ENCODE az64,
    vertical_distance_to_hydrology bigint ENCODE az64,
    horizontal_distance_to_roadways bigint ENCODE az64,
    hillshade_9am bigint ENCODE az64,
    hillshade_noon bigint ENCODE az64,
    hillshade_3pm bigint ENCODE az64,
```

```
    horizontal_distance_to_fire_points bigint ENCODE az64,
    wilderness_area1 bigint ENCODE az64,
    wilderness_area2 bigint ENCODE az64,
    wilderness_area3 bigint ENCODE az64,
    wilderness_area4 bigint ENCODE az64,
    soil_type1 bigint ENCODE az64,
    soil_type2 bigint ENCODE az64,
    soil_type3 bigint ENCODE az64,
    soil_type4 bigint ENCODE az64,
    soil_type5 bigint ENCODE az64,
    soil_type6 bigint ENCODE az64,
    soil_type7 bigint ENCODE az64,
    soil_type8 bigint ENCODE az64,
    soil_type9 bigint ENCODE az64,
    soil_type10 bigint ENCODE az64,
    soil_type11 bigint ENCODE az64,
    soil_type12 bigint ENCODE az64,
    soil_type13 bigint ENCODE az64,
    soil_type14 bigint ENCODE az64,
    soil_type15 bigint ENCODE az64,
    soil_type16 bigint ENCODE az64,
    soil_type17 bigint ENCODE az64,
    soil_type18 bigint ENCODE az64,
    soil_type19 bigint ENCODE az64,
    soil_type20 bigint ENCODE az64,
    soil_type21 bigint ENCODE az64,
    soil_type22 bigint ENCODE az64,
    soil_type23 bigint ENCODE az64,
    soil_type24 bigint ENCODE az64,
    soil_type25 bigint ENCODE az64,
    soil_type26 bigint ENCODE az64,
    soil_type27 bigint ENCODE az64,
    soil_type28 bigint ENCODE az64,
    soil_type29 bigint ENCODE az64,
    soil_type30 bigint ENCODE az64,
    soil_type31 bigint ENCODE az64,
    soil_type32 bigint ENCODE az64,
    soil_type33 bigint ENCODE az64,
    soil_type34 bigint ENCODE az64,
    soil_type35 bigint ENCODE az64,
    soil_type36 bigint ENCODE az64,
    soil_type37 bigint ENCODE az64,
    soil_type38 bigint ENCODE az64,
    soil_type39 bigint ENCODE az64,
```

```
    soil_type40 bigint ENCODE az64,
    cover_type bigint ENCODE az64
) DISTSTYLE AUTO;
```

2. A consulta a seguir copia os dados de exemplo do conjunto de dados Covertype (Tipo de cobertura) no Amazon S3 para a tabela `covertype_data` que você criou anteriormente no Amazon Redshift.

```
COPY public.covertype_data
FROM
    's3://redshift-ml-multiclass/covtype.data.gz' IAM_ROLE DEFAULT gzip DELIMITER ','
 REGION 'us-east-1';
```

3. Ao dividir manualmente os dados nesta etapa, você poderá verificar a precisão do modelo alocando um conjunto adicional de testes. A consulta a seguir divide os dados em três conjuntos. Você usará a tabela `covertype_training` para treinamento, a tabela `covertype_validation` para validação e a tabela `covertype_test` para testar o modelo. Você usará o conjunto de treinamento para treinar o modelo e o conjunto de validação para validar o desenvolvimento do modelo. Em seguida, você usará o conjunto de testes para testar a performance do modelo e ver se o modelo está superajustando ou subajustando o conjunto de dados.

```
CREATE TABLE public.covertype_data_prep AS
SELECT
    a.*,
    CAST (random() * 100 AS int) AS data_group_id
FROM
    public.covertype_data a;

--training dataset
 CREATE TABLE public.covertype_training as
SELECT
    *
FROM
    public.covertype_data_prep
WHERE
    data_group_id < 80;

--validation dataset
 CREATE TABLE public.covertype_validation AS
SELECT
    *
```

```
FROM
    public.covertype_data_prep
WHERE
    data_group_id BETWEEN 80
    AND 89;

--test dataset
 CREATE TABLE public.covertype_test AS
SELECT
    *
FROM
    public.covertype_data_prep
WHERE
    data_group_id > 89;
```

Etapa 2: Criar o modelo de machine learning

Nesta etapa, você usa a instrução CREATE MODEL para criar o modelo de machine learning com o algoritmo de aprendizado linear.

A consulta a seguir cria o modelo de aprendizado linear com a operação CREATE MODEL usando seu bucket do S3. Substitua amzn-s3-demo-bucket por seu próprio bucket do S3.

```
CREATE MODEL forest_cover_type_model
FROM
    (
        SELECT
            Elevation,
            Aspect,
            Slope,
            Horizontal_distance_to_hydrology,
            Vertical_distance_to_hydrology,
            Horizontal_distance_to_roadways,
            HIllshade_9am,
            Hillshade_noon,
            Hillshade_3pm,
            Horizontal_Distance_To_Fire_Points,
            Wilderness_Area1,
            Wilderness_Area2,
            Wilderness_Area3,
            Wilderness_Area4,
            soil_type1,
```

```
            Soil_Type2,
            Soil_Type3,
            Soil_Type4,
            Soil_Type5,
            Soil_Type6,
            Soil_Type7,
            Soil_Type8,
            Soil_Type9,
            Soil_Type10,
            Soil_Type11,
            Soil_Type12,
            Soil_Type13,
            Soil_Type14,
            Soil_Type15,
            Soil_Type16,
            Soil_Type17,
            Soil_Type18,
            Soil_Type19,
            Soil_Type20,
            Soil_Type21,
            Soil_Type22,
            Soil_Type23,
            Soil_Type24,
            Soil_Type25,
            Soil_Type26,
            Soil_Type27,
            Soil_Type28,
            Soil_Type29,
            Soil_Type30,
            Soil_Type31,
            Soil_Type32,
            Soil_Type33,
            Soil_Type34,
            Soil_Type36,
            Soil_Type37,
            Soil_Type38,
            Soil_Type39,
            Soil_Type40,
            Cover_type
        from
            public.covertype_training
    ) TARGET cover_type FUNCTION predict_cover_type IAM_ROLE default MODEL_TYPE
LINEAR_LEARNER PROBLEM_TYPE MULTICLASS_CLASSIFICATION OBJECTIVE 'Accuracy' SETTINGS (
        S3_BUCKET 'amzn-s3-demo-bucket',
```

```
        S3_GARBAGE_COLLECT OFF,
        MAX_RUNTIME 15000
    );
```

Mostrar o status do modelo de treinamento (opcional)

Você pode usar o comando SHOW MODEL para saber quando o modelo está pronto.

Use a consulta a seguir para monitorar o andamento do treinamento do modelo.

```
SHOW MODEL forest_cover_type_model;
```

Quando o modelo estiver pronto, a saída da operação anterior será semelhante à saída do exemplo a seguir. Observe que a saída fornece a métrica `validation:multiclass_accuracy`, que você pode ver no lado direito do exemplo a seguir. A precisão multiclasse mede a porcentagem de pontos de dados que são classificados corretamente pelo modelo. Você usará a precisão multiclasse para validar a exatidão do modelo na próxima etapa.

```
+--------------------------------
+-------------------------------------------------------------------------------------------------
+
|              Key               |




 Value




                                                          |
+--------------------------------
+-------------------------------------------------------------------------------------------------
+
| Model Name                     | forest_cover_type_model
```

```
                                                    |
| Schema Name                    | public
                                                    |
| Owner                          | awsuser
                                                    |
| Creation Time                  | Tue, 12.07.2022 20:24:32
                                                    |
| Model State                    | READY
                                                    |
| validation:multiclass_accuracy |
```

```
                                        0.724952 |
| Estimated Cost                 |

                                        5.341750 |
|                                |

                                                 |
| TRAINING DATA:                 |

                                                 |
| Query                          | SELECT ELEVATION, ASPECT, SLOPE,
 HORIZONTAL_DISTANCE_TO_HYDROLOGY, VERTICAL_DISTANCE_TO_HYDROLOGY,
 HORIZONTAL_DISTANCE_TO_ROADWAYS, HILLSHADE_9AM, HILLSHADE_NOON, HILLSHADE_3PM ,
 HORIZONTAL_DISTANCE_TO_FIRE_POINTS, WILDERNESS_AREA1, WILDERNESS_AREA2,
 WILDERNESS_AREA3, WILDERNESS_AREA4, SOIL_TYPE1, SOIL_TYPE2, SOIL_TYPE3, SOIL_TYPE4,
 SOIL_TYPE5, SOIL_TYPE6, SOIL_TYPE7, SOIL_TYPE8, SOIL_TYPE9, SOIL_TYPE10 , SOIL_TYPE11,
```

```
 SOIL_TYPE12 , SOIL_TYPE13 , SOIL_TYPE14, SOIL_TYPE15, SOIL_TYPE16, SOIL_TYPE17,
 SOIL_TYPE18, SOIL_TYPE19, SOIL_TYPE20, SOIL_TYPE21, SOIL_TYPE22, SOIL_TYPE23,
 SOIL_TYPE24, SOIL_TYPE25, SOIL_TYPE26, SOIL_TYPE27, SOIL_TYPE28, SOIL_TYPE29,
 SOIL_TYPE30, SOIL_TYPE31, SOIL_TYPE32, SOIL_TYPE33, SOIL_TYPE34, SOIL_TYPE36,
 SOIL_TYPE37, SOIL_TYPE38, SOIL_TYPE39, SOIL_TYPE40, COVER_TYPE |
|                              | FROM PUBLIC.COVERTYPE_TRAINING

                                                                |
| Target Column                | COVER_TYPE

                                                                |
|                              |

                                                                |
| PARAMETERS:                  |
```

```
                                                    |
| Model Type                    | linear_learner

                                                    |
| Problem Type                  | MulticlassClassification

                                                    |
| Objective                     | Accuracy

                                                    |
| AutoML Job Name               | redshiftml-20220712202432187659

                                                    |
| Function Name                 | predict_cover_type
```

```
                                                  |
| Function Parameters            | elevation aspect slope
 horizontal_distance_to_hydrology vertical_distance_to_hydrology
 horizontal_distance_to_roadways hillshade_9am hillshade_noon hillshade_3pm
 horizontal_distance_to_fire_points wilderness_area1 wilderness_area2 wilderness_area3
 wilderness_area4 soil_type1 soil_type2 soil_type3 soil_type4 soil_type5 soil_type6
 soil_type7 soil_type8 soil_type9 soil_type10 soil_type11 soil_type12 soil_type13
 soil_type14 soil_type15 soil_type16 soil_type17 soil_type18 soil_type19 soil_type20
 soil_type21 soil_type22 soil_type23 soil_type24 soil_type25 soil_type26 soil_type27
 soil_type28 soil_type29 soil_type30 soil_type31 soil_type32 soil_type33 soil_type34
 soil_type36 soil_type37 soil_type38 soil_type39 soil_type40
                                                 |
| Function Parameter Types       | int8 int8 int8 int8 int8 int8 int8 int8 int8 int8
 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8
 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8
 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8
 int8 int8 int8 int8 int8 int8 int8 int8 int8

                                                        |
| IAM Role                       | default-aws-iam-role

                                                  |
| S3 Bucket                      | amzn-s3-demo-bucket
```

```
                                                    |
| Max Runtime                    |

                                                15000 |
+--------------------------------
+-----------------------------------------------------------------------------------------------
+
```

Etapa 3: Validar o modelo

1. A consulta de previsão a seguir valida a precisão do modelo no conjunto de dados `covertype_validation` calculando a precisão multiclasse. A precisão multiclasse é a porcentagem das previsões do modelo que estão corretas.

```
SELECT
    CAST(sum(t1.match) AS decimal(7, 2)) AS predicted_matches,
    CAST(sum(t1.nonmatch) AS decimal(7, 2)) AS predicted_non_matches,
    CAST(sum(t1.match + t1.nonmatch) AS decimal(7, 2)) AS total_predictions,
    predicted_matches / total_predictions AS pct_accuracy
FROM
    (
        SELECT
            Elevation,
            Aspect,
            Slope,
            Horizontal_distance_to_hydrology,
            Vertical_distance_to_hydrology,
            Horizontal_distance_to_roadways,
            HIllshade_9am,
            Hillshade_noon,
            Hillshade_3pm,
            Horizontal_Distance_To_Fire_Points,
```

```
            Wilderness_Area1,
            Wilderness_Area2,
            Wilderness_Area3,
            Wilderness_Area4,
            soil_type1,
            Soil_Type2,
            Soil_Type3,
            Soil_Type4,
            Soil_Type5,
            Soil_Type6,
            Soil_Type7,
            Soil_Type8,
            Soil_Type9,
            Soil_Type10,
            Soil_Type11,
            Soil_Type12,
            Soil_Type13,
            Soil_Type14,
            Soil_Type15,
            Soil_Type16,
            Soil_Type17,
            Soil_Type18,
            Soil_Type19,
            Soil_Type20,
            Soil_Type21,
            Soil_Type22,
            Soil_Type23,
            Soil_Type24,
            Soil_Type25,
            Soil_Type26,
            Soil_Type27,
            Soil_Type28,
            Soil_Type29,
            Soil_Type30,
            Soil_Type31,
            Soil_Type32,
            Soil_Type33,
            Soil_Type34,
            Soil_Type36,
            Soil_Type37,
            Soil_Type38,
            Soil_Type39,
            Soil_Type40,
            Cover_type AS actual_cover_type,
```

```
            predict_cover_type(
                Elevation,
                Aspect,
                Slope,
                Horizontal_distance_to_hydrology,
                Vertical_distance_to_hydrology,
                Horizontal_distance_to_roadways,
                HIllshade_9am,
                Hillshade_noon,
                Hillshade_3pm,
                Horizontal_Distance_To_Fire_Points,
                Wilderness_Area1,
                Wilderness_Area2,
                Wilderness_Area3,
                Wilderness_Area4,
                soil_type1,
                Soil_Type2,
                Soil_Type3,
                Soil_Type4,
                Soil_Type5,
                Soil_Type6,
                Soil_Type7,
                Soil_Type8,
                Soil_Type9,
                Soil_Type10,
                Soil_Type11,
                Soil_Type12,
                Soil_Type13,
                Soil_Type14,
                Soil_Type15,
                Soil_Type16,
                Soil_Type17,
                Soil_Type18,
                Soil_Type19,
                Soil_Type20,
                Soil_Type21,
                Soil_Type22,
                Soil_Type23,
                Soil_Type24,
                Soil_Type25,
                Soil_Type26,
                Soil_Type27,
                Soil_Type28,
                Soil_Type29,
```

```
                Soil_Type30,
                Soil_Type31,
                Soil_Type32,
                Soil_Type33,
                Soil_Type34,
                Soil_Type36,
                Soil_Type37,
                Soil_Type38,
                Soil_Type39,
                Soil_Type40
            ) AS predicted_cover_type,
            CASE
                WHEN actual_cover_type = predicted_cover_type THEN 1
                ELSE 0
            END AS match,
            CASE
                WHEN actual_cover_type <> predicted_cover_type THEN 1
                ELSE 0
            END AS nonmatch
        FROM
            public.covertype_validation
    ) t1;
```

A saída do comando será semelhante à do exemplo a seguir. O valor da métrica de precisão multiclasse deve ser semelhante ao da métrica `validation:multiclass_accuracy` mostrada pela saída da operação SHOW MODEL.

```
+------------------+----------------------+------------------+--------------+
| predicted_matches | predicted_non_matches | total_predictions | pct_accuracy |
+------------------+----------------------+------------------+--------------+
|             41211 |                16324 |             57535 |   0.71627704 |
+------------------+----------------------+------------------+--------------+
```

2. A consulta a seguir prevê o tipo de cobertura mais comum para `wilderness_area2`. Esse conjunto de dados inclui quatro áreas selvagens e sete tipos de cobertura. Uma área selvagem pode ter vários tipos de cobertura.

```
SELECT t1. predicted_cover_type, COUNT(*)
FROM
(
SELECT
   Elevation,
```

```
    Aspect,
    Slope,
    Horizontal_distance_to_hydrology,
    Vertical_distance_to_hydrology,
    Horizontal_distance_to_roadways,
    HIllshade_9am,
    Hillshade_noon,
    Hillshade_3pm ,
    Horizontal_Distance_To_Fire_Points,
    Wilderness_Area1,
    Wilderness_Area2,
    Wilderness_Area3,
    Wilderness_Area4,
    soil_type1,
    Soil_Type2,
    Soil_Type3,
    Soil_Type4,
    Soil_Type5,
    Soil_Type6,
    Soil_Type7,
    Soil_Type8,
    Soil_Type9,
    Soil_Type10 ,
    Soil_Type11,
    Soil_Type12 ,
    Soil_Type13 ,
    Soil_Type14,
    Soil_Type15,
    Soil_Type16,
    Soil_Type17,
    Soil_Type18,
    Soil_Type19,
    Soil_Type20,
    Soil_Type21,
    Soil_Type22,
    Soil_Type23,
    Soil_Type24,
    Soil_Type25,
    Soil_Type26,
    Soil_Type27,
    Soil_Type28,
    Soil_Type29,
    Soil_Type30,
    Soil_Type31,
```

```
    Soil_Type32,
    Soil_Type33,
    Soil_Type34,
    Soil_Type36,
    Soil_Type37,
    Soil_Type38,
    Soil_Type39,
    Soil_Type40,
    predict_cover_type( Elevation,
    Aspect,
    Slope,
    Horizontal_distance_to_hydrology,
    Vertical_distance_to_hydrology,
    Horizontal_distance_to_roadways,
    HIllshade_9am,
    Hillshade_noon,
    Hillshade_3pm ,
    Horizontal_Distance_To_Fire_Points,
    Wilderness_Area1,
    Wilderness_Area2,
    Wilderness_Area3,
    Wilderness_Area4,
    soil_type1,
    Soil_Type2,
    Soil_Type3,
    Soil_Type4,
    Soil_Type5,
    Soil_Type6,
    Soil_Type7,
    Soil_Type8,
    Soil_Type9,
    Soil_Type10,
    Soil_Type11,
    Soil_Type12,
    Soil_Type13,
    Soil_Type14,
    Soil_Type15,
    Soil_Type16,
    Soil_Type17,
    Soil_Type18,
    Soil_Type19,
    Soil_Type20,
    Soil_Type21,
    Soil_Type22,
```

```
   Soil_Type23,
   Soil_Type24,
   Soil_Type25,
   Soil_Type26,
   Soil_Type27,
   Soil_Type28,
   Soil_Type29,
   Soil_Type30,
   Soil_Type31,
   Soil_Type32,
   Soil_Type33,
   Soil_Type34,
   Soil_Type36,
   Soil_Type37,
   Soil_Type38,
   Soil_Type39,
   Soil_Type40) AS predicted_cover_type

FROM public.covertype_test
WHERE wilderness_area2 = 1)
t1
GROUP BY 1;
```

A saída da operação anterior será semelhante à saída do exemplo a seguir. Essa saída significa que o modelo previu que a maioria das coberturas é do tipo 1, e há alguns tipos de cobertura 2 e 7.

```
+----------------------+-------+
| predicted_cover_type | count |
+----------------------+-------+
|                    2 |   564 |
|                    7 |    97 |
|                    1 |  2309 |
+----------------------+-------+
```

3. A consulta a seguir mostra o tipo de cobertura mais comum em uma única área selvagem. A consulta exibe a quantidade desse tipo de cobertura e a respectiva área selvagem.

```
SELECT t1. predicted_cover_type, COUNT(*), wilderness_area
FROM
(
SELECT
```

```
    Elevation,
    Aspect,
    Slope,
    Horizontal_distance_to_hydrology,
    Vertical_distance_to_hydrology,
    Horizontal_distance_to_roadways,
    HIllshade_9am,
    Hillshade_noon,
    Hillshade_3pm ,
    Horizontal_Distance_To_Fire_Points,
    Wilderness_Area1,
    Wilderness_Area2,
    Wilderness_Area3,
    Wilderness_Area4,
    soil_type1,
    Soil_Type2,
    Soil_Type3,
    Soil_Type4,
    Soil_Type5,
    Soil_Type6,
    Soil_Type7,
    Soil_Type8,
    Soil_Type9,
    Soil_Type10 ,
    Soil_Type11,
    Soil_Type12 ,
    Soil_Type13 ,
    Soil_Type14,
    Soil_Type15,
    Soil_Type16,
    Soil_Type17,
    Soil_Type18,
    Soil_Type19,
    Soil_Type20,
    Soil_Type21,
    Soil_Type22,
    Soil_Type23,
    Soil_Type24,
    Soil_Type25,
    Soil_Type26,
    Soil_Type27,
    Soil_Type28,
    Soil_Type29,
    Soil_Type30,
```

```
    Soil_Type31,
    Soil_Type32,
    Soil_Type33,
    Soil_Type34,
    Soil_Type36,
    Soil_Type37,
    Soil_Type38,
    Soil_Type39,
    Soil_Type40,
    predict_cover_type( Elevation,
    Aspect,
    Slope,
    Horizontal_distance_to_hydrology,
    Vertical_distance_to_hydrology,
    Horizontal_distance_to_roadways,
    HIllshade_9am,
    Hillshade_noon,
    Hillshade_3pm ,
    Horizontal_Distance_To_Fire_Points,
    Wilderness_Area1,
    Wilderness_Area2,
    Wilderness_Area3,
    Wilderness_Area4,
    soil_type1,
    Soil_Type2,
    Soil_Type3,
    Soil_Type4,
    Soil_Type5,
    Soil_Type6,
    Soil_Type7,
    Soil_Type8,
    Soil_Type9,
    Soil_Type10,
    Soil_Type11,
    Soil_Type12,
    Soil_Type13,
    Soil_Type14,
    Soil_Type15,
    Soil_Type16,
    Soil_Type17,
    Soil_Type18,
    Soil_Type19,
    Soil_Type20,
    Soil_Type21,
```

```
   Soil_Type22,
   Soil_Type23,
   Soil_Type24,
   Soil_Type25,
   Soil_Type26,
   Soil_Type27,
   Soil_Type28,
   Soil_Type29,
   Soil_Type30,
   Soil_Type31,
   Soil_Type32,
   Soil_Type33,
   Soil_Type34,
   Soil_Type36,
   Soil_Type37,
   Soil_Type38,
   Soil_Type39,
   Soil_Type40) AS predicted_cover_type,
   CASE WHEN Wilderness_Area1 = 1 THEN 1
        WHEN Wilderness_Area2 = 1 THEN 2
        WHEN Wilderness_Area3 = 1 THEN 3
        WHEN Wilderness_Area4 = 1 THEN 4
        ELSE 0
   END AS wilderness_area

FROM public.covertype_test)
t1
GROUP BY 1, 3
ORDER BY 2 DESC
LIMIT 1;
```

A saída da operação anterior será semelhante à saída do exemplo a seguir.

```
+----------------------+-------+-----------------+
| predicted_cover_type | count | wilderness_area |
+----------------------+-------+-----------------+
|                    2 | 15738 |               1 |
+----------------------+-------+-----------------+
```

Tópicos relacionados

Para obter mais informações sobre o Amazon Redshift ML, consulte a seguinte documentação:

- Custos para usar o ML do Amazon Redshift
- Operação CREATE MODEL
- Função EXPLAIN_MODEL

Para obter mais informações sobre machine learning, consulte a seguinte documentação:

- Visão geral do Machine Learning
- Machine Learning para iniciantes e especialistas
- What Is Fairness and Model Explainability for Machine Learning Predictions? (O que é equidade e explicabilidade de modelo para previsões de machine learning?

Integração do Amazon Redshift ML com o Amazon Bedrock

Esta seção descreve como usar a integração do Amazon Redshift ML com o Amazon Bedrock. Com esse recurso, você pode invocar um modelo do Amazon Bedrock usando SQL e utilizar seus dados de um data warehouse do Amazon Redshift para criar aplicações de IA generativa, como geração de texto, análise de sentimentos ou tradução.

Tópicos

- Criar ou atualizar um perfil do IAM para a integração do Amazon Redshift ML com o Amazon Bedrock
- Criar ou atualizar um modelo externo para a integração do Amazon Redshift ML com o Amazon Bedrock
- Usar um modelo externo para a integração do Amazon Redshift ML com o Amazon Bedrock
- Engenharia de prompts para a integração do Amazon Redshift ML com o Amazon Bedrock

Criar ou atualizar um perfil do IAM para a integração do Amazon Redshift ML com o Amazon Bedrock

Esta seção descreve como criar um perfil do IAM para usar a integração do Amazon Redshift ML com o Amazon Bedrock.

Adicione a seguinte política ao perfil do IAM que você usa com a integração do Amazon Redshift ML com o Amazon Bedrock:

- `AmazonBedrockFullAccess`

Para permitir que o Amazon Redshift assuma um perfil para interagir com outros serviços, adicione a seguinte política de confiança aio perfil do IAM:

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Principal": {
            "Service": [
               "redshift.amazonaws.com"
            ]
         },
         "Action": "sts:AssumeRole"
      }
   ]
}
```

Se o cluster ou namespace estiver em uma VPC, siga as instruções em Definir cluster e configuração para administração do Amazon Redshift ML.

Se precisar de uma política mais restritiva, você pode criar uma que inclua somente as permissões do Amazon Bedrock especificadas nas seguintes páginas:

- Definir cluster e configuração para administração do Amazon Redshift ML
- Permissões necessárias para usar o machine learning (ML) do Amazon Redshift)

Para ter informações sobre como a criar um perfil do IAM, consulte Criar um perfil do IAM no Guia do usuário do AWS Identity and Access Management.

Criar ou atualizar um modelo externo para a integração do Amazon Redshift ML com o Amazon Bedrock

Esta seção mostra como criar um modelo externo para usar como interface para o Amazon Bedrock no data warehouse do Amazon Redshift.

Para invocar um modelo do Amazon Bedrock por meio do Amazon Redshift, você deve primeiro executar o comando `CREATE EXTERNAL MODEL`. Esse comando cria um objeto de modelo externo no banco de dados e uma função de usuário associada que você usa para gerar conteúdo textual com o Amazon Bedrock.

O seguinte exemplo de código mostra um comando `CREATE EXTERNAL MODEL` básico:

```
CREATE EXTERNAL MODEL llm_claude
FUNCTION llm_claude_func
IAM_ROLE '<IAM role arn>'
MODEL_TYPE BEDROCK
SETTINGS (
   MODEL_ID 'anthropic.claude-v2:1',
   PROMPT 'Summarize the following text:');
```

O comando `CREATE EXTERNAL MODEL` tem uma interface unificada e consistente com o Amazon Bedrock para todos os modelos de base (FMs) que comportam mensagens. Essa é a opção padrão ao usar o comando `CREATE EXTERNAL MODEL` ou ao especificar explicitamente que o tipo de solicitação deve ser `UNIFIED`. Para ter mais informações, consulte a documentação da API Converse na documentação de API do Amazon Bedrock.

Se um FM não comportar mensagens, você deverá definir a configuração `request_type` como `RAW`, como demonstra o exemplo a seguir. Ao definir `request_type` como `RAW`, você deve criar a solicitação enviada ao Amazon Bedrock ao usar a função de inferência com base no FM selecionado. Habilite o acesso ao modelo Titan Text G1 — Express no Amazon Bedrock antes de executar o exemplo a seguir.

```
CREATE EXTERNAL MODEL titan_raw
FUNCTION func_titan_raw
IAM_ROLE '<IAM role arn>'
MODEL_TYPE BEDROCK
SETTINGS (
   MODEL_ID 'amazon.titan-text-express-v1',
```

```
   REQUEST_TYPE RAW,
   RESPONSE_TYPE SUPER);
```

Se precisar de mais informações sobre uma solicitação de entrada, como total de tokens, você poderá solicitar `RESPONSE_TYPE` para ser `super` ao criar o modelo.

```
CREATE EXTERNAL MODEL patient_recommendations_v2
FUNCTION patient_recommendations_func_v2
IAM_ROLE '<IAM role arn>'
MODEL_TYPE BEDROCK
SETTINGS (
   MODEL_ID 'anthropic.claude-v2',
   PROMPT 'Generate personalized diet plan for following patient:',
   RESPONSE_TYPE SUPER);
```

O parâmetro `PROMPT` para o comando `CREATE EXTERNAL MODEL` é um prompt estático. Se você precisar de um prompt dinâmico para sua aplicação, deverá especificá-lo ao usar a função de inferência. Para obter mais detalhes, consulte Engenharia de prompts para a integração do Amazon Redshift ML com o Amazon Bedrock.

Para ter mais informações sobre a instrução `CREATE EXTERNAL MODEL` e os respectivos parâmetros e configurações, consulte CREATE EXTERNAL MODEL.

Usar um modelo externo para a integração do Amazon Redshift ML com o Amazon Bedrock

Esta seção mostra como invocar um modelo externo para gerar texto em resposta aos prompts fornecidos. Para invocar um modelo externo, use a função de inferência que você criou com `CREATE EXTERNAL MODEL`.

Tópicos

- Inferência com modelos do tipo de solicitação UNIFIED
- Inferência com modelos do tipo de solicitação RAW
- A inferência funciona como funções somente de líder
- Observações sobre o uso de funções de inferência

Inferência com modelos do tipo de solicitação **UNIFIED**

A função de inferência para modelos com o tipo de solicitação `UNIFIED` tem os três seguintes parâmetros, que são passados sequencialmente para a função:

- Texto de entrada (obrigatório): esse parâmetro especifica o texto de entrada que o Amazon Redshift passa para o Amazon Bedrock.
- Configuração de inferência e Campos adicionais de solicitação de modelo (opcional): o Amazon Redshift passa esses parâmetros para os parâmetros correspondentes da API de modelo Converse.

O seguinte exemplo de código mostra como usar uma função de inferência do tipo `UNIFIED`:

```
SELECT llm_claude_func(input_text, object('temperature', 0.7, 'maxtokens', 500))
   FROM some_data;
```

Inferência com modelos do tipo de solicitação **RAW**

A função de inferência para modelos com o tipo de solicitação `RAW` tem apenas um parâmetro do tipo de dados `SUPER`. A sintaxe desse parâmetro depende do modelo do Amazon Bedrock que é usado.

O seguinte exemplo de código mostra como usar uma função de inferência do tipo `RAW`:

```
SELECT llm_titan_func(
    object(
        "inputText", "Summarize the following text: " | input_text,
        "textGenerationConfig", object("temperature", 0.5, "maxTokenCount", 500)
    )
)
FROM some_data;
```

A inferência funciona como funções somente de líder

As funções de inferência para modelos do Amazon Bedrock podem ser executadas como funções somente de nós líderes quando a consulta que as utiliza não faz referência a nenhuma tabela. Isso pode ser útil se você quiser fazer uma pergunta rapidamente a um LLM.

O seguinte exemplo de código mostra como usar uma função de inferência somente de líder:

```
SELECT general_titan_llm_func('Summarize the benefits of LLM on data analytics in 100
 words');
```

Observações sobre o uso de funções de inferência

Observe o seguinte ao usar funções de inferência com a integração do Amazon Redshift ML com o Amazon Bedrock:

- Os nomes dos parâmetros de todos os modelos do Amazon Bedrock diferenciam maiúsculas e minúsculas. Se seus parâmetros não corresponderem aos exigidos pelo modelo, o Amazon Bedrock poderá ignorá-los discretamente.
- O throughput das consultas de inferência é limitada pelas cotas de runtime dos diferentes modelos oferecidos pelo Amazon Bedrock em diferentes regiões. Para ter mais informações, consulte Quotas for Amazon Bedrock no Guia do usuário do Amazon Bedrock.
- Se você precisar de um throughput garantido e consistente, considere a possibilidade de obter throughput provisionado do Amazon Bedrock para o modelo de que você precisa. Para ter mais informações, consulte Increase model invocation capacity with Provisioned Throughput in Amazon Bedrock no Guia do usuário do Amazon Bedrock.
- Consultas de inferência com grandes quantidades de dados podem ter exceções de controle de utilização. Isso se deve às cotas limitadas de tempo de execução do Amazon Bedrock. O Amazon Redshift repete as solicitações várias vezes, mas as consultas ainda assim podem ter controle de utilização porque o throughput para modelos não provisionados pode ser variável.
- Se você encontrar exceções de controle de utilização provenientes do Amazon Bedrock, como `Too many requests, please wait before trying again.`, mesmo com pequenas quantidades de dados, verifique as cotas em Cotas de serviço na sua conta do Amazon Bedrock. Verifique se a cota aplicada no nível da conta é pelo menos igual ao valor da cota padrão da AWS para as solicitações InvokeModel do modelo que você está usando.

Engenharia de prompts para a integração do Amazon Redshift ML com o Amazon Bedrock

Esta seção mostra como usar prompts estáticos com um modelo externo.

Para usar prompts de prefixo e sufixo estáticos em seu modelo externo, forneça-os usando os parâmetros `SUFFIX` e `PROMPT` da instrução `CREATE EXTERNAL MODEL`. Esses prompts são adicionados a cada consulta usando o modelo externo.

O seguinte exemplo mostra como adicionar prompts de prefixo e sufixo a um modelo externo:

```
CREATE EXTERNAL MODEL llm_claude
FUNCTION llm_claude_func
IAM_ROLE '<IAM role arn>'
MODEL_TYPE BEDROCK
SETTINGS (
   MODEL_ID 'anthropic.claude-v2:1',
   PROMPT 'Summarize the following text:',
   SUFFIX 'Respond in an analytic tone');
```

Para usar prompts dinâmicos, você pode fornecê-los ao usar a função de inferência concatenando-os na entrada da função. O seguinte exemplo mostra como usar prompts dinâmicos com uma função de inferência:

```
SELECT llm_claude_func('Summarize the following review:' | input_text | 'The review
 should have formal tone.')
FROM some_data
```

Ajustar o desempenho da consulta

O Amazon Redshift usa consultas baseadas em linguagem de consulta estruturada (SQL) para interagir com dados e objetos no sistema. A linguagem de manipulação de dados (DML) é um subconjunto da SQL usado para ver, adicionar, alterar e excluir dados. A linguagem de definição de dados (DDL) é um subconjunto de SQL usado para adicionar, alterar excluir objetos do banco de dados, como tabelas e visualizações.

Depois que o sistema estiver configurado, você normalmente trabalhará mais com DML, especialmente com o comando SELECT para recuperar e visualizar dados. Para gravar consultas de recuperação de dados eficazes no Amazon Redshift, familiarize-se com o SELECT e aplique as dicas descritas em Práticas recomendadas do Amazon Redshift para projetar tabelas para maximizar a eficiência da consulta.

Para entender como o Amazon Redshift processa as consultas, use as seções Processamento de consulta e Análise e aprimoramento de consultas. Depois, você poderá aplicar essas informações junto com ferramentas de diagnóstico para identificar e remover problemas na performance da consulta.

Para identificar e resolver alguns dos problemas mais comuns e mais sérios que você provavelmente encontrará com as consultas do Amazon Redshift, use a seção Solução de problemas de consulta.

Tópicos

- Processamento de consulta
- Análise e aprimoramento de consultas
- Solução de problemas de consulta

Processamento de consulta

O Amazon Redshift roteia uma consulta SQL enviada por meio do analisador e otimizador para desenvolver um plano de consulta. O mecanismo de execução então converte o plano de consulta em código e envia esse código para nós de computação para execução.

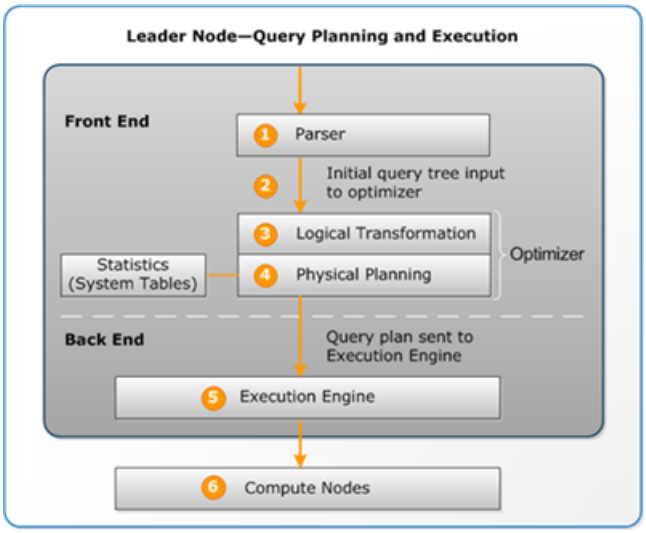
Tópicos

- Planejamento de consulta e fluxo de trabalho de execução
- Criar e interpretar um plano de consulta

- Revisar as etapas do plano de consulta
- Fatores que afetam a performance da consulta

Planejamento de consulta e fluxo de trabalho de execução

A ilustração a seguir fornece uma visão detalhada do planejamento e fluxo de trabalho da execução da consulta.



O planejamento e fluxo de trabalho de execução da consulta seguem as seguintes etapas:

1. O nó líder recebe a consulta e analisa o SQL.
2. O analisador produz uma árvore de consulta inicial que é uma representação lógica da consulta original. Em seguida, o Amazon Redshift insere essa árvore de consulta no otimizador de consultas.
3. O otimizador avalia e, se necessário, reescreve a consulta para maximizar sua eficiência. Esse processo às vezes resulta na criação de múltiplas consultas relacionadas para substituir uma única consulta.
4. O otimizador gera um plano de consulta (ou vários, se a etapa anterior resultou em múltiplas consultas) para a execução com a melhor performance. O plano de consulta especifica as opções de execução, tais como tipos de junção, ordem de junção, opções de agregação e os requisitos de distribuição de dados.

   Use o comando EXPLAIN para visualizar o plano de consulta. O plano de consulta é uma ferramenta fundamental para analisar e ajustar consultas complexas. Para obter mais informações, consulte Criar e interpretar um plano de consulta.

5. O mecanismo de execução converte o plano de consulta em etapas, segmentos e fluxos:

   Etapa

   Cada etapa é uma operação individual necessária durante a execução da consulta. As etapas podem ser combinadas para permitir que os nós de computação executem uma consulta, junção ou outra operação do banco de dados.

   Segmento

   Uma combinação de várias etapas que podem ser realizadas por um único processo, como também a menor unidade de compilação executável por uma fatia do nó de computação. Uma fatia é a unidade de processamento paralelo no Amazon Redshift. Os segmentos em um fluxo são executados em paralelo.

   Fluxo

   Uma coleção de segmentos a serem divididos entre as fatias disponíveis do nó de computação.

   O mecanismo de execução gera o código compilado com base nas etapas, segmentos e fluxos. O código compilado é executado mais rapidamente que o código interpretado e usa menos capacidade computacional. Este código compilado é, então, transmitido para os nós de computação.

   > **Note**
   >
   > Ao avaliar suas consultas, você deve sempre comparar os tempos para segunda execução de uma consulta, pois a primeira execução inclui as despesas gerais de compilação do código. Para obter mais informações, consulte Fatores que afetam a performance da consulta.

6. As fatias do nó de computação executam os segmentos da consulta em paralelo. Como parte desse processo, o Amazon Redshift aproveita a comunicação de rede, a memória e o gerenciamento de disco otimizados para passar resultados intermediários de uma etapa do plano de consulta para a próxima. Isso também ajuda a acelerar a execução da consulta.

As etapas 5 e 6 acontecem uma vez para cada fluxo. O mecanismo cria os segmentos executáveis para um fluxo e os envia para os nós de computação. Quando os segmentos daquele fluxo são completados, o mecanismo gera os segmentos para o próximo fluxo. Desta forma, o mecanismo

pode analisar o que aconteceu no fluxo anterior (por exemplo, se as operações foram baseadas em disco) para influenciar a geração de segmentos no fluxo seguinte.

Quando os nós de computação terminam, eles retornam os resultados da consulta para o nó de liderança para processamento final. O nó de liderança efetua a fusão dos dados em um único conjunto de resultados e aborda qualquer classificação ou agregação necessária. O nó de liderança, então, retorna os resultados ao cliente.

> **Note**
>
> Os nós de computação podem retornar alguns dados ao nó de liderança durante a execução de consulta, caso necessário. Por exemplo, se você tiver uma subconsulta com uma cláusula LIMIT, o limite será aplicado no nó de liderança antes que os dados sejam redistribuídos através do cluster para processamento adicional.

Criar e interpretar um plano de consulta

É possível usar o plano de consulta para obter informações sobre operações individuais necessárias para executar uma consulta. Antes de trabalhar com um plano de consulta, recomendamos que você primeiro entenda como o Amazon Redshift lida com o processamento de consultas e a criação de planos de consulta. Para obter mais informações, consulte Planejamento de consulta e fluxo de trabalho de execução.

Para criar um plano de consulta, execute o comando EXPLAIN seguido pelo texto real da consulta. O plano de consulta oferece as seguintes informações:

- Quais operações o mecanismo de execução executa, lendo os resultados de baixo para cima.
- Que tipo de etapa cada operação executa.
- Quais tabelas e colunas são usadas em cada operação.
- Quantos dados são processados em cada operação, em termos de número de linhas e largura de dados em bytes.
- O custo relativo da operação. Custo é uma medida que compara os tempos relativos de execução das etapas em um plano. O custo não fornece qualquer informação precisa sobre o real tempo de execução ou consumo de memória, nem fornece uma comparação significativa entre planos de execução. Ele fornece uma indicação de quais operações em uma consulta estão consumindo a maioria dos recursos.

O comando EXPLAIN não executa a consulta de fato. Ele mostra apenas o plano que o Amazon Redshift executa se a consulta for executada nas condições operacionais atuais. Se você alterar o esquema ou os dados de uma tabela e executar ANALYZE novamente para atualizar os metadados estatísticos, o plano de consulta poderá ser diferente.

A saída do plano de consulta pelo comando EXPLAIN é uma exibição de alto nível simplificada da execução da consulta. Ela não ilustra os detalhes do processamento paralelo da consulta. Para visualizar informações detalhadas, execute a própria consultas e obtenha as informações de resumo na visualização SVL_QUERY_SUMMARY ou SVL_QUERY_REPORT. Para obter mais informações sobre como usar essas visualizações, consulte Analisar o resumo da consulta.

O seguinte exemplo mostra a saída de EXPLAIN para uma consulta GROUP BY simples na tabela EVENT:

```
explain select eventname, count(*) from event group by eventname;

                            QUERY PLAN
------------------------------------------------------------------
XN HashAggregate  (cost=131.97..133.41 rows=576 width=17)
  ->  XN Seq Scan on event  (cost=0.00..87.98 rows=8798 width=17)
```

EXPLAIN retorna as seguintes métricas para cada operação:

Custo

Um valor relativo que é útil para comparar operações em um plano. Custo consiste em dois valores decimais separados por dois pontos, por exemplo `cost=131.97..133.41`. O primeiro valor, nesse caso 131,97, fornece o custo relativo de retornar a primeira linha para essa operação. O segundo valor, nesse caso 133,41, fornece o custo relativo de concluir a operação. Os custos em plano de consulta são cumulativos à medida que você lê o plano, portanto o custo HashAggregate neste exemplo (131,97..133,41) inclui o custo da Seq Scan abaixo dele (0.00..87.98).

Linhas

O número previsto de linhas a retornar. Neste exemplo, a previsão de retorno da varredura é 8.798 linhas. O operador HashAggregate deve, sozinho, retornar 576 linhas (depois que os nomes de evento duplicados são descartados do conjunto de resultados).

> **Note**
>
> A estimativa de linhas baseia-se nas estatísticas disponíveis geradas pelo comando ANALYZE. Se ANALYZE não tiver sido executado recentemente, a estimativa será menos confiável.

Largura

A largura estimada da linha média, em bytes. Neste exemplo, a previsão de largura da linha média é de 17 bytes.

Operadores EXPLAIN

Esta seção descreve resumidamente os operadores que você verá com mais frequência na saída de EXPLAIN. Para uma lista completa dos operadores, consulte EXPLAIN na seção de comandos SQL.

Operador de varredura sequencial

O operador de varredura sequencial (Seq Scan) indica uma varredura de tabela. A Seq Scan faz a varredura de cada coluna na tabela sequencialmente do começo ao fim e avalia as restrições da consulta (na cláusula WHERE) para cada linha.

Operadores de junção

O Amazon Redshift seleciona operadores de junção com base no design físico das tabelas que estão sendo juntadas, a localização dos dados necessários para a junção e os requisitos específicos da própria consulta.

- Loop aninhado

  Um loop aninhado, a junção menos ideal, é usado sobretudo para junções cruzadas (produtos cartesianos) e algumas junções de desigualdade.
- Hash e junção hash

  Normalmente mais rápida que uma junção de loop aninhado, hash e junção hash são usadas para junções internas e junções externas da esquerda e direita. Esses operadores são usados para juntar tabelas onde as colunas de junção não são chaves de distribuição e chaves de classificação. O operador hash cria a tabela hash para a junção da tabela interna; o operador de

junção hash lê a tabela externa, hash a coluna de junção e localiza correspondências na tabela hash interna.

- Junção de mesclagem

  Normalmente a junção mais rápida, uma junção de mesclagem é usada para junções internas e externas. A junção de mesclagem não é usada para junções completas. Este operador é usado para juntar tabelas onde as colunas de junção são chaves de distribuição e chaves de classificação e quando menos que 20 por cento das tabelas para junção não estão classificadas. Ele lê duas tabelas classificadas na ordem e localiza as linhas correspondentes. Para ver a porcentagem de linhas não classificadas, consulte a tabela de sistema SVV_TABLE_INFO.

- Junção espacial

  Normalmente, uma junção rápida com base na proximidade de dados espaciais, usados em tipos de dados `GEOMETRY` e `GEOGRAPHY`.

Operadores de agregação

O plano de consulta usa os seguintes operadores em consultas que envolvem funções agregadas e operações GROUP BY.

- Agregar

  Operador para funções agregadas escalares, tais como AVG e SUM.

- HashAggregate

  Operador para funções agregadas agrupadas não classificadas.

- GroupAggregate

  Operador para funções agregadas agrupadas classificadas.

Operadores de classificação

O plano de consulta usa os seguintes operadores quando as consultas precisam classificar ou mesclar conjuntos de resultados.

- Classificar

  Avalia a cláusula ORDER BY e outras operações de classificação, tais como classificações exigidas por consultas e junções UNION, consultas SELECT DISTINCT e funções de janela.

- Mesclar

  Produz resultados finais classificados de acordo com os resultados classificados intermediários derivados de operações paralelas.

Operadores UNION, INTERSECT e EXCEPT

O plano de consulta usa os seguintes operadores para consultas que envolvem operações de conjunto com UNION, INTERSECT e EXCEPT.

- Subconsulta

  Usada para executar consultas UNION.

- Hash Intersect Distinct

  Usado para executar consultas INTERSECT .

- SetOp Except

  Usada para executar consultas EXCEPT (ou MINUS).

Outros operadores

Os operadores a seguir também aparecem frequentemente na saída de EXPLAIN para consultas de rotina.

- Exclusivo

  Remove duplicidades para consultas SELECT DISTINCT e UNION.

- Limite

  Processa a cláusula LIMIT.

- Window

  Executa funções de janela.

- Resultado

  Executa funções escalares que não envolvem qualquer acesso à tabela.

- Subplan

Usado para determinadas subconsultas.

- Rede da

  Envia resultados intermediários ao nó de liderança para processamento adicional.

- Materialize

  Salva linhas para entrada em junções de loop aninhado e algumas junções de mesclagem.

Junções em EXPLAIN

O otimizador de consulta usa diferentes tipos de junção para recuperar dados da tabela, dependendo da estrutura da consulta e das tabelas subjacentes. A saída de EXPLAIN menciona o tipo de junção, as tabelas usadas e a forma que os dados da tabela estão distribuídos pelo cluster para descrever como a consulta é processada.

Exemplos de tipos de junção

Os seguintes exemplos mostram os diferentes tipos de junção que o otimizador de consulta pode utilizar. O tipo de junção usado no plano de consulta depende do design físico das tabelas envolvidas.

Exemplo: junção hash de duas tabelas

A seguinte consulta faz a junção de EVENT e CATEGORY na coluna CATID. CATID é a chave de distribuição e classificação para CATEGORY, mas não para EVENT. Uma junção hash é realizada com EVENT como a tabela externa e CATEGORY como a tabela interna. Como CATEGORY é a tabela menor, o planejador transmite uma cópia dela para os nós de computação durante o processamento da consulta usando DS_BCAST_INNER. O custo da junção neste exemplo corresponde à maioria do custo cumulativo do plano.

```
explain select * from category, event where category.catid=event.catid;

                              QUERY PLAN
------------------------------------------------------------------------
 XN Hash Join DS_BCAST_INNER  (cost=0.14..6600286.07 rows=8798 width=84)
   Hash Cond: ("outer".catid = "inner".catid)
   ->  XN Seq Scan on event  (cost=0.00..87.98 rows=8798 width=35)
   ->  XN Hash  (cost=0.11..0.11 rows=11 width=49)
         ->  XN Seq Scan on category  (cost=0.00..0.11 rows=11 width=49)
```

> **Note**
>
> Os recuos alinhados para operadores na saída de EXPLAIN às vezes indicam que aquelas operações não dependem uma das outras e podem começar em paralelo. No exemplo anterior, embora a varredura na tabela EVENT e a operação de hash estejam alinhadas, a varredura de EVENT deve esperar até que a operação de hash seja totalmente concluída.

Exemplo: junção de mesclagem de duas tabelas

A seguinte consulta também usa SELECT *, mas faz a junção de SALES e LISTING na coluna LISTID, onde LISTID foi definida tanto como a chave de distribuição quanto a chave de classificação para ambas as tabelas. Um junção de mesclagem é escolhida e nenhuma redistribuição de dados é necessária para a junção (DS_DIST_NONE).

```
explain select * from sales, listing where sales.listid = listing.listid;
QUERY PLAN
------------------------------------------------------------------------------
XN Merge Join DS_DIST_NONE  (cost=0.00..6285.93 rows=172456 width=97)
  Merge Cond: ("outer".listid = "inner".listid)
  ->  XN Seq Scan on listing  (cost=0.00..1924.97 rows=192497 width=44)
  ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=53)
```

O seguinte exemplo demonstra os diferentes tipos de junção dentro da mesma consulta. Como no exemplo anterior, a junção de mesclagem de SALES e LISTING ocorre, mas a junção da terceira tabela, EVENT, deve ser uma junção hash com os resultados da junção de mesclagem. Outra vez, a junção hash incorre em custos de transmissão.

```
explain select * from sales, listing, event
where sales.listid = listing.listid and sales.eventid = event.eventid;
                                  QUERY PLAN
------------------------------------------------------------------------------
XN Hash Join DS_BCAST_INNER  (cost=109.98..3871130276.17 rows=172456 width=132)
  Hash Cond: ("outer".eventid = "inner".eventid)
  ->  XN Merge Join DS_DIST_NONE  (cost=0.00..6285.93 rows=172456 width=97)
        Merge Cond: ("outer".listid = "inner".listid)
        ->  XN Seq Scan on listing  (cost=0.00..1924.97 rows=192497 width=44)
        ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=53)
  ->  XN Hash  (cost=87.98..87.98 rows=8798 width=35)
        ->  XN Seq Scan on event  (cost=0.00..87.98 rows=8798 width=35)
```

Exemplo: junção, agregação e classificação

A seguinte consulta executa a junção hash das tabelas SALES e EVENT, seguidas por operações de agregação e classificação para contabilizar a função SUM agrupada e a cláusula ORDER BY. O operador de classificação inicial executa em paralelo nos nós de computação. Então, o operador de rede envia os resultados ao nó de liderança, onde o operador de mesclagem produz os resultados classificados finais.

```
explain select eventname, sum(pricepaid) from sales, event
where sales.eventid=event.eventid group by eventname
order by 2 desc;
                                          QUERY PLAN
------------------------------------------------------------------------------
 XN Merge  (cost=1002815366604.92..1002815366606.36 rows=576 width=27)
  Merge Key: sum(sales.pricepaid)
  ->  XN Network  (cost=1002815366604.92..1002815366606.36 rows=576 width=27)
        Send to leader
        ->  XN Sort  (cost=1002815366604.92..1002815366606.36 rows=576 width=27)
              Sort Key: sum(sales.pricepaid)
              ->  XN HashAggregate  (cost=2815366577.07..2815366578.51 rows=576
 width=27)
                    ->  XN Hash Join DS_BCAST_INNER  (cost=109.98..2815365714.80
 rows=172456 width=27)
                          Hash Cond: ("outer".eventid = "inner".eventid)
                          ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456
 width=14)
                          ->  XN Hash  (cost=87.98..87.98 rows=8798 width=21)
                                ->  XN Seq Scan on event  (cost=0.00..87.98 rows=8798
 width=21)
```

Redistribuição de dados

A saída EXPLAIN para junções também especifica um método de como os dados serão movidos em torno de um cluster para facilitar a junção. Esta movimentação de dados pode ser uma transmissão ou uma redistribuição. Em uma transmissão, os valores dos dados de um lado de uma junção são copiados de cada nó de computação para todos os outros nós de computação, de forma que todos os nós de computação tenham uma cópia completa dos dados. Em uma redistribuição, os valores dos dados participantes são enviados de sua fatia atual para uma nova fatia (possivelmente em um nó diferente). Os dados são tipicamente redistribuídos para corresponder à chave de distribuição de outra tabela que participa da junção se aquela chave de distribuição for uma das colunas de junção.

Se nenhuma das tabelas possui chaves de distribuição em uma das colunas de junção, ambas as tabelas são distribuídas ou a tabela interna é transmitida para cada nó.

A saída EXPLAIN também faz referência à tabelas internas e externas. A varredura da tabela interna é realizada primeiro e aparece mais próximo da parte inferior do plano de consulta. A tabela interna é a tabela que é examinada quanto a correspondências. Ela é geralmente mantida na memória e normalmente é a tabela de origem para hashing e, se possível, é a menor das duas tabelas da junção. A tabela externa é a origem das linhas para correspondência na tabela interna. Ela é geralmente lida a partir do disco. O otimizador de consulta escolhe a tabela interna e externa com base em estatísticas do banco de dados e do comando ANALYZE executado mais recentemente. A ordem das tabelas na cláusula FROM de uma consulta não determina qual tabela é interna e qual é externa.

Use the following attributes in query plans to identify how data is moved to facilitate a query:

- DS_BCAST_INNER

  Uma cópia de toda a tabela interna é transmitida a todos os nós de computação.

- DS_DIST_ALL_NONE

  Nenhuma redistribuição é necessária, pois a tabela interna já foi distribuída a todos os nós usando DISTSTYLE ALL.

- DS_DIST_NONE

  Nenhuma tabela é redistribuída. As junções colocadas são possíveis pois as fatias correspondentes são juntadas sem movimentação de dados entre nós.

- DS_DIST_INNER

  A tabela interna é redistribuída.

- DS_DIST_OUTER

  A tabela externa é redistribuída.

- DS_DIST_ALL_INNER

  Toda a tabela interna é redistribuída a uma única fatia, pois a tabela externa usa DISTSTYLE ALL.

- DS_DIST_BOTH

  Ambas as tabelas são redistribuídas.

Revisar as etapas do plano de consulta

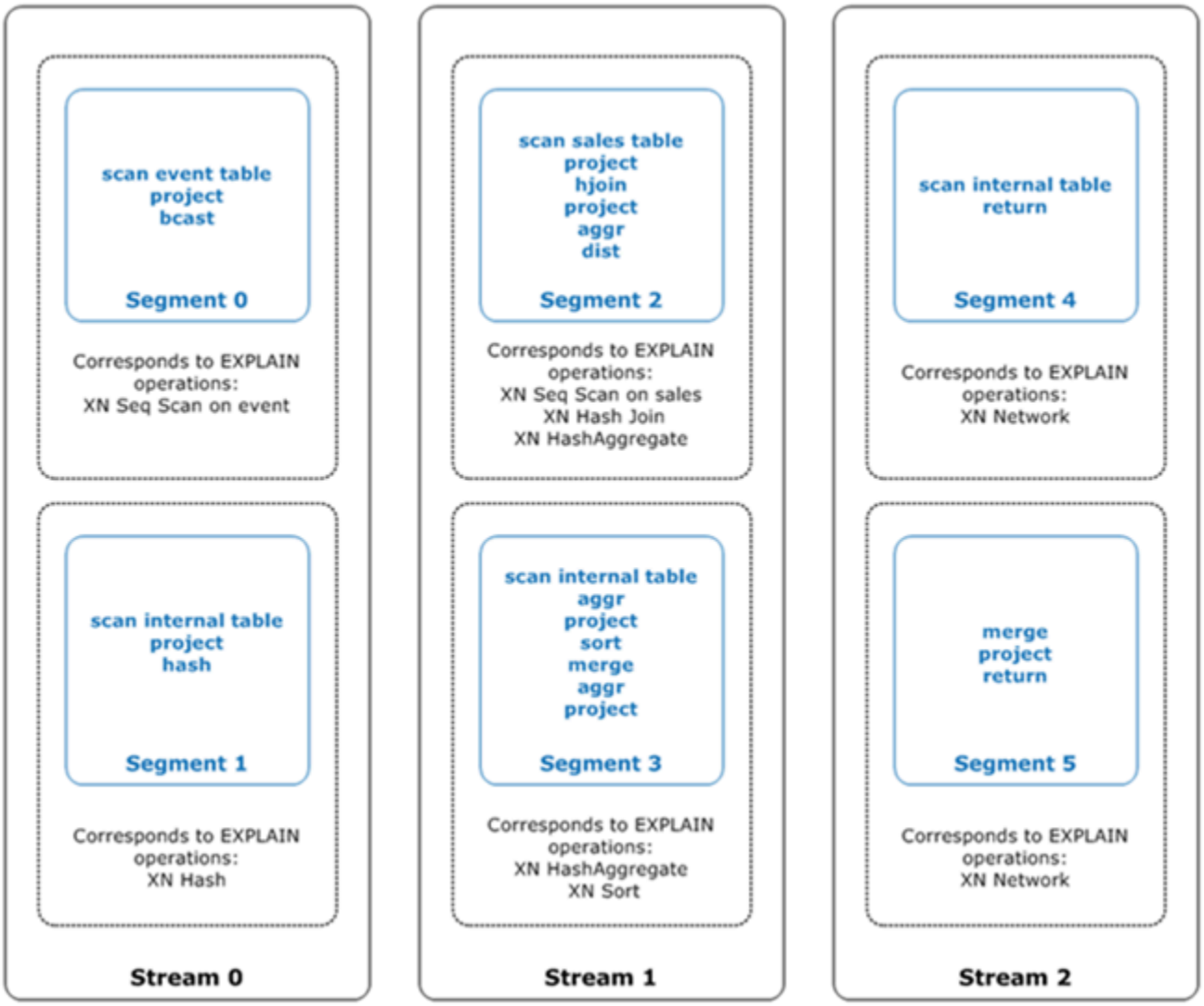
Você pode visualizar as etapas em um plano de consulta executando o comando EXPLAIN. O exemplo a seguir mostra uma consulta SQL e explica a saída. Ao ler o plano de consulta de baixo para cima, você verá cada uma das operações lógicas usadas para executar a consulta. Para obter mais informações, consulte Criar e interpretar um plano de consulta.

```
explain
select eventname, sum(pricepaid) from sales, event
where sales.eventid = event.eventid
group by eventname
order by 2 desc;
```

```
XN Merge  (cost=1002815366604.92..1002815366606.36 rows=576 width=27)
  Merge Key: sum(sales.pricepaid)
  ->  XN Network  (cost=1002815366604.92..1002815366606.36 rows=576 width=27)
        Send to leader
        ->  XN Sort  (cost=1002815366604.92..1002815366606.36 rows=576 width=27)
              Sort Key: sum(sales.pricepaid)
              ->  XN HashAggregate  (cost=2815366577.07..2815366578.51 rows=576
 width=27)
                    ->  XN Hash Join DS_BCAST_INNER  (cost=109.98..2815365714.80
 rows=172456 width=27)
                          Hash Cond: ("outer".eventid = "inner".eventid)
                          ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456
 width=14)
                          ->  XN Hash  (cost=87.98..87.98 rows=8798 width=21)
                                ->  XN Seq Scan on event  (cost=0.00..87.98 rows=8798
 width=21)
```

Como parte da geração de um plano de consulta, o otimizador de consulta divide o plano em fluxos, segmentos e etapas. O otimizador de consulta divide o plano para se preparar para distribuir os dados e o workload da consulta para os nós de computação. Para obter mais informações sobre segmentos e etapas, consulte Planejamento de consulta e fluxo de trabalho de execução.

A ilustração a seguir mostra a consulta anterior e o plano de consulta associado. Ele exibe como as operações de consulta envolvidas são mapeadas para as etapas que o Amazon Redshift usa para gerar o código compilado para as fatias do nó de computação. Cada operação do plano de consulta mapeia para várias etapas dentro dos segmentos e, às vezes, para vários segmentos dentro dos fluxos.



Nesta ilustração, o otimizador de consulta executa o plano de consulta da seguinte forma:

1. No `Stream 0`, a consulta executa `Segment 0` com uma operação de varredura sequencial para verificar a tabela `events`. A consulta continua para `Segment 1` com uma operação de hash para criar a tabela de hash para a tabela interna na junção.
2. No `Stream 1`, a consulta executa `Segment 2` com uma operação de varredura sequencial para verificar a tabela `sales`. Ele continua com `Segment 2` com uma junção hash para unir tabelas onde as colunas de junção não são chaves de distribuição e chaves de classificação. Ele continua novamente com `Segment 2` com um agregado de hash para agregar resultados. Depois, a consulta executa `Segment 3` com uma operação de agregação de hash para executar funções agregadas agrupadas não classificadas e uma operação de classificação para avaliar a cláusula ORDER BY e outras operações de classificação.

3. Em `Stream 2`, a consulta executa uma operação de rede no `Segment 4` e `Segment 5`, para enviar resultados intermediários para o nó líder para processamento posterior.

O último segmento de uma consulta retorna os dados. Se o conjunto de retorno for agregado ou classificado, cada um dos nós de computação enviará sua parte do resultado intermediário para o nó líder. O nó líder mescla então os dados para que o resultado final possa ser enviado de volta para o cliente solicitante.

Para obter mais informações sobre operadores EXPLAIN, consulte EXPLAIN.

Fatores que afetam a performance da consulta

Inúmeros fatores podem afetar a performance da consulta. Os seguintes aspectos de seus dados, cluster operações de banco de dados têm um papel na velocidade do processamento de suas consultas.

- Número de nós, processadores ou fatias – Um nó de computação é dividido em fatias. Mais nós significam mais processadores e mais fatias, o que permite que suas consultas sejam processadas mais rápido executando partes da consulta simultaneamente entre as fatias. No entanto, mais nós também significam mais despesas, portanto você precisa encontrar o equilíbrio entre custo e a performance mais apropriada para seu sistema. Para obter mais informações sobre a arquitetura de cluster do Amazon Redshift, consulte Arquitetura de sistema do data warehouse.
- Tipos de nó: um cluster do Amazon Redshift pode usar um dos vários tipos de nó. Cada tipo de nó oferece diferentes tamanhos e limites para ajudá-lo a escalar seu cluster adequadamente. O tamanho do nó determina a capacidade de armazenamento, memória, CPU e preço de cada nó no cluster. Para obter mais informações sobre tipos de nós, consulte “Clusters do Amazon Redshift” no Guia de gerenciamento de clusters do Amazon Redshift.
- Distribuição de dados – O Amazon Redshift armazena dados da tabela nos nós de computação de acordo com o estilo de distribuição da tabela. Quando você executa uma consulta, o otimizador de consulta redistribui os dados aos nós de computação conforme necessário para executar junções e agregações. A escolha do estilo correto de distribuição para uma tabela ajuda a minimizar o impacto das etapas de redistribuição ao localizar os dados onde eles devem estar antes que as junções sejam executadas. Para obter mais informações, consulte Distribuição de dados para otimização de consultas.
- Ordem de classificação de dados – O Amazon Redshift armazena dados da tabela no disco em ordem classificada de acordo com as chaves de classificação de uma tabela. O otimizador de consulta e o processador de consulta usam as informações sobre onde os dados estão localizados

para reduzir o número de blocos que precisam ser varridos e, portanto, melhoram a velocidade da consulta. Para obter mais informações, consulte Chaves de classificação.

- Tamanho do conjunto de dados – Um volume maior de dados no cluster pode diminuir a performance da consulta para consultas, porque mais linhas precisam ser verificadas e redistribuídas. Você pode atenuar este efeito com a limpeza e arquivamento regulares dos dados e ao usar um predicado para restringir o conjunto de dados da consulta.
- Operações simultâneas – A execução de várias operações ao mesmo tempo pode afetar a performance da consulta. Cada operação ocupa uma ou mais vagas em uma fila de consulta disponível e usa a memória associada a estas vagas. Se outras operações estiverem em execução, pode não haver vagas disponíveis suficientes na fila de consulta. Nesse caso, a consulta terá que esperar a abertura de vagas para poder iniciar o processamento. Para obter mais informações sobre a criação e configuração de filas de consulta, consulte Gerenciamento do workload.
- Estrutura da consulta – Como sua consulta é escrita afeta sua performance. Sempre que possível, grave consultas para processar e retornar a menor quantidade de dados de acordo com suas necessidades. Para obter mais informações, consulte Práticas recomendadas do Amazon Redshift para criar consultas.
- Compilação de código: o Amazon Redshift gera e compila o código otimizado para cada plano de execução de consulta. O código compilado é executado mais rápido porque elimina a sobrecarga de usar um interpretador. Para minimizar a latência de novas consultas e, ao mesmo tempo, preservar os benefícios de performance do código compilado, o Amazon Redshift usa uma técnica chamada composição. A composição gera um arranjo leve de lógica preexistente para processar novas consultas imediatamente e, ao mesmo tempo, compilar código altamente otimizado e específico da consulta em segundo plano. Isso remove a compilação do caminho crítico da execução da consulta, para que novas consultas sejam iniciadas mais rapidamente e ofereçam performance consistente com as execuções subsequentes.

  O Amazon Redshift também usa um serviço de compilação sem servidor para escalar compilações de consulta além dos recursos de computação de um cluster do Amazon Redshift. Os segmentos de código compilados são armazenados em cache localmente no cluster e em um cache praticamente ilimitado que se mantém após a reinicialização do cluster. As execuções subsequentes da mesma consulta são mais rápidas porque podem pular a fase de compilação. Usando um serviço de compilação escalável, o Amazon Redshift compele código em paralelo para oferecer uma performance consistentemente rápida.

Análise e aprimoramento de consultas

A recuperação de informações de um data warehouse do Amazon Redshift requer a execução de consultas complexas em quantidades extremamente grandes de dados, o que pode exigir um longo tempo de processamento. Para garantir que as consultas sejam processadas o mais rápido possível, há várias ferramentas que você pode usar para identificar possíveis problemas de performance.

Tópicos

- Fluxo de trabalho da análise de consulta
- Revisar alertas da consulta
- Analisar o plano de consulta
- Analisar o resumo da consulta
- Aprimoramento do desempenho das consultas
- Consultas de diagnóstico para ajuste da consulta

Fluxo de trabalho da análise de consulta

Se uma consulta estiver levando mais tempo do que o esperado, use as seguintes etapas para identificar e corrigir problemas que possam estar afetando negativamente a performance da consulta. Se você não tiver certeza de quais consultas em seu sistema podem se beneficiar do ajuste de performance, comece executando uma consulta diagnóstica em Identificar consultas que são os principais candidatos para ajuste.

1. Garanta que suas tabelas sejam projetadas de acordo com as práticas recomendadas. Para obter mais informações, consulte Práticas recomendadas do Amazon Redshift para projetar tabelas.
2. Veja se você pode excluir ou arquivar alguns dados não necessários em suas tabelas. Por exemplo, suponha que suas consultas sempre visem os últimos 6 meses de dados, mas você tenha o valor dos últimos 18 meses em suas tabelas. Nesse caso, você pode excluir ou arquivar os dados mais antigos para reduzir o número de registros que precisam ser verificados e distribuídos.
3. Execute o comando VACUUM nas tabelas da consulta para recuperar espaço e reclassificar as linhas. A execução de VACUUM ajuda se a região não classificada for grande e a consulta usar a chave de classificação em uma junção ou predicado.
4. Execute o comando ANALYZE nas tabelas da consulta para garantir que as estatísticas sejam atualizadas. A execução de ANALYZE ajuda se qualquer uma das tabelas da consulta tiverem

recentemente mudado muito de tamanho. Se a execução completa do comando ANALYZE levar muito tempo, execute ANALYZE em uma única coluna para reduzir o tempo de processamento. Essa abordagem ainda atualizará as estatísticas de tamanho da tabela; o tamanho da tabela é um fator significativo no planejamento da consulta.

5. Certifique-se de que sua consulta tenha sido executada uma vez para cada tipo de cliente (com base em qual tipo de protocolo de conexão o cliente usa) para que a consulta seja compilada e armazenada em cache. Essa abordagem acelera execuções subsequentes da consulta. Para obter mais informações, consulte Fatores que afetam a performance da consulta.
6. Verifique a tabela STL_ALERT_EVENT_LOG para identificar e corrigir possíveis problemas com sua consulta. Para obter mais informações, consulte Revisar alertas da consulta.
7. Execute o comando EXPLAIN para obter o plano de consulta e usá-lo para otimizar a consulta. Para obter mais informações, consulte Analisar o plano de consulta.
8. Use as exibições SVL_QUERY_SUMMARY e SVL_QUERY_REPORT para obter informações de resumo e para usá-las para otimizar a consulta. Para obter mais informações, consulte Analisar o resumo da consulta.

Às vezes, uma consulta que deveria ser executada rapidamente é forçada a esperar a conclusão de outra pesquisa de execução mais longa. Nesse caso, você pode não ter nada a melhorar na própria consulta, mas pode melhorar a performance geral do sistema ao criar e usar filas de consulta para diferentes tipos de consultas. Para obter uma ideia de tempo de espera na fila para suas consultas, consulte Como revisar os tempos de espera na fila para consultas. Para obter mais informações sobre a configuração de filas de consultas, consulte Gerenciamento do workload.

Revisar alertas da consulta

Para usar a tabela de sistema STL_ALERT_EVENT_LOG para identificar e corrigir problemas potenciais de performance com sua consulta, siga estas etapas:

1. Execute o seguinte para determinar o ID de sua consulta:

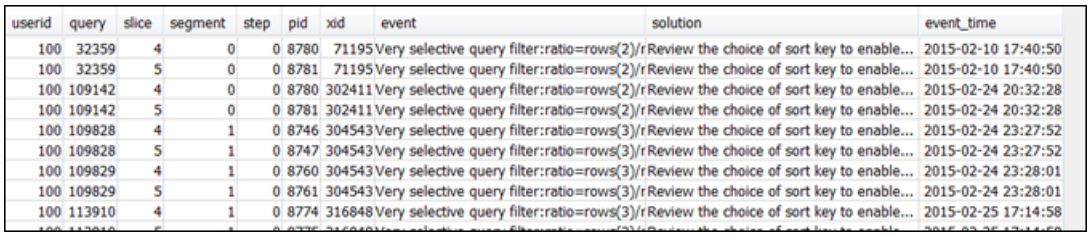
```
select query, elapsed, substring
from svl_qlog
order by query
desc limit 5;
```

   Examine o texto truncado da consulta no campo `substring` para determinar qual valor de `query` deve ser selecionado. Se você executou a consulta mais de uma vez, use o valor de `query`

da linha com o menor valor `elapsed`. Esta é a linha para a versão compilada. Se você estiver executando várias consultas, poderá aumentar o valor usado pela cláusula LIMIT usada para certificar-se de que sua consulta seja incluída.

2. Selecione linhas do STL_ALERT_EVENT_LOG para sua consulta:

```
Select * from stl_alert_event_log where query = MyQueryID;
```



userid	query	slice	segment	step	pid	xid	event	solution	event_time
100	32359	4	0	0	8780	71195	Very selective query filter:ratio=rows(2)/r	Review the choice of sort key to enable...	2015-02-10 17:40:50
100	32359	5	0	0	8781	71195	Very selective query filter:ratio=rows(2)/r	Review the choice of sort key to enable...	2015-02-10 17:40:50
100	109142	4	0	0	8780	302411	Very selective query filter:ratio=rows(2)/r	Review the choice of sort key to enable...	2015-02-24 20:32:28
100	109142	5	0	0	8781	302411	Very selective query filter:ratio=rows(2)/r	Review the choice of sort key to enable...	2015-02-24 20:32:28
100	109828	4	1	0	8746	304543	Very selective query filter:ratio=rows(3)/r	Review the choice of sort key to enable...	2015-02-24 23:27:52
100	109828	5	1	0	8747	304543	Very selective query filter:ratio=rows(3)/r	Review the choice of sort key to enable...	2015-02-24 23:27:52
100	109829	4	1	0	8760	304543	Very selective query filter:ratio=rows(3)/r	Review the choice of sort key to enable...	2015-02-24 23:28:01
100	109829	5	1	0	8761	304543	Very selective query filter:ratio=rows(3)/r	Review the choice of sort key to enable...	2015-02-24 23:28:01
100	113910	4	1	0	8774	316848	Very selective query filter:ratio=rows(3)/r	Review the choice of sort key to enable...	2015-02-25 17:14:58

3. Avalie os resultados para sua consulta. Use a tabela a seguir para localizar possíveis soluções para quaisquer problemas que você tenha identificado.

> **Note**
>
> Nem todas as consultas terão linhas em STL_ALERT_EVENT_LOG, somente aquelas com problemas identificados.

Problema	Valor do evento	Valor da solução	Solução recomendada
As estatísticas para as tabelas na consulta estão ausentes ou desatualizadas.	Ausência de estatísticas do planejador de consultas	Execute o comando ANALYZE	Consulte Estatísticas de tabela ausentes ou desatualizadas.
Há uma junção de loop aninhado (a junção menos ideal) no plano de consulta.	Junção de loop aninhado no plano de consulta	Revise os predicados de junção para evitar produtos cartesianos	Consulte Loop aninhado.

Problema	Valor do evento	Valor da solução	Solução recomendada
A varredura ignorou um número relativamente grande de linhas que estão marcadas como excluídas, mas não como limpas, ou linhas que foram inseridas, mas não confirmadas.	Efetuou a varredura de um grande número de linhas excluídas	Execute o comando de VACUUM para recuperar o espaço excluído	Consulte Linhas fantasmas ou linhas não confirmadas.
Mais de 1.000.000 de linhas foram redistribuídas para uma junção hash ou agregação.	Um grande número de linhas foi distribuído pelas linhas network:RowCount para processar a agregação	Revise a escolha de chave de distribuição para colocar a junção ou agregação	Consulte Distribuição de dados pouco satisfatória.
Mais de 1.000.000 de linhas foram transmitidas para uma junção hash.	Transmitido um grande número de linhas na rede	Revise a escolha da chave de distribuição para colocar a junção e considerar o uso de tabelas distribuídas	Consulte Distribuição de dados pouco satisfatória.
Um estilo de redistribuição DS_DIST_ALL_INNER foi indicado no plano de consulta, o que força uma execução em série, pois toda a tabela interna foi redistribuída para um único nó.	DS_DIST_ALL_INNER para junção hash no plano de consulta	Revise a escolha de estratégia de distribuição para distribuir a tabela interna em vez da externa	Consulte Distribuição de dados pouco satisfatória.

Analisar o plano de consulta

Execute o comando EXPLAIN para obter um plano de consulta.

Antes de analisar o plano de consulta, você deve ser familiarizar com sua leitura. Se você não estiver familiarizado com a leitura de um plano de consulta, sugerimos a leitura de Criar e interpretar um plano de consulta antes de continuar.

Para analisar os dados fornecidos pelo plano de consulta, siga estas etapas:

1. Identifique as etapas com o custo mais alto. Concentre-se em otimizar estas etapas ao avançar pelas etapas restantes.
2. Veja os tipos de junção:
   - Loop aninhado: Tais junções geralmente ocorrem quando uma condição de junção é omitida. Para soluções recomendadas, consulte Loop aninhado.
   - Hash e junção hash: Junções hash são usadas durante a junção de tabelas em que as colunas de junção não são chaves de distribuição e também não são chaves de classificação. Para soluções recomendadas, consulte Junção de hash.
   - Junção de mesclagem: Nenhuma alteração é necessária.
3. Observe qual tabela é usada para a junção interna e qual é usada para a junção externa. O mecanismo de consulta geralmente escolhe a menor tabela para a junção interna e a maior tabela para a junção externa. Se essa escolha não ocorre, suas estatísticas provavelmente estão desatualizadas. Para soluções recomendadas, consulte Estatísticas de tabela ausentes ou desatualizadas.
4. Verifique se há alguma operação de classificação de alto custo. Se houver, consulte Linhas não classificadas ou mal classificadas para as soluções recomendadas.
5. Procure os seguintes operadores de transmissão onde há operações de alto custo:
   - DS_BCAST_INNER: indica que a tabela é transmitida a todos os nós de computação. Isso é adequado para uma tabela pequena, mas não é ideal para uma tabela maior.
   - DS_DIST_ALL_INNER: Indica que toda o workload está em uma única fatia.
   - DS_DIST_BOTH: Indica uma redistribuição pesada.

   Para as soluções recomendadas para essas situações, consulte Distribuição de dados pouco satisfatória.

Analisar o resumo da consulta

Para obter as etapas de exclusão e estatísticas com mais detalhes do que no plano de consulta que EXPLAIN produz, use as exibições de sistema SVL_QUERY_SUMMARY e SVL_QUERY_REPORT.

SVL_QUERY_SUMMARY fornece estatísticas da consulta por fluxo. Você pode usar as informações fornecidas para identificar problemas com etapas caras, etapas longas e etapas que gravam no disco.

Com a visualização do sistema SVL_QUERY_REPORT, é possível ver informações semelhantes às de SVL_QUERY_SUMMARY, mas organizadas por fatia do nó de computação, e não por fluxo. Você pode usar informações do nível de fatia para detectar a distribuição desigual de dados no cluster (também conhecida como desvio da distribuição de dados), o que força alguns nós a trabalhar mais do que outros e prejudica a performance da consulta.

Tópicos

- Usar a visualização SVL_QUERY_SUMMARY
- como usar a visualização SVL_QUERY_REPORT
- Mapeamento do plano de consulta para o resumo da consulta

Usar a visualização SVL_QUERY_SUMMARY

Para analisar as informações de resumo de consulta por fluxo usando SVL_QUERY_SUMMARY, faça o seguinte:

1. Execute a seguinte consulta para determinar o ID de sua consulta:

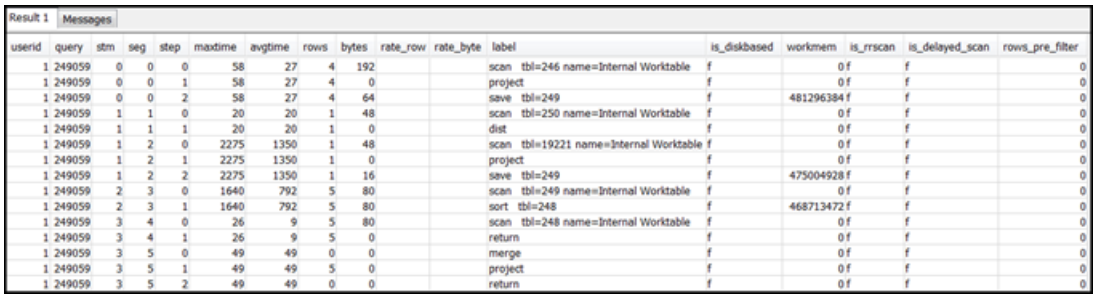
```
select query, elapsed, substring
from svl_qlog
order by query
desc limit 5;
```

   Examine o texto truncado da consulta no campo `substring` para determinar qual valor de `query` representa a sua consulta. Se você executou a consulta mais de uma vez, use o valor de `query` da linha com o menor valor `elapsed`. Esta é a linha para a versão compilada. Se você estiver executando várias consultas, poderá aumentar o valor usado pela cláusula LIMIT usada para certificar-se de que sua consulta seja incluída.

2. Selecione linhas do SVL_QUERY_SUMMARY para sua consulta. Classifique os resultados por fluxo, segmento e etapa:

```
select * from svl_query_summary where query = MyQueryID order by stm, seg, step;
```

   Este é um resultado de exemplo.



Result 1 | Messages

userid	query	stm	seg	step	maxtime	avgtime	rows	bytes	rate_row	rate_byte	label	is_diskbased	workmem	is_rrscan	is_delayed_scan	rows_pre_filter
1	249059	0	0	0	58	27	4	192			scan tbl=246 name=Internal Worktable	f	0	f	f	0
1	249059	0	0	1	58	27	4	0			project	f	0	f	f	0
1	249059	0	0	2	58	27	4	64			save tbl=249	f	481296384	f	f	0
1	249059	1	1	0	20	20	1	48			scan tbl=250 name=Internal Worktable	f	0	f	f	0
1	249059	1	1	1	20	20	1	0			dist	f	0	f	f	0
1	249059	1	2	0	2275	1350	1	48			scan tbl=19221 name=Internal Worktable	f	0	f	f	0
1	249059	1	2	1	2275	1350	1	0			project	f	0	f	f	0
1	249059	1	2	2	2275	1350	1	16			save tbl=249	f	475004928	f	f	0
1	249059	2	3	0	1640	792	5	80			scan tbl=249 name=Internal Worktable	f	0	f	f	0
1	249059	2	3	1	1640	792	5	80			sort tbl=248	f	468713472	f	f	0
1	249059	3	4	0	26	9	5	80			scan tbl=248 name=Internal Worktable	f	0	f	f	0
1	249059	3	4	1	26	9	5	0			return	f	0	f	f	0
1	249059	3	5	0	49	49	0	0			merge	f	0	f	f	0
1	249059	3	5	1	49	49	5	0			project	f	0	f	f	0
1	249059	3	5	2	49	49	0	0			return	f	0	f	f	0

3. Mapeie as etapas às operações de consulta no plano de consulta usando as informações em Mapeamento do plano de consulta para o resumo da consulta. Elas devem ter aproximadamente os mesmos valores para linhas e bytes (linhas * largura do plano de consulta). Se elas não tiverem, consulte Estatísticas de tabela ausentes ou desatualizadas para as soluções recomendadas.
4. Verifique se o campo `is_diskbased` tem um valor de `t` (verdadeiro) para qualquer etapa. Os hash, agregados e classificações são operadores que provavelmente irão gravar dados em disco se o sistema não tiver memória suficiente alocada para o processamento de consulta.

   Se `is_diskbased` for verdadeiro, consulte Memória insuficiente alocada para a consulta para as soluções recomendadas.
5. Revise os valores do campo `label` e verifique se há uma sequência AGG-DIST-AGG em qualquer lugar das etapas. Sua presença indica a agregação em duas etapas, que é cara. Para corrigir isso, altere a cláusula GROUP BY para usar a chave de distribuição (a primeira chave, se houver várias).
6. Revise o valor de `maxtime` para cada segmento (ele é o mesmo para todas as etapas no segmento). Identifique um segmento com o maior valor de `maxtime` e analise as etapas neste segmento quanto aos operadores a seguir.

> **Note**
>
> Um valor `maxtime` alto não necessariamente indica um problema com o segmento. Independente de um valor alto, o segmento pode não ter levado muito tempo para processar. Todos os segmentos em um fluxo começam a ser cronometrados

simultaneamente. No entanto, alguns segmentos downstream podem não ser executados até obterem dados dos segmentos upstream. Este efeito pode fazer parecer que eles tenham levado muito tempo, pois seus valores de `maxtime` incluirão tanto seu tempo de espera quanto o tempo de processamento.

- BCAST ou DIST: nesses casos, o valor `maxtime` alto pode ser o resultado da redistribuição de um grande número linhas. Para soluções recomendadas, consulte Distribuição de dados pouco satisfatória.
- HJOIN (junção hash): Se a etapa em questão tem um valor muito elevado no campo de `rows` em comparação ao valor de `rows` na etapa final do RETURN da consulta, consulte Junção de hash para as soluções recomendadas.
- SCAN/SORT: procura uma sequência de etapas SCAN, SORT, SCAN, MERGE imediatamente antes de uma etapa de junção. Este padrão indica que os dados não classificados estão sendo varridos e, então, mesclados com a área classificada da tabela.

  Verifique se o valor de linhas para a etapa SCAN tem um valor muito alto em comparação ao valor de linhas na etapa final de RETURN da consulta. Este padrão indica que o mecanismo de execução está fazendo a varredura de linhas que serão posteriormente rejeitadas, o que é ineficiente. Para soluções recomendadas, consulte Predicado insuficientemente restritivo.

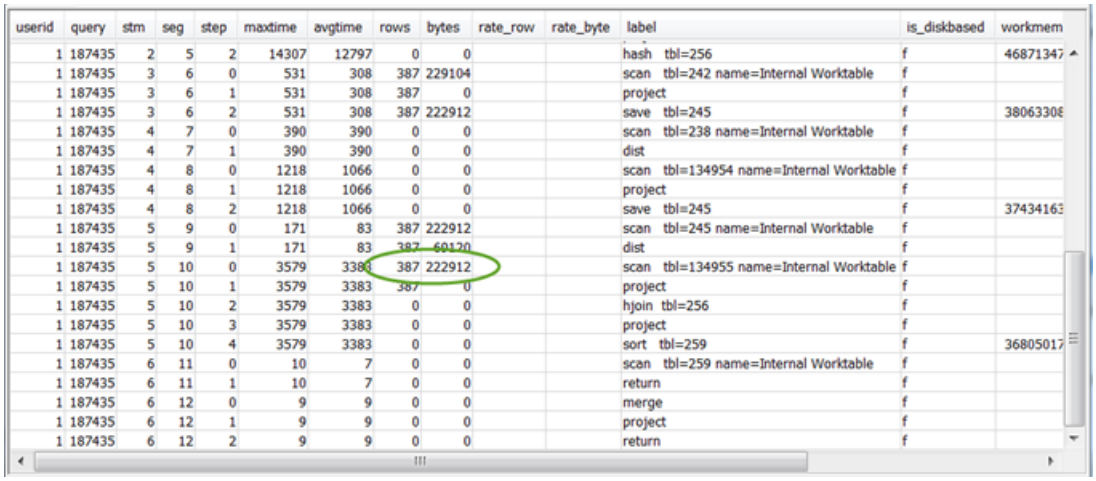
  Se o valor de `maxtime` para etapa SCAN é alto, consulte Cláusula WHERE pouco satisfatória para as soluções recomendadas.

  Se o valor de `rows` para a etapa SORT não for zero, consulte Linhas não classificadas ou mal classificadas para as soluções recomendadas.

7. Revise os valores de `rows` e `bytes` para as etapas de 5 a 10 que precedem a etapa final RETURN para ter uma ideia da quantidade de dados que foi retornada ao cliente. Esse processo pode ser um tipo de arte.

   Por exemplo, no resumo de consulta de amostra a seguir, a terceira etapa de PROJECT fornece um valor de `rows`, mas não um valor de `bytes`. Ao examinarmos as etapas anteriores para encontrar uma etapa com o mesmo valor de `rows`, encontramos a etapa SCAN, que fornece informações tanto de linha quanto de bytes:

   Veja a seguir um resultado de exemplo.



userid	query	stm	seg	step	maxtime	avgtime	rows	bytes	rate_row	rate_byte	label	is_diskbased	workmem
1	187435	2	5	2	14307	12797	0	0			hash tbl=256	f	46871347
1	187435	3	6	0	531	308	387	229104			scan tbl=242 name=Internal Worktable	f	
1	187435	3	6	1	531	308	387	0			project	f	
1	187435	3	6	2	531	308	387	222912			save tbl=245	f	38063308
1	187435	4	7	0	390	390	0	0			scan tbl=238 name=Internal Worktable	f	
1	187435	4	7	1	390	390	0	0			dist	f	
1	187435	4	8	0	1218	1066	0	0			scan tbl=134954 name=Internal Worktable	f	
1	187435	4	8	1	1218	1066	0	0			project	f	
1	187435	4	8	2	1218	1066	0	0			save tbl=245	f	37434163
1	187435	5	9	0	171	83	387	222912			scan tbl=245 name=Internal Worktable	f	
1	187435	5	9	1	171	83	387	60120			dist	f	
1	187435	5	10	0	3579	3383	387	222912			scan tbl=134955 name=Internal Worktable	f	
1	187435	5	10	1	3579	3383	387	0			project	f	
1	187435	5	10	2	3579	3383	0	0			hjoin tbl=256	f	
1	187435	5	10	3	3579	3383	0	0			project	f	
1	187435	5	10	4	3579	3383	0	0			sort tbl=259	f	36805017
1	187435	6	11	0	10	7	0	0			scan tbl=259 name=Internal Worktable	f	
1	187435	6	11	1	10	7	0	0			return	f	
1	187435	6	12	0	9	9	0	0			merge	f	
1	187435	6	12	1	9	9	0	0			project	f	
1	187435	6	12	2	9	9	0	0			return	f	

Se você estiver retornando um volume de dados inusitadamente alto, consulte Conjunto de resultados muito grande para as soluções recomendadas.

8. Verifique se o valor de `bytes` é alto em relação ao valor de `rows` para qualquer etapa, em comparação a outras etapas. Este padrão pode indicar que você está selecionando muitas colunas. Para soluções recomendadas, consulte Lista SELECT grande.

como usar a visualização SVL_QUERY_REPORT

Para analisar as informações de resumo de consulta por fatia usando SVL_QUERY_REPORT, faça o seguinte:

1. Execute o seguinte para determinar o ID de sua consulta:

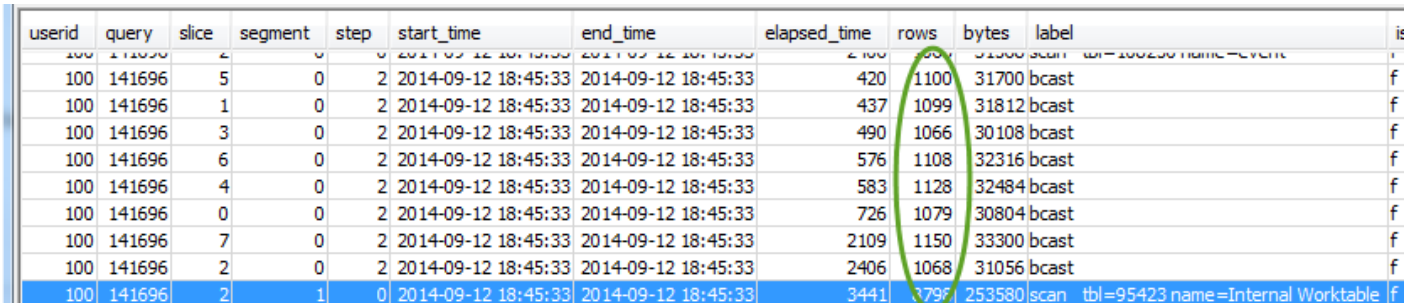
```
select query, elapsed, substring
from svl_qlog
order by query
desc limit 5;
```

Examine o texto truncado da consulta no campo `substring` para determinar qual valor de `query` representa a sua consulta. Se você executou a consulta mais de uma vez, use o valor de `query` da linha com o menor valor `elapsed`. Esta é a linha para a versão compilada. Se você estiver executando várias consultas, poderá aumentar o valor usado pela cláusula LIMIT usada para certificar-se de que sua consulta seja incluída.

2. Selecione linhas do SVL_QUERY_REPORT para sua consulta. Classifique os resultados por segmento, etapa, tempo decorrido e linhas:

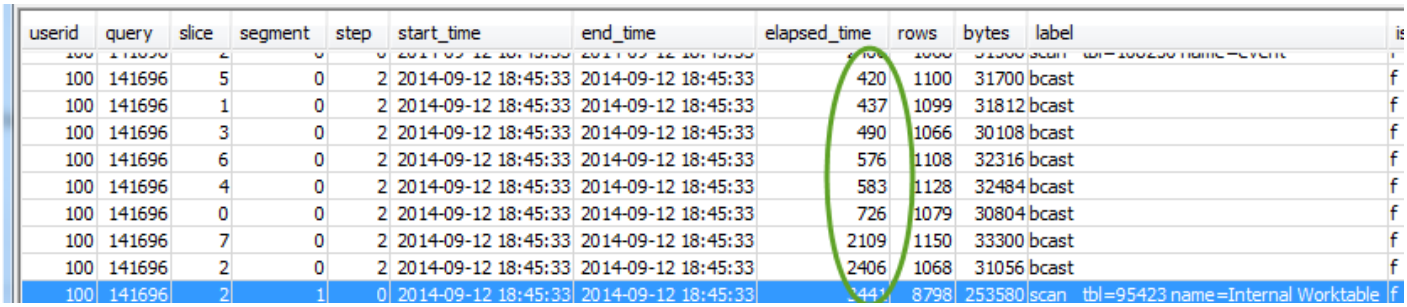
```
select * from svl_query_report where query = MyQueryID order by segment, step,
 elapsed_time, rows;
```

3. Para cada etapa, verifique se todas as fatias estão processando aproximadamente o mesmo número de linhas:



userid	query	slice	segment	step	start_time	end_time	elapsed_time	rows	bytes	label	is
100	141696	5	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	420	1100	31700	bcast	f
100	141696	1	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	437	1099	31812	bcast	f
100	141696	3	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	490	1066	30108	bcast	f
100	141696	6	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	576	1108	32316	bcast	f
100	141696	4	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	583	1128	32484	bcast	f
100	141696	0	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	726	1079	30804	bcast	f
100	141696	7	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	2109	1150	33300	bcast	f
100	141696	2	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	2406	1068	31056	bcast	f
100	141696	2	1	0	2014-09-12 18:45:33	2014-09-12 18:45:33	3441	8798	253580	scan tbl=95423 name=Internal Worktable	f

Verifique também se todas as fatias estão utilizando, aproximadamente, a mesma quantidade de tempo:



userid	query	slice	segment	step	start_time	end_time	elapsed_time	rows	bytes	label	is
100	141696	5	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	420	1100	31700	bcast	f
100	141696	1	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	437	1099	31812	bcast	f
100	141696	3	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	490	1066	30108	bcast	f
100	141696	6	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	576	1108	32316	bcast	f
100	141696	4	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	583	1128	32484	bcast	f
100	141696	0	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	726	1079	30804	bcast	f
100	141696	7	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	2109	1150	33300	bcast	f
100	141696	2	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	2406	1068	31056	bcast	f
100	141696	2	1	0	2014-09-12 18:45:33	2014-09-12 18:45:33	3441	8798	253580	scan tbl=95423 name=Internal Worktable	f

Grandes discrepâncias nesses valores podem indicar desvio de distribuição de dados em decorrência de um estilo menos satisfatório de distribuição para esta consulta específica. Para soluções recomendadas, consulte Distribuição de dados pouco satisfatória.

Mapeamento do plano de consulta para o resumo da consulta

Por meio da análise do resumo da consulta, é possível obter detalhes adicionais correlacionando as operações do plano de consulta às etapas (identificadas pelos valores com campo de rótulo) no resumo da consulta. A tabela a seguir correlaciona as operações do plano de consulta às etapas do resumo da consulta.

Operação do plano de consulta	Valor do campo de rótulo	Descrição
Aggregate HashAggregate GroupAggregate	AGGR	Avalia funções agregadas e condições GROUP BY.
DS_BCAST_INNER	BCAST (transmissão)	Transmite uma tabela completa ou alguns conjuntos de linhas (tais como um conjunto filtrado de linhas de uma tabela) para todos os nós.
Não aparece no plano de consulta	DELETE	Exclui linhas de tabelas.
DS_DIST_NONE DS_DIST_ALL_NONE DS_DIST_INNER DS_DIST_ALL_INNER DS_DIST_ALL_BOTH	DIST (distribuir)	Distribui linhas aos nós para fins de junção paralela ou outros processamentos paralelos.
HASH	HASH	Cria a tabela hash para uso em junções hash.
Hash Join	HJOIN (junção hash)	Realiza uma junção hash de duas tabelas ou conjuntos intermediários de resultados.
Não aparece no plano de consulta	INSERT	Insere linhas em tabelas.

Operação do plano de consulta	Valor do campo de rótulo	Descrição
Limite	LIMIT	Aplica uma cláusula LIMIT aos conjuntos de resultados.
Merge	MERGE	Mescla linhas derivadas de operações paralelas de classificação ou junção.
Merge Join	MJOIN (junção de mesclagem)	Realiza uma junção de mesclagem de duas tabelas ou conjuntos intermediários de resultados.
Loop aninhado	NLOOP (loop aninhado)	Realiza uma junção de loop aninhado de duas tabelas ou conjuntos intermediários de resultados.
Não aparece no plano de consulta	PARSE	Analisa strings em valores binários para carregamento.
Projeto	PROJECT	Avalia expressões.
Rede	RETURN	Retorna linhas ao principal ou ao cliente.
Não aparece no plano de consulta	SAVE	Materializa linhas para uso na próxima etapa de processamento.
Seq Scan	SCAN	Faz a varredura de tabelas ou conjuntos intermediários de resultados.

Operação do plano de consulta	Valor do campo de rótulo	Descrição
Sort	SORT	Classifica linhas ou conjuntos intermediários de resultado s conforme exigido pelas operações subsequentes (tais como junções ou agregaçõe s) ou para satisfazer uma cláusula ORDER BY.
Unique	UNIQUE	Aplica uma cláusula SELECT DISTINCT ou elimina duplicida des conforme exigido por outras operações.
Window	WINDOW	Computa funções agregadas e funções da janela de classific ação.

Aprimoramento do desempenho das consultas

A seguir são apresentados alguns problemas comuns que afetam o desempenho de consulta do Amazon Redshift e instruções sobre como diagnosticá-los e resolvê-los.

Tópicos

- Estatísticas de tabela ausentes ou desatualizadas
- Loop aninhado
- Junção de hash
- Linhas fantasmas ou linhas não confirmadas
- Linhas não classificadas ou mal classificadas
- Distribuição de dados pouco satisfatória
- Memória insuficiente alocada para a consulta
- Cláusula WHERE pouco satisfatória
- Predicado insuficientemente restritivo

- Conjunto de resultados muito grande
- Lista SELECT grande

Estatísticas de tabela ausentes ou desatualizadas

Se estatísticas da tabela estiverem ausentes ou desatualizadas, você pode ver o seguinte:

- Uma mensagem de advertência nos resultados do comando EXPLAIN.
- Um evento de alerta de estatísticas ausentes em STL_ALERT_EVENT_LOG. Para obter mais informações, consulte Revisar alertas da consulta.

Para corrigir esse problema, execute ANALYZE.

Loop aninhado

Se um loop aninhado estiver presente, você poderá ver um evento de alerta de loop aninhado em STL_ALERT_EVENT_LOG. Você também pode identificar esse tipo de evento ao executar a consulta em Como identificar consultas com loops aninhados. Para obter mais informações, consulte Revisar alertas da consulta.

Para corrigir isso, revise sua consulta para junções cruzadas e remova-as, se possível. Junções cruzadas sem uma condição de junção que resulte no produto cartesiano de duas tabelas. São tipicamente executadas como junções de loops aninhados, que são os mais baixos dos possíveis tipos de junção.

Junção de hash

Se uma junção hash estiver presente, você poderá ver o seguinte:

- Operações de hash e de junções hash no plano de consulta. Para obter mais informações, consulte Analisar o plano de consulta.
- Uma etapa HJOIN no segmento com o valor mais alto de maxtime em SVL_QUERY_SUMMARY. Para obter mais informações, consulte Usar a visualização SVL_QUERY_SUMMARY.

Para corrigir esse problema, há duas opções de abordagem:

- Reescreva a consulta para usar uma junção de mesclagem, se possível. Você pode fazer isso especificando as colunas de junção que são tanto chaves de distribuição quanto chaves de classificação.
- Se uma etapa HJOIN em SVL_QUERY_SUMMARY tiver um valor muito elevado no campo de linhas comparado ao valor de linhas na etapa final de RETURN da consulta, verifique se você pode reescrever a consulta para junção em uma coluna exclusiva. Quando a junção de uma consulta não ocorre em uma coluna exclusiva, tal como a chave primária que aumenta o número de linhas envolvidas na junção.

Linhas fantasmas ou linhas não confirmadas

Se linhas fantasmas ou linhas não confirmadas estiverem presentes, você poderá ver um evento de alerta em STL_ALERT_EVENT_LOG que indicará linhas fantasmas excessivas. Para obter mais informações, consulte Revisar alertas da consulta.

Para corrigir esse problema, há duas opções de abordagem:

- Verifique a guia Carregamentos do console do Amazon Redshift para operações de carregamento ativo em qualquer uma das tabelas de consulta. Se você vir operações de carregamento ativas, aguarde a conclusão destas operações antes de realizar uma ação.
- Se não houver operações de carregamento ativas, execute VACUUM nas tabelas da consulta para remover as linhas excluídas.

Linhas não classificadas ou mal classificadas

Se linhas não classificadas ou mal classificadas estiverem presentes, você poderá ver um evento alerta de filtro muito seletivo em STL_ALERT_EVENT_LOG. Para obter mais informações, consulte Revisar alertas da consulta.

Você também pode verificar se uma das tabelas em sua consulta tem grandes áreas não classificadas ao executar a consulta em Identificar tabelas com desvio de dados ou linhas não classificadas.

Para corrigir esse problema, há duas opções de abordagem:

- Execute VACUUM nas tabelas da consulta para reclassificar as linhas.
- Revise as chaves de classificação nas tabelas da consulta para ver se melhorias podem ser feitas. Lembre-se de considerar a performance desta consulta em relação à performance de outras

consultas importantes, além do sistema de forma geral antes de fazer quaisquer alterações. Para obter mais informações, consulte Chaves de classificação.

Distribuição de dados pouco satisfatória

Se a distribuição de dados for pouco satisfatória, você poderá ver o seguinte:

- Uma execução em série, uma grande transmissão ou um evento de alerta de grande distribuição aparece em STL_ALERT_EVENT_LOG. Para obter mais informações, consulte Revisar alertas da consulta.
- Fatias não estão processando, aproximadamente, o mesmo número de linhas para uma determinada etapa. Para obter mais informações, consulte como usar a visualização SVL_QUERY_REPORT.
- Fatias não estão utilizando, aproximadamente, a mesma quantidade de tempo para determinada etapa. Para obter mais informações, consulte como usar a visualização SVL_QUERY_REPORT.

Se nenhuma das condições anteriores for verdadeira, você também poderá ver se alguma das tabelas em sua consulta possui desvio de dados ao executar a consulta em Identificar tabelas com desvio de dados ou linhas não classificadas.

Para corrigir esse problema, analise os estilos de distribuição das tabelas da consulta e veja se é possível realizar alguma melhoria. Lembre-se de considerar a performance desta consulta em relação à performance de outras consultas importantes, além do sistema de forma geral antes de fazer quaisquer alterações. Para obter mais informações, consulte Distribuição de dados para otimização de consultas.

Memória insuficiente alocada para a consulta

Quando uma consulta precisa de mais memória do que a alocada, o Amazon Redshift grava os resultados intermediários em excesso no disco. Isso é chamado de gravação temporária em disco. O Amazon Redshift grava primeiro os dados vazados no disco local e, se excederem a capacidade do disco local, os dados serão transferidos para o Amazon S3. As etapas de consulta com maior probabilidade de gravação temporária são junções de hash, agregações (como GROUP BY e DISTINCT), funções de janela e classificações (ORDER BY) em grandes conjuntos de dados.

A gravação temporária em disco reduz a velocidade da consulta, pois o acesso ao disco é muito mais lento do que a memória. Como o disco local e a largura de banda de E/S são compartilhados em todo o cluster, o excesso de informações também pode atrasar outras consultas em execução

ao mesmo tempo. A redução da gravação temporária melhora a latência de consultas individuais e o throughput geral do cluster.

Identifique as consultas que vazam. Consulte SYS_QUERY_DETAIL e revise as colunas `spilled_block_remote_disk` e `spilled_block_local_disk`. As consultas com alto valor de `spilled_block_remote_disk` estão indo além do disco local e são as que mais se beneficiam com o ajuste. Em um cluster provisionado, você também pode revisar a métrica `query_temp_blocks_to_disk` em SVL_QUERY_METRICS_SUMMARY.

Como reduzir gravação temporária em disco

- Reduza o volume de dados intermediários. Filtre mais cedo, selecione somente as colunas necessárias e pré-agregue sempre que possível para que menos linhas fluam em junções, agregações e classificações.
- Divida consultas grandes em etapas. Materialize os resultados intermediários em tabelas temporárias (CREATE TEMP TABLE AS) para que cada etapa seja executada com um espaço menor na memória e tenha menos probabilidade de vazar. Forneça à tabela temporária uma distribuição e uma chave de classificação apropriadas e execute ANALYZE para que as etapas posteriores sejam executadas com eficiência.
- Forneça mais memória à consulta aumentando temporariamente sua contagem de slots se você usar o WLM manual (consulte wlm_query_slot_count).
- Adicione nós ou mude para um tipo de nó maior quando o vazamento for inevitável em seu volume de dados.

Monitore a gravação temporária em disco. Crie uma regra de monitoramento de consultas (QMR) na métrica `query_temp_blocks_to_disk`. Essa métrica mede o espaço em disco temporário usado para resultados intermediários, em blocos de 1 MB. Depois, selecione uma ação:

log

Registra a consulta para análise.

abort

Interrompe a consulta antes que a gravação temporária em disco degrade outras workloads.

Por exemplo, `query_temp_blocks_to_disk > 100000` (cerca de 100 GB) registra em log ou cancela consultas que geram quantidades excepcionalmente grandes. Ajuste o limite para o cluster

ou grupo de trabalho. Em um cluster provisionado, configure o QMR por meio de suas filas do WLM. No Redshift sem servidor, configure-o por meio dos limites de consulta do grupo de trabalho. Para obter mais informações, consulte Regras de monitoramento de consulta do WLM.

Cláusula WHERE pouco satisfatória

Se sua cláusula WHERE causar varreduras excessivas da tabela, talvez você veja uma etapa SCAN no segmento com o maior valor `maxtime` em SVL_QUERY_SUMMARY. Para obter mais informações, consulte Usar a visualização SVL_QUERY_SUMMARY.

Para corrigir esse problema, adicione uma cláusula WHERE à consulta com base na coluna de classificação principal da maior tabela. Essa abordagem ajuda a minimizar o tempo de varredura. Para obter mais informações, consulte Práticas recomendadas do Amazon Redshift para projetar tabelas.

Predicado insuficientemente restritivo

Se sua consulta tem um predicado insuficientemente restritivo, você pode ver uma etapa SCAN no segmento com o maior valor `maxtime` em SVL_QUERY_SUMMARY que tem um valor de `rows` muito alto comparado ao valor de `rows` na etapa final de RETURN da consulta. Para obter mais informações, consulte Usar a visualização SVL_QUERY_SUMMARY.

Para corrigir esse problema, tente adicionar um predicado à consulta ou tornar o predicado existente mais restritivo para restringir a saída.

Conjunto de resultados muito grande

Se sua consulta retornar um conjunto de resultados muito grande, considere reescrever a consulta para usar UNLOAD para gravar os resultados no Amazon S3. Essa abordagem melhorará a performance da etapa final de RETURN aproveitando o processamento paralelo. Para obter mais informações sobre a verificação da existência de um conjunto muito grande de resultados, consulte Usar a visualização SVL_QUERY_SUMMARY.

Lista SELECT grande

Se sua consulta tiver uma lista SELECT inesperadamente grande, você poderá ver um valor `bytes` alto em relação ao valor de `rows` para qualquer etapa (em comparação às outras etapas) em SVL_QUERY_SUMMARY. Este valor `bytes` alto pode ser um indicador de que você está selecionando muitas colunas. Para obter mais informações, consulte Usar a visualização SVL_QUERY_SUMMARY.

Para corrigir esse problema, revise as colunas que você está selecionando e verifique se alguma pode ser removida.

Consultas de diagnóstico para ajuste da consulta

Use as consultas a seguir para identificar problemas com consultas ou tabelas subjacentes que possam afetar a performance da consulta. Recomendamos o uso dessas consultas com os processos de ajuste de consulta abordados em Análise e aprimoramento de consultas.

> **Note**
>
> Essas consultas são sobre clusters provisionados do Amazon Redshift. Essas consultas não devem ser usadas com grupos de trabalho do Redshift sem servidor.

Tópicos

- Identificar consultas que são os principais candidatos para ajuste
- Identificar tabelas com desvio de dados ou linhas não classificadas
- Como identificar consultas com loops aninhados
- Como revisar os tempos de espera na fila para consultas
- Como revisar alertas de consulta por tabela
- Como identificar tabelas com estatísticas ausentes

Identificar consultas que são os principais candidatos para ajuste

A consulta a seguir identifica as 50 instruções com maior consumo de tempo que tenham sido executadas nos últimos 7 dias. Você pode usar os resultados para identificar consultas que estão demandando um tempo além do normal. Você também pode identificar consultas executadas com frequência (as que aparecem mais de uma vez no conjunto de resultados). Essas consultas frequentemente são boas candidatas para ajustar o aprimoramento da performance do sistema.

Essa consulta também fornece uma contagem dos eventos de alerta associados a cada consulta identificada. Esses alertas fornecem detalhes que você pode usar para melhorar a performance da consulta. Para obter mais informações, consulte Revisar alertas da consulta.

```
select trim(database) as db, count(query) as n_qry,
max(substring (qrytext,1,80)) as qrytext,
```

```
min(run_minutes) as "min" ,
max(run_minutes) as "max",
avg(run_minutes) as "avg", sum(run_minutes) as total,
max(query) as max_query_id,
max(starttime)::date as last_run,
sum(alerts) as alerts, aborted
from (select userid, label, stl_query.query,
trim(database) as database,
trim(querytxt) as qrytext,
md5(trim(querytxt)) as qry_md5,
starttime, endtime,
(datediff(seconds, starttime,endtime)::numeric(12,2))/60 as run_minutes,
alrt.num_events as alerts, aborted
from stl_query
left outer join
(select query, 1 as num_events from stl_alert_event_log group by query ) as alrt
on alrt.query = stl_query.query
where userid <> 1 and starttime >=  dateadd(day, -7, current_date))
group by database, label, qry_md5, aborted
order by total desc limit 50;
```

Identificar tabelas com desvio de dados ou linhas não classificadas

A seguinte consulta identifica as tabelas que possuem uma distribuição desigual de dados (desvio de dados) ou uma porcentagem alta de linhas não classificadas.

Um valor `skew` baixo indica que os dados daquela tabela estão distribuídos adequadamente. Se uma tabela tem um valor `skew` de 4,00 ou mais, considere modificar seu estilo de distribuição de dados. Para obter mais informações, consulte Distribuição de dados pouco satisfatória.

Se uma tabela tem um valor `pct_unsorted` maior que 20 por cento, considere executar o comando VACUUM. Para obter mais informações, consulte Linhas não classificadas ou mal classificadas.

Você também deve revisar os valores `mbytes` e `pct_of_total` de cada tabela. Essas colunas identificam o tamanho da tabela e a porcentagem de espaço bruto em disco que a tabela utiliza. O espaço em disco bruto inclui o espaço reservado pelo Amazon Redshift para uso interno, portanto, é maior do que a capacidade nominal do disco, que é a quantidade de espaço em disco disponível para o usuário. Use essas informações para verificar se você tem espaço livre em disco igual a pelo menos 2,5 vezes o tamanho de sua maior tabela. Ter esse espaço disponível permite que o sistema grave resultados intermediários no disco ao processar consultas complexas.

```
select trim(pgn.nspname) as schema,
```

```
trim(a.name) as table, id as tableid,
decode(pgc.reldiststyle,0, 'even',1,det.distkey ,8,'all') as distkey,
 dist_ratio.ratio::decimal(10,4) as skew,
det.head_sort as "sortkey",
det.n_sortkeys as "#sks", b.mbytes,
decode(b.mbytes,0,0,((b.mbytes/part.total::decimal)*100)::decimal(5,2)) as
 pct_of_total,
decode(det.max_enc,0,'n','y') as enc, a.rows,
decode( det.n_sortkeys, 0, null, a.unsorted_rows ) as unsorted_rows ,
decode( det.n_sortkeys, 0, null, decode( a.rows,0,0, (a.unsorted_rows::decimal(32)/
a.rows)*100) )::decimal(5,2) as pct_unsorted
from (select db_id, id, name, sum(rows) as rows,
sum(rows)-sum(sorted_rows) as unsorted_rows
from stv_tbl_perm a
group by db_id, id, name) as a
join pg_class as pgc on pgc.oid = a.id
join pg_namespace as pgn on pgn.oid = pgc.relnamespace
left outer join (select tbl, count(*) as mbytes
from stv_blocklist group by tbl) b on a.id=b.tbl
inner join (select attrelid,
min(case attisdistkey when 't' then attname else null end) as "distkey",
min(case attsortkeyord when 1 then attname  else null end ) as head_sort ,
max(attsortkeyord) as n_sortkeys,
max(attencodingtype) as max_enc
from pg_attribute group by 1) as det
on det.attrelid = a.id
inner join ( select tbl, max(mbytes)::decimal(32)/min(mbytes) as ratio
from (select tbl, trim(name) as name, slice, count(*) as mbytes
from svv_diskusage group by tbl, name, slice )
group by tbl, name ) as dist_ratio on a.id = dist_ratio.tbl
join ( select sum(capacity) as  total
from stv_partitions where part_begin=0 ) as part on 1=1
where mbytes is not null
order by  mbytes desc;
```

Como identificar consultas com loops aninhados

A seguinte consulta identifica consultas que tiveram eventos de alerta registrados para loops aninhados. Para obter informações sobre como corrigir a condição de loop aninhado, consulte Loop aninhado.

```
select query, trim(querytxt) as SQL, starttime
from stl_query
```

```
where query in (
select distinct query
from stl_alert_event_log
where event like 'Nested Loop Join in the query plan%')
order by starttime desc;
```

Como revisar os tempos de espera na fila para consultas

A consulta a seguir mostra quanto tempo as consultas recentes esperaram por uma vaga disponível em uma fila de consulta antes da execução. Se você vir uma tendência de altos tempos de espera, poderá modificar sua configuração de fila de consulta para melhor rendimento. Para obter mais informações, consulte Implementar o WLM manual.

```
select trim(database) as DB , w.query,
substring(q.querytxt, 1, 100) as querytxt,  w.queue_start_time,
w.service_class as class, w.slot_count as slots,
w.total_queue_time/1000000 as queue_seconds,
w.total_exec_time/1000000 exec_seconds, (w.total_queue_time+w.total_Exec_time)/1000000
 as total_seconds
from stl_wlm_query w
left join stl_query q on q.query = w.query and q.userid = w.userid
where w.queue_start_Time >= dateadd(day, -7, current_Date)
and w.total_queue_Time > 0  and w.userid >1
and q.starttime >= dateadd(day, -7, current_Date)
order by w.total_queue_time desc, w.queue_start_time desc limit 35;
```

Como revisar alertas de consulta por tabela

A consulta a seguir identifica as tabelas que tiveram eventos de alerta registrados e também identifica que tipo de alerta surge com mais frequência.

Se o valor `minutes` para uma linha com uma tabela identificada for alto, verifique a tabela para ver se ela precisa de manutenção de rotina, tal como a execução de ANALYZE ou VACUUM nela.

Se o valor `count` for alto para uma linha mas o valor `table` for nulo, execute uma consulta em STL_ALERT_EVENT_LOG pelo valor `event` associado para investigar por que esse alerta está sendo ativado com mais frequência.

```
select trim(s.perm_table_name) as table,
(sum(abs(datediff(seconds, s.starttime, s.endtime)))/60)::numeric(24,0) as minutes,
 trim(split_part(l.event,':',1)) as event,  trim(l.solution) as solution,
```

```
max(l.query) as sample_query, count(*)
from stl_alert_event_log as l
left join stl_scan as s on s.query = l.query and s.slice = l.slice
and s.segment = l.segment and s.step = l.step
where l.event_time >=  dateadd(day, -7, current_Date)
group by 1,3,4
order by 2 desc,6 desc;
```

Como identificar tabelas com estatísticas ausentes

A consulta a seguir fornece uma contagem das consultas que você está executando em tabelas cujas estatísticas estão ausentes. Se essa consulta retornar qualquer linha, verifique o valor `plannode` para determinar a tabela afetada e, então, execute ANALYZE nela.

```
select substring(trim(plannode),1,100) as plannode, count(*)
from stl_explain
where plannode like '%missing statistics%'
group by plannode
order by 2 desc;
```

Solução de problemas de consulta

Esta seção fornece uma referência rápida para identificar e resolver alguns dos problemas mais comuns e mais graves que você provavelmente encontrará nas consultas do Amazon Redshift.

Tópicos

- Falhas na conexão
- Consultas travadas
- A consulta leva muito tempo
- Falha do carregamento
- O carregamento leva muito tempo
- Os dados de carregamento estão incorretos
- Como configurar o parâmetro JDBC para o tamanho da busca

Essas sugestões fornecem um ponto inicial para a solução de problemas. Você também pode consultar os recursos a seguir para obter informações mais detalhadas.

Para ter informações sobre mudanças de comportamento na funcionalidade do Amazon Redshift que podem afetar sua aplicação, consulte Mudanças de comportamento.

- Acessar clusters e bancos de dados do Amazon Redshift
- Otimização automática de tabelas
- Carregar dados no Amazon Redshift
- Tutorial: Carregar dados do Amazon S3

Falhas na conexão

A conexão de uma consulta pode falhar pelos motivos a seguir. Sugerimos as abordagens de solução de problemas a seguir.

O cliente não pode conectar-se ao servidor

Se você estiver usando o SSL ou certificados de servidor, remova essa complexidade antes de tentar solucionar o problema de conexão. Então, adicione certificados SSL ou de servidor novamente quanto você tiver encontrado uma solução. Para obter mais informações, acesse “Configurar as opções de segurança para conexões” no Guia de gerenciamento de clusters do Amazon Redshift.

A conexão é recusada

Geralmente, quando você recebe uma mensagem de erro indicando que houve falha em estabelecer uma conexão, isso significa que há um problema com a permissão para acessar o cluster. Para obter mais informações, acesse “A conexão é recusada ou falha” no Guia de gerenciamento de clusters do Amazon Redshift.

Consultas travadas

A consulta pode travar ou parar de responder pelos motivos a seguir. Sugerimos as abordagens de solução de problemas a seguir.

A conexão com o banco de dados é interrompida

Reduza o tamanho da unidade de transmissão máxima (Maximum Transmission Unit, MTU). O tamanho de MTU determina o tamanho máximo, em bytes, de um pacote que pode ser transferido

em um quadro Ethernet através de sua conexão de rede. Para obter mais informações, acesse “As consultas parecem travar e, às vezes, não se comunicam com o cluster” no Guia de gerenciamento de clusters do Amazon Redshift.

A conexão com o banco de dados atinge o tempo limite

A conexão do cliente com o banco de dados parece estar travada ou ter expirado por exceder o tempo limite ao executar consultas longas, como um comando COPY. Nesse caso, você pode observar que o console do Amazon Redshift exibe que a consulta foi concluída, mas a própria ferramenta do cliente ainda parece estar executando a consulta. Os resultados da consulta podem estar ausentes ou incompletos, dependendo de quando a conexão foi interrompida. Este efeito acontece quando conexões ociosas são encerradas por um componente intermediário de rede. Para obter mais informações, acesse “Conexão de fora do Amazon EC2 - problema de tempo limite do firewall” no Guia de gerenciamento de clusters do Amazon Redshift.

Erro de falta de memória no lado do cliente ocorre com ODBC

Se seu aplicativo cliente usa uma conexão ODBC e sua consulta cria um conjunto de resultados que é muito grande para caber na memória, você pode transmitir o conjunto de resultados para seu aplicativo cliente usando um cursor. Para obter mais informações, consulte DECLARE e Considerações sobre a performance ao usar cursores.

Erro de falta de memória no lado do cliente ocorre com JDBC

Quando você tenta recuperar grandes conjuntos de resultados através de uma conexão JDBC, pode encontrar erros de falta de memória por parte do cliente. Para obter mais informações, consulte Como configurar o parâmetro JDBC para o tamanho da busca.

Há um deadlock em potencial

Se houver um deadlock em potencial, tente o seguinte:

- Visualize as tabelas de sistema STV_LOCKS e STL_TR_CONFLICT para localizar conflitos envolvendo atualizações em mais de uma tabela.
- Use a função PG_CANCEL_BACKEND para cancelar uma ou mais consultas em conflito.
- Use a função PG_TERMINATE_BACKEND para encerrar uma sessão, o que força todas as transações atualmente em execução na sessão encerrada a liberar todas as travas e reverter a transação.
- Programe operações de gravação simultâneas cuidadosamente. Para obter mais informações, consulte Gerenciamento de operações de gravação simultâneas.

A consulta leva muito tempo

A consulta pode demorar muito pelos motivos a seguir. Sugerimos as abordagens de solução de problemas a seguir.

As tabelas não estão otimizadas

Defina a chave de classificação, estilo de distribuição e codificação de compactação das tabelas para aproveitar a vantagem completa do processamento paralelo. Para obter mais informações, consulte . Otimização automática de tabelas

A consulta esta gravando em disco

Suas consultas podem estar sendo gravadas em disco durante pelo menos parte da execução da consulta, o que torna a consulta mais lenta e pode afetar outras consultas no cluster. Para identificar consultas que são gravadas temporariamente em disco, consulte SYS_QUERY_DETAIL e revise as colunas `spilled_block_local_disk` e `spilled_block_remote_disk`. Para ter mais informações sobre a redução e o monitoramento da gravação temporária em disco, consulte Aprimoramento do desempenho das consultas.

A consulta deve esperar a conclusão de outras consultas

Você pode melhorar a performance geral do sistema criando filas de consultas e atribuindo tipos de consultas diferentes para as filas adequadas. Para obter mais informações, consulte Gerenciamento do workload.

As consultas não estão otimizadas

Analise o plano de explicação para encontrar oportunidades para reescrever consultas ou otimizar o banco de dados. Para obter mais informações, consulte Criar e interpretar um plano de consulta.

A consulta precisa de mais memória para execução

Se uma consulta específica precisa de mais memória, você pode ampliar a memória disponível aumentando o wlm_query_slot_count.

O banco de dados exige a execução de um comando VACUUM

Execute o comando VACUUM sempre que você adicionar, excluir ou modificar um grande número de linhas, a menos que carregue seus dados por ordem de chave de classificação. O comando VACUUM reorganiza seus dados para manter a ordem de classificação e restaurar a performance. Para obter mais informações, consulte Vacuum de tabelas.

Recursos adicionais para solucionar problemas com consultas de longa duração

Veja a seguir os tópicos de visualização do sistema e outras seções da documentação que são úteis para o ajuste de consultas:

- A visualização do sistema STV_INFLIGHT mostra quais consultas estão sendo executadas no cluster. Pode ser útil usá-la junto com STV_RECENTS para determinar quais consultas estão em execução ou foram concluídas recentemente.
- SYS_QUERY_HISTORY é útil para a solução de problemas. Ele mostra consultas DDL e DML com propriedades relevantes, como seu status atual (por exemplo, `running` ou `failed`), o tempo que cada uma levou para ser executada e se uma consulta foi executada em um cluster de escalabilidade simultânea.
- STL_QUERYTEXT captura o texto da consulta para os comandos SQL. Além disso, SVV_QUERY_INFLIGHT, que une STL_QUERYTEXT a STV_INFLIGHT, mostra mais metadados de consulta.
- Um conflito de bloqueio de transação pode ser uma possível fonte de problemas de performance da consulta. Para obter informações sobre transações que no momento mantêm bloqueios em tabelas, consulte SVV_TRANSACTIONS.
- Identificar consultas que são os principais candidatos para ajuste fornece uma consulta de solução de problemas que ajuda a determinar quais consultas executadas recentemente consumiram mais tempo. Isso pode ajudar você a concentrar seus esforços em questões que precisam ser melhoradas.
- Se você quiser explorar mais o gerenciamento de consultas e entender como gerenciar filas de consultas, Gerenciamento do workload mostra como fazer isso. O gerenciamento da workload é um atributo avançado, e recomendamos o gerenciamento automatizado da workload na maioria dos casos.

Falha do carregamento

O carregamento de dados pode falhar pelos motivos a seguir. Sugerimos as abordagens de solução de problemas a seguir.

A origem dos dados está em uma região d AWS diferente

Por padrão, o bucket do Amazon S3 ou a tabela Amazon DynamoDB especificada no comando COPY deve estar na mesma região da AWS do cluster. Se os seus dados e seu cluster estiverem em regiões diferentes, você receberá um erro similar ao seguinte:

```
The bucket you are attempting to access must be addressed using the specified endpoint.
```

Se possível, o cluster e a origem dos dados devem estar na mesma região. Você pode especificar uma região diferente usando a opção REGION com o comando COPY.

> **Note**
>
> Se o seu cluster e sua fonte de dados estiverem em regiões diferentes da AWS, você terá custos de transferência de dados. Você também terá maior latência.

Falha do comando COPY

Consulte STL_LOAD_ERRORS para descobrir erros que ocorreram durante os carregamentos específicos. Para obter mais informações, consulte STL_LOAD_ERRORS.

O carregamento leva muito tempo

A operação de carregamento pode demorar muito pelos motivos a seguir. Sugerimos as abordagens de solução de problemas a seguir.

COPY carrega dados de um único arquivo

Divida seus dados de carregamento em vários arquivos. Quando você carrega todos os dados de um único arquivo grande, o Amazon Redshift é forçado a executar um carregamento serializado, que é muito mais lento. O número de arquivos deve ser um múltiplo do número de fatias em seu cluster e os arquivos devem ter aproximadamente o mesmo tamanho, entre 1 MB e 1 GB após a compactação. Para obter mais informações, consulte Práticas recomendadas do Amazon Redshift para criar consultas.

A operação de carregamento usa vários comandos COPY

Se você usar vários comandos COPY simultâneos para carregar uma tabela de vários arquivos, o Amazon Redshift é forçado a executar um carregamento serializado, que é muito mais lento. Nesse caso, use um único comando COPY.

Os dados de carregamento estão incorretos

A operação COPY pode carregar dados incorretos das maneiras indicadas a seguir. Sugerimos as abordagens de solução de problemas a seguir.

Arquivos errados são carregados

O uso de um prefixo de objeto para especificar arquivos de dados pode causar a leitura de arquivos indesejados. Em vez disso, use um arquivo manifesto para especificar exatamente quais arquivos devem ser carregados. Para obter mais informações, consulte a opção copy_from_s3_manifest_file para o comando COPY e Example: COPY from Amazon S3 using a manifest nos exemplos de COPY.

Como configurar o parâmetro JDBC para o tamanho da busca

Por padrão, o driver JDBC do Redshift usa um buffer em anel para gerenciar a memória com eficiência e evitar erros de falta de memória. O parâmetro de tamanho da busca só é aplicável quando o buffer em anel está explicitamente desabilitado. Para acessar mais informações, consulte link. Nessa configuração, você deve definir o tamanho da busca para controlar quantas linhas são recuperadas em cada lote.

Quando usar o tamanho da busca?

Use o parâmetro de tamanho da busca quando:

- Você precisa de controle refinado sobre lotes baseados em linhas.
- Trabalhar com aplicações legadas que exigem um comportamento tradicional de tamanho da busca

Definir o tamanho da busca

Por padrão, quando o buffer em anel está desabilitado, o driver JDBC coleta todos os resultados de uma consulta de uma só vez. As consultas que exibem grandes conjuntos de resultados podem consumir memória excessiva. Para recuperar conjuntos de resultados em lote em vez de todos de uma vez, defina o parâmetro de tamanho da busca JDBC em sua aplicação.

> **Note**
>
> O tamanho de busca não é compatível com ODBC.

Para obter uma melhor performance, defina o tamanho de busca como o maior valor que não resulte em erros de falta de memória. Um valor menor de tamanho de busca resulta em mais viagens do

servidor, o que prolonga os tempos de execução. O servidor reserva recursos, incluindo a vaga de consulta WLM e memória associada, até que o cliente recupere todo o conjunto de resultados ou até que a consulta seja cancelada. Quando você ajusta o tamanho de busca adequadamente, esses recursos são liberados mais rapidamente, disponibilizando-os para outras consultas.

> **Note**
>
> Se você precisar extrair grandes conjuntos de dados, recomendamos o uso de uma instrução UNLOAD para transferir os dados ao Amazon S3. Quando você usa UNLOAD, os nós de computação funcionam em paralelo para acelerar a transferência de dados.

Para obter mais informações sobre a configuração do parâmetro JDBC para o tamanho de busca, acesse Obtenção de resultados com base em um cursor na documentação do PostgreSQL.

Gerenciamento do workload

Você pode configurar o WLM do Amazon Redshift para ser executado com WLM automático ou WLM manual.

Com o Amazon Redshift, é possível gerenciar e priorizar workloads de usuários e consultas simultâneas para otimizar o desempenho e a utilização de recursos. Com o gerenciamento de workloads (WLM), é possível definir filas, grupos de usuários e outras estruturas para controlar os recursos alocados a diferentes tipos de consultas ou usuários.

As seções a seguir descrevem os recursos específicos de gerenciamento de workloads no Amazon Redshift e oferecem orientações sobre como configurá-los e monitorá-los.

WLM automático

Para maximizar a taxa de transferência do sistema e usar os recursos de maneira eficaz, você pode habilitar o Amazon Redshift para gerenciar como os recursos são divididos para executar consultas simultâneas com WLM automático. O WLM automático gerencia os recursos necessários para executar consultas. O Amazon Redshift determina quantas consultas são executadas simultaneamente e quanta memória é alocada para cada consulta enviada. Use WLM automático quando você quiser que o Amazon Redshift gerencie como os recursos devem ser divididos para executar consultas simultâneas. Para obter mais informações, consulte Implementar o WLM automático.

Ao trabalhar com a escalabilidade da simultaneidade e com o WLM automático, é possível oferecer suporte a praticamente infinitos usuários e consultas simultâneos, com performance de consultas consistentemente rápida. Para obter mais informações, consulte Escalabilidade simultânea..

> **Note**
>
> Na maioria dos casos, recomendamos que você use o WLM automático. Se você estiver usando o WLM manual e quiser migrar para o WLM automático, consulte Migrar do WLM manual para o WLM automático.

Com o WLM automático, é possível definir prioridades de consulta para workloads em uma fila. Para obter mais informações sobre prioridade de consultas, acesse Prioridade da consulta.

WLM manual

Você poderá ter várias sessões ou usuários executando consultas ao mesmo tempo. Algumas consultas poderão consumir recursos de cluster por períodos longos e afetar a performance de outras. O WLM manual pode ajudar a gerenciar isso para casos de uso especializados. Use WLM manual quando quiser ter maior controle sobre a simultaneidade.

É possível gerenciar o desempenho do sistema modificando a configuração do WLM a fim de criar filas separadas para as consultas demoradas e as rápidas. No tempo de execução, é possível rotear consultas para essas filas de acordo com grupos de usuários ou de consultas.

Você pode configurar regras a fim de rotear consultas para filas em especial com base no usuário que executa a consulta ou nos rótulos especificados por você. Você também pode configurar o valor de memória alocada para cada fila, de maneira que consultas grandes sejam executadas em filas com mais memória do que outras filas. Você também pode configurar uma regra de monitoramento de consulta (QMR) para limitar as consultas de longa execução. Para obter mais informações, consulte Implementar o WLM manual.

> **Note**
>
> Recomendamos a configuração de suas filas de consultas de WLM manual com um total de 15 slots de consulta ou menos. Para obter mais informações, consulte Nível de simultaneidade.

Observe que em uma configuração de WLM manual, a quantidade máxima de slots que você pode alocar para uma fila é 50. No entanto, isso não significa que um cluster do Amazon Redshift sempre execute 50 consultas simultaneamente em uma configuração de WLM automático. Isso pode mudar com base nas necessidades de memória ou em outros tipos de alocação de recursos no cluster.

Tópicos

- Alternar o modo do WLM
- Modificar a configuração do WLM
- Implementar o WLM automático
- Implementar o WLM manual
- Escalabilidade simultânea.
- Aceleração de consulta breve
- Regras de atribuição de fila do WLM
- Atribuir consultas a filas

- Propriedades de configuração dinâmicas e estáticas do WLM
- Regras de monitoramento de consulta do WLM
- Tabelas de sistema e visualizações do WLM

Alternar o modo do WLM

É possível habilitar o WLM automático ou manual usando o console do Amazon Redshift:

1. Escolha Switch WLM mode (Alternar modo de WLM).
2. Para configurá-lo como WLM automático, escolha WLM automático. Com essa opção, até oito filas são usadas para gerenciar consultas e os campos Memory (Memória) e Concurrency on main (Simultaneidade no principal) são definidos como auto (automático). Além disso, a prioridade padrão das consultas está definida como Normal.
3. Para habilitar a configuração manual usando o console do Amazon Redshift, mude para o WLM manual. Com essa opção, especifique as filas usadas para gerenciar consultas e os valores dos campos Memory (Memória) e Concurrency on main (Simultaneidade no principal). Com uma configuração manual, é possível configurar até oito filas de consultas e definir o número de consultas que pode ser executado em cada uma dessas filas simultaneamente.

Modificar a configuração do WLM

A maneira mais fácil de modificar a configuração do WLM é usando o console do Amazon Redshift. Você também pode usar a AWS CLI ou a API do Amazon Redshift.

Ao alternar o cluster entre WLM automático e manual, ele é colocado no estado `pending reboot`. A alteração não entra em vigor até que o próximo cluster seja reinicializado.

Para obter informações detalhadas sobre como modificar as configurações do WLM, consulte “Configurar o gerenciamento de workload” no Guia de gerenciamento de clusters do Amazon Redshift.

Migrar do WLM manual para o WLM automático

Para maximizar a taxa de transferência do sistema e usar os recursos de maneira mais efetiva, recomendamos que você configure o WLM automático para as filas. Considere adotar a abordagem a seguir para configurar uma transição tranquila do WLM manual para o WLM automático.

Para migrar de WLM manual para WLM automático e usar prioridades de fila, recomendamos que você crie um grupo de parâmetros e anexe-o ao cluster. Para obter mais informações, consulte "Grupos de parâmetros do Amazon Redshift" no Guia de gerenciamento de clusters do Amazon Redshift.

> **Important**
>
> Alterar o grupo de parâmetros ou mudar de WLM manual para automático exige uma reinicialização do cluster. Para obter mais informações, consulte Propriedades de configuração dinâmicas e estáticas do WLM.

Vamos ver um exemplo no qual há três filas manuais do WLM. Uma para uma workload de ETL, outra para uma workload de análise e mais uma para uma workload de ciência de dados. O workload de ETL é executado a cada 6 horas, o workload de análise é executado durante todo o dia e o workload de ciência de dados pode apresentar períodos de pico a qualquer momento. Com o WLM manual, especifique a memória e a simultaneidade que cada fila de workload obtém com base na sua compreensão da importância de cada uma delas para os negócios. A especificação da memória e da simultaneidade não é só difícil de descobrir, mas também faz com que os recursos do cluster sejam particionados estaticamente e, portanto, sejam desperdiçados quando somente um subconjunto das workloads está em execução.

É possível usar o WLM automático com prioridades de consulta para indicar as prioridades relativas dos workloads, evitando os problemas anteriores. Para este exemplo, siga as seguintes etapas:

- Crie um grupo de parâmetros e alterne para o modo Auto WLM (WLM automático).
- Adicione filas para cada uma das três workloads: workload de ETL, workload de análise e workload de ciência de dados. Use os mesmos grupos de usuários para cada workload usado com o modo Manual WLM (WLM manual).
- Defina a prioridade para o workload de ETL como `High`, a do workload de análise como `Normal` e a de ciência de dados como `Low`. Essas prioridades refletem as prioridades de seus negócios para as diferentes workloads ou grupos de usuários.
- Opcionalmente, habilite a escalabilidade da simultaneidade para a fila de análise ou de ciência de dados para que as consultas nessas filas obtenham uma performance consistente, mesmo quando a workload de ETL estiver sendo executada a cada 6 horas.

Com as prioridades de consulta, quando somente a workload de análise estiver em execução no cluster, ela obterá o sistema todo para si mesma. Isso gera alto throughput com melhor utilização do sistema. No entanto, quando o workload de ETL é iniciado, ele obtém a precedência, já que tem uma prioridade maior. As consultas em execução como parte da workload de ETL obtêm prioridade durante a admissão, além de alocação de recursos preferencial após serem admitidas. Como consequência, o workload de ETL é executado de maneira previsível, independentemente do que mais possa estar sendo executado no sistema. A performance previsível para uma workload de alta prioridade ocorre em detrimento de outra. As workloads de baixa prioridade que são executadas por mais tempo, seja porque as consultas estão aguardando que consultas mais importantes sejam concluídas. Ou porque eles estão recebendo uma fração menor de recursos quando são executados simultaneamente com consultas de prioridade mais alta. Os algoritmos de programação usados pelo Amazon Redshift possibilitam que as consultas de prioridade mais baixa não sofram de inanição, mas continuem progredindo, embora em um ritmo mais lento.

Note

- O campo de tempo limite não está disponível para WLM automático. Em vez disso, use a regra QMR, `query_execution_time`. Para obter mais informações, consulte Regras de monitoramento de consulta do WLM.
- A ação de QMR, HOP, não se aplica a WLM automático. Em vez disso, use a ação `change priority`. Para obter mais informações, consulte Regras de monitoramento de consulta do WLM.
- Os clusters usam filas do WLM automático e do WLM manual de forma diferente, o que pode causar confusão com suas configurações. Por exemplo, você pode configurar a propriedade de prioridade em filas do WLM automático, mas não em filas do WLM manual. Dessa forma, evite misturar filas do WLM automático e filas do WLM manual em um grupo de parâmetros. Em vez disso, crie outro grupo de parâmetros ao migrar para o WLM automático.

Implementar o WLM automático

Com o gerenciamento automático de workload (WLM), o Amazon Redshift gerencia a simultaneidade de consultas e a alocação de memória. Você pode criar até oito filas com os identificadores de classe de serviço 100–107. Cada fila tem uma prioridade. Para obter mais informações, consulte Prioridade da consulta.

O WLM automático determina a quantidade de recursos que as consultas precisam e ajusta a simultaneidade com base na workload. Quando consultas que exigem grandes quantidades de recursos estiverem no sistema (por exemplo, uniões de hash entre tabelas grandes), a simultaneidade será menor. Quando consultas mais leves (como inserções, exclusões, varreduras ou agregações simples) forem enviadas, a simultaneidade será maior.

O WLM automático é separado da aceleração de consulta breve (SQA) e avalia as consultas de maneira diferente. O WLM automático e a SQA funcionam em conjunto para permitir que consultas leves e de execução breve sejam concluídas mesmo quando consultas de execução longa e com grande utilização de recursos estão ativas. Para obter mais informações sobre SQA, consulte Aceleração de consulta breve.

O Amazon Redshift habilita o WLM automático por meio de grupos de parâmetros:

- Se seus clusters usam o grupo de parâmetro padrão, o Amazon Redshift habilita o WLM automático para eles.
- Se os clusters usarem grupos de parâmetros padrão, será possível configurar os clusters para habilitar o WLM automático. Recomendamos que você crie um grupo de parâmetros separado para a configuração do WLM automático.

Para configurar o WLM, edite o parâmetro `wlm_json_configuration` em um grupo de parâmetros que possa ser associado a um ou mais clusters. Para obter mais informações, consulte Modificar a configuração do WLM.

As filas de consulta são definidas na configuração do WLM. Você pode adicionar filas de consultas adicionais à configuração do WLM padrão, até um total de oito filas de usuários. Você pode configurar o seguinte para cada fila de consultas:

- Prioridade
- Modo de escalabilidade da simultaneidade
- User groups (Grupos de usuários)
- Grupos de consultas
- Regras de monitoramento de consulta

Prioridade

É possível definir a importância relativa de consultas em um workload definindo um valor de prioridade. A prioridade é especificada para uma fila e herdada por todas as consultas associadas à fila. Para obter mais informações, consulte Prioridade da consulta.

Modo de escalabilidade da simultaneidade

Quando a escalabilidade de simultaneidade está habilitada, o Amazon Redshift adiciona automaticamente capacidade de cluster quando necessário para processar um aumento nas consultas de leitura e gravação simultâneas. Seus usuários veem os dados mais atuais, sejam as consultas executadas no cluster principal ou em um cluster de escalabilidade de simultaneidade.

Gerencie quais consultas são enviadas para o cluster de escalabilidade da simultaneidade configurando filas do WLM. Ao habilitar a escalabilidade de simultaneidade para uma fila, as consultas qualificadas são enviadas ao cluster de escalabilidade de simultaneidade em vez de aguardar na fila. Para obter mais informações, consulte Escalabilidade simultânea..

User groups (Grupos de usuários)

Você pode atribuir um conjunto de grupos de usuários a uma fila especificando o nome de cada grupo de usuários ou usando curingas. Quando um membro de um grupo de usuários listado executa uma consulta, esta é executada na fila correspondente. Não há limite definido quanto ao número de grupos de usuários que podem ser atribuídos a uma fila. Para obter mais informações, consulte Atribuir consultas a filas com base em grupos de usuários.

Funções de usuário

Você pode atribuir um conjunto de perfis de usuário a uma fila especificando o nome de cada perfil de usuário ou usando curingas. Quando um membro de um perfil de usuário listado executa uma consulta, esta é executada na fila correspondente. Não há limite definido quanto ao número de perfis de usuário que podem ser atribuídos a uma fila. Para obter mais informações, consulte Atribuir consultas a filas com base em perfis de usuário.

Grupos de consultas

Você pode atribuir um conjunto de grupos de consultas a uma fila especificando o nome de cada grupo de consultas ou usando curingas. Um grupo de consultas é apenas um rótulo. No tempo de

execução, é possível atribuir o rótulo do grupo de consultas a uma série de consultas. Todas as consultas atribuídas a um grupo de consultas listado são executadas na fila correspondente. Não há limite definido para o número de grupos de consultas que podem ser atribuídos a uma fila. Para obter mais informações, consulte Atribuir uma consulta a um grupo de consultas.

Curingas

Se os curingas estiverem habilitados na configuração da fila WLM, você pode atribuir grupos de usuários e grupos de consulta a uma fila individualmente ou usando curingas no estilo shell do Unix. A comparação de padrões não diferencia maiúsculas de minúsculas.

Por exemplo, o caractere curinga "*" corresponde a qualquer número de caracteres. Portanto, se você adicionar `dba_*` à lista de grupos de usuários para uma fila, qualquer consulta executada por usuário, que pertença a um grupo com um nome que começa com `dba_`, será atribuída a essa fila. Os exemplos são `dba_admin` ou `DBA_primary`. O caractere curinga "?" corresponde a qualquer caractere único. Portanto, se a fila incluir o grupo de usuários `dba?1`, os grupos de usuários chamados `dba11` e `dba21` corresponderão, mas `dba12` não corresponderá.

Por padrão, os curingas não estão habilitados.

Regras de monitoramento de consulta

As regras de monitoramento de consultas definem limites de performance baseados em métricas para filas do WLM e especificam qual ação tomar quando uma consulta vai além desses limites. Por exemplo, para uma fila dedicada a consultas rápidas, convém criar uma regra que cancele consultas executadas por mais de 60 segundos. Para acompanhar consultas mal projetadas, convém ter outra regra que registre consultas que contenham loops aninhados. Para obter mais informações, consulte Regras de monitoramento de consulta do WLM.

Verificação do WLM automático

Para verificar se o WLM automático está habilitado, execute a seguinte consulta. Se a consulta retornar pelo menos uma linha, o WLM automático estará habilitado.

```
select * from stv_wlm_service_class_config
where service_class >= 100;
```

A consulta a seguir exibe o numero de consultas que passaram por cada fila de consulta (classe de serviço). Ele também mostra o tempo de execução médio, o número de consultas com tempo

de espera no percentil 90º, e o tempo de espera médio. As consultas de WLM automático usam as classes de serviço 100 a 107.

```
select final_state, service_class, count(*), avg(total_exec_time),
percentile_cont(0.9) within group (order by total_queue_time), avg(total_queue_time)
from stl_wlm_query where userid >= 100 group by 1,2 order by 2,1;
```

Para descobrir quais consultas foram executadas pelo WLM automático e concluídas com êxito, execute a seguinte consulta.

```
select a.queue_start_time, a.total_exec_time, label, trim(querytxt)
from stl_wlm_query a, stl_query b
where a.query = b.query and a.service_class >= 100 and a.final_state = 'Completed'
order by b.query desc limit 5;
```

Prioridade da consulta

Com o Amazon Redshift, é possível gerenciar a priorização de consultas e a alocação de recursos em workloads e consultas simultâneas usando o gerenciamento de workloads (WLM). As seções a seguir detalham como configurar filas de consulta do WM, definir propriedades da fila, como alocação de memória e escalabilidade simultânea, e implementar regras de prioridade adaptadas aos requisitos da workload.

Nem todas as consultas têm a mesma importância e, geralmente, a performance de um workload ou de um conjunto de usuários pode ser mais importante. Se você habilitou o WLM automático pode definir a importância relativa de consultas em um workload ao configurar um valor de prioridade. A prioridade é especificada para uma fila e herdada por todas as consultas associadas à fila. Associe as consultas a uma fila mapeando grupo de usuários e de consultas à fila. É possível definir as seguintes prioridades (listadas da prioridade mais alta para a mais baixa):

1. `HIGHEST`
2. `HIGH`
3. `NORMAL`
4. `LOW`
5. `LOWEST`

Os administradores usam essas prioridades para mostrar a importância relativa de seus workloads quando há consultas com prioridades diferentes competindo pelos mesmos recursos. O Amazon

Redshift usa a prioridade ao permitir consultas no sistema e determinar a quantidade de recursos alocados a uma consulta. Por padrão, as consultas são executadas com a prioridade definida como `NORMAL`.

Uma prioridade adicional, `CRITICAL`, que é uma prioridade mais alta que `HIGHEST`, está disponível para superusuários. Para definir essa prioridade, é possível usar as funções CHANGE_QUERY_PRIORITY, CHANGE_SESSION_PRIORITY e CHANGE_USER_PRIORITY. Para conceder a um usuário de banco de dados permissão para usar essas funções, é possível criar um procedimento armazenado e conceder permissão a um usuário. Para ver um exemplo, consulte CHANGE_SESSION_PRIORITY.

> **Note**
>
> Somente uma consulta `CRITICAL` pode ser executada por vez.
> As reversões sempre são executadas com prioridade CRITICAL.

Vamos ver um exemplo onde a prioridade de um workload de extração, transformação e carregamento (ETL) é mais alta que a prioridade do workload de análise. O workload de ETL é executado a cada seis horas, e o workload de análise é executado durante todo o dia. Quando somente o workload de análise estiver em execução no cluster, ele obterá o sistema todo para ele mesmo, gerando alta taxa de transferência com utilização ideal do sistema. No entanto, quando o workload de ETL é iniciado, ele obtém direito ao caminho, pois tem uma prioridade maior. As consultas em execução como parte do workload de ETL obtém direito ao caminho durante a admissão, além da alocação de recursos preferencial após serem admitidas. Como consequência, o workload de ETL é executado de maneira previsível, independentemente do que mais possa estar sendo executado no sistema. Assim, ela fornece performance previsível e a capacidade de os administradores oferecerem acordos de nível de serviço (SLAs) para os usuários de negócios.

Em determinado cluster, a performance previsível para uma workload de prioridade alta ocorre em detrimento de outras workloads de prioridade baixa. Os workloads de prioridade baixa podem ser executados por mais tempo, pois suas consultas aguardam que consultas mais importantes sejam concluídas. Ou elas podem ser executadas por mais tempo porque recebem uma fração menor de recursos quando são executadas simultaneamente com consultas de prioridade mais alta. As consultas de prioridade mais baixa não sofrem esgotamento, mas continuam a progredir em um ritmo mais lento.

No exemplo anterior, o administrador pode habilitar a escalabilidade da simultaneidade para o workload de análise. Isso permite que o workload mantenha sua taxa de transferência, mesmo que o workload de ETL esteja sendo executado com uma prioridade alta.

Configurar a prioridade da fila

Se você habilitou o WLM automático, cada fila tem um valor de prioridade. As consultas são roteadas para as filas com base nos grupos de usuários e nos grupos de consultas. Comece com uma prioridade de fila definida como `NORMAL`. Defina a prioridade como mais alta ou mais baixa com base no workload associado aos grupos de usuários e aos grupos de consultas da fila.

Você pode alterar a prioridade de uma fila no console do Amazon Redshift. No console do Amazon Redshift, a página Gerenciamento de workload exibe as filas e permite a edição das propriedades da fila, como Prioridade. Para definir a prioridade usando a CLI ou as operações de API, use o parâmetro `wlm_json_configuration`. Para obter mais informações, consulte "Configurar o gerenciamento de workload" no Guia de gerenciamento de clusters do Amazon Redshift.

O exemplo de `wlm_json_configuration` a seguir define três grupos de usuários (`ingest`, `reporting` e `analytics`). As consultas enviadas de usuários de um desses grupos são executadas com prioridade `highest`, `normal` e `low`, respectivamente.

```
[
    {
        "user_group": [
            "ingest"
        ],
        "priority": "highest",
        "queue_type": "auto"
    },
    {
        "user_group": [
            "reporting"
        ],
        "priority": "normal",
        "queue_type": "auto"
    },
    {
        "user_group": [
            "analytics"
        ],
        "priority": "low",
        "queue_type": "auto",
```

```
        "auto_wlm": true
    }
]
```

Alterar a prioridade da consulta com as regras de monitoramento de consulta

Com as regras de monitoramento de consultas (QMR), é possível alterar a prioridade de uma consulta com base no comportamento dela durante a execução. É possível fazer isso especificando o atributo de prioridade em um predicado QMR, além de uma ação. Para obter mais informações, consulte Regras de monitoramento de consulta do WLM.

Por exemplo, é possível definir uma regra para cancelar qualquer consulta classificada como prioridade `high` que seja executada por mais de 10 minutos.

```
"rules" :[
  {
    "rule_name":"rule_abort",
    "predicate":[
      {
        "metric_name":"query_cpu_time",
        "operator":">",
        "value":600
      },
      {
        "metric_name":"query_priority",
        "operator":"=",
        "value":"high"
      }
    ],
    "action":"abort"
  }
]
```

Outro exemplo é definir uma regra para alterar a prioridade de consulta para `lowest` para qualquer consulta com a prioridade atual `normal` que derrame mais de 1 TB no disco.

```
"rules":[
  {
    "rule_name":"rule_change_priority",
    "predicate":[
      {
        "metric_name":"query_temp_blocks_to_disk",
```

```
        "operator":">",
        "value":1000000
      },
      {
        "metric_name":"query_priority",
        "operator":"=",
        "value":"normal"
      }
    ],
    "action":"change_query_priority",
    "value":"lowest"
  }
]
```

Monitorar a prioridade da consulta

Para exibir a prioridade de consultas em espera e em execução, visualize a coluna `query_priority` na tabela stv_wlm_query_state do sistema.

```
query    | service_cl | wlm_start_time             | state            | queue_time |
 query_priority
---------+------------+----------------------------+------------------+------------
+----------------
2673299  | 102        | 2019-06-24 17:35:38.866356 | QueuedWaiting    | 265116     |
 Highest
2673236  | 101        | 2019-06-24 17:35:33.313854 | Running          | 0          |
 Highest
2673265  | 102        | 2019-06-24 17:35:33.523332 | Running          | 0          |
 High
2673284  | 102        | 2019-06-24 17:35:38.477366 | Running          | 0          |
 Highest
2673288  | 102        | 2019-06-24 17:35:38.621819 | Running          | 0          |
 Highest
2673310  | 103        | 2019-06-24 17:35:39.068513 | QueuedWaiting    | 62970      |
 High
2673303  | 102        | 2019-06-24 17:35:38.968921 | QueuedWaiting    | 162560     |
 Normal
2673306  | 104        | 2019-06-24 17:35:39.002733 | QueuedWaiting    | 128691     |
 Lowest
```

Para listar a prioridade de consultas concluídas, consulte a coluna `query_priority` na tabela stl_wlm_query do sistema.

```
select query, service_class as svclass, service_class_start_time as starttime,
 query_priority
from stl_wlm_query order by 3 desc limit 10;
```

```
  query   | svclass |         starttime          |    query_priority
---------+---------+----------------------------+----------------------
 2723254 |     100 | 2019-06-24 18:14:50.780094 | Normal
 2723251 |     102 | 2019-06-24 18:14:50.749961 | Highest
 2723246 |     102 | 2019-06-24 18:14:50.725275 | Highest
 2723244 |     103 | 2019-06-24 18:14:50.719241 | High
 2723243 |     101 | 2019-06-24 18:14:50.699325 | Low
 2723242 |     102 | 2019-06-24 18:14:50.692573 | Highest
 2723239 |     101 | 2019-06-24 18:14:50.668535 | Low
 2723237 |     102 | 2019-06-24 18:14:50.661918 | Highest
 2723236 |     102 | 2019-06-24 18:14:50.643636 | Highest
```

Para otimizar a taxa de transferência do workload, o Amazon Redshift pode modificar a prioridade das consultas enviadas pelo usuário. O Amazon Redshift usa algoritmos avançados de Machine Learning para determinar quando essa otimização beneficia seu workload e a aplica automaticamente quando todas as condições a seguir forem atendidas.

- O WLM automático está habilitado.
- Apenas uma fila WLM é definida.
- Você não definiu regras de monitoramento de consulta (QMRs) que definem a prioridade da consulta. Tais regras incluem a métrica QMR `query_priority` ou a ação QMR `change_query_priority`. Para obter mais informações, consulte Regras de monitoramento de consulta do WLM.

Implementar o WLM manual

Com WLM manual, é possível gerenciar a performance do sistema e a experiência dos usuários modificando a configuração de WLM a fim de criar filas separadas para as consultas demoradas e as rápidas.

Quando os usuários executam consultas no Amazon Redshift, as consultas são encaminhadas para filas de consulta. Cada fila de consultas contém alguns slots de consulta. Cada fila recebe uma parte da memória disponível do cluster. A memória de uma fila é dividida entre os slots de consulta da

fila. Você pode habilitar o Amazon Redshift para gerenciar a simultaneidade de consultas com WLM automático. Para obter mais informações, consulte Implementar o WLM automático.

Como alternativa, é possível configurar as propriedades do WLM para cada fila de consultas. Faça isso especificando a maneira pela qual a memória é alocada entre slots e como as consultas podem ser roteadas para filas específicas em runtime. Você também pode configurar as propriedades de WLM para cancelar consultas demoradas.

Por padrão, o Amazon Redshift configura as seguintes filas de consulta:

- Uma fila de usuários avançados

  A fila de superusuários é reservada somente para estes e não pode ser configurada. Use essa fila quando precisar executar consultas que afetam o sistema ou para fins de solução de problemas. Por exemplo, use essa fila quando você precisar cancelar a consulta demorada de um usuário ou adicionar usuários ao banco de dados. Não use para realizar consultas de rotina. A fila não é exibida no console, mas aparece nas tabelas do sistema do banco de dados como a quinta fila. Para executar uma consulta na fila de superusuários, um usuário deve estar conectado como superusuário e executar a consulta usando o grupo de consultas `superuser` predefinido.
- Uma fila de usuários padrão

  A fila padrão é configurada inicialmente para executar simultaneamente cinco consultas. Ao usar WLM manual, você pode alterar as propriedades de simultaneidade, tempo limite e alocação da memória da fila padrão, mas não pode especificar grupos de usuários ou consultas. A fila padrão deve ser a última na configuração do WLM. Algumas consultas que não são roteadas para outras filas são executadas na fila padrão.

As filas de consultas são definidas na configuração do WLM. A configuração do WLM é um parâmetro editável (`wlm_json_configuration`) em um parameter group, que pode ser associado a um ou mais clusters. Para obter mais informações, consulte “Configurar o gerenciamento de workload” no Guia de gerenciamento de clusters do Amazon Redshift.

Você pode adicionar filas de consultas adicionais à configuração do WLM padrão, até um total de oito filas de usuários. Você pode configurar o seguinte para cada fila de consultas:

- Modo de escalabilidade da simultaneidade
- Nível de simultaneidade
- User groups (Grupos de usuários)

- Grupos de consultas
- Porcentagem de memória do WLM a ser usada
- Tempo limite do WLM
- Salto na fila de consultas do WLM
- Regras de monitoramento de consulta

Modo de escalabilidade da simultaneidade

Quando a escalabilidade de simultaneidade está habilitada, o Amazon Redshift adiciona automaticamente capacidade de cluster quando necessário para processar um aumento nas consultas de leitura e gravação simultâneas. Os usuários veem os dados mais recentes, sejam as consultas executadas no cluster principal ou em um cluster de escalabilidade da simultaneidade.

Gerencie quais consultas são enviadas para o cluster de escalabilidade da simultaneidade configurando filas do WLM. Ao habilitar a escalabilidade de simultaneidade para uma fila, as consultas qualificadas são enviadas ao cluster de escalabilidade de simultaneidade em vez de aguardar na fila. Para obter mais informações, consulte Escalabilidade simultânea..

Nível de simultaneidade

As consultas em uma fila são executadas simultaneamente até que atinjam o número de slots de consulta do WLM ou o nível de simultaneidade, definido para essa fila. Em seguida, as consultas subsequentes aguardam na fila.

> **Note**
>
> O nível de simultaneidade do WLM é diferente do número de conexões de usuário simultâneas que podem ser estabelecidas com um cluster. Para obter mais informações, consulte “Conectar-se a um cluster” no Guia de gerenciamento de clusters do Amazon Redshift.

Em uma configuração de WLM automático (recomendada), o nível de simultaneidade é definido como Automático. O Amazon Redshift aloca memória dinamicamente para consultas, o que posteriormente determina quantas serão executadas simultaneamente. Isso se baseia nos recursos necessários para consultas em execução e consultas em fila. O WLM automático não é configurável. Para obter mais informações, consulte Implementar o WLM automático.

Em uma configuração de WLM manual, o Amazon Redshift aloca estaticamente uma quantidade fixa de memória para cada fila. A memória da fila é dividida igualmente entre os slots de consulta. Para ilustrar, se uma fila receber 20% da memória de um cluster e tiver 10 slots, cada consulta receberá 2% da memória do cluster. A alocação de memória permanece fixa, independentemente do número de consultas em execução simultânea. Devido a essa alocação de memória fixa, consultas executadas inteiramente na memória quando o número de slots é 5 talvez gravem resultados intermediários em disco, caso o número de slots seja aumentado para 20. Nesse caso, a parte de cada consulta na memória da fila é reduzida de 1/5 para 1/20. A E/S de disco adicional pode afetar a performance.

O número máximo de slots para todas as filas definidas pelo usuário é 50. Isso limita o total de slots para todas as filas, incluindo a fila padrão. A única fila que não está sujeita ao limite é a fila reservada para superusuários.

Por padrão, as filas do WLM manuais têm um nível de simultaneidade de 5. O workload pode se beneficiar de um nível de simultaneidade superior em determinados casos, como o seguinte:

- Se muitas consultas pequenas forem forçadas a aguardar consultas demoradas, crie uma fila à parte com um número de slots superior e atribua as consultas menores a essa fila. Uma fila com um nível de simultaneidade superior tem menos memória alocada para cada slot de consulta, mas as consultas menores exigem menos memória.

  > **Note**
  >
  > Se você habilitar a aceleração de consultas breves (SQA), o WLM priorizará automaticamente as consultas breves sobre as consultas demoradas, para que você não precise de uma fila separada para consultas breves na maioria dos fluxos de trabalho. Para obter mais informações, consulte Aceleração de consulta breve.

- Se você tiver várias consultas que acessam dados individualmente em uma única fatia, configure uma fila do WLM à parte para executar essas consultas simultaneamente. O Amazon Redshift atribui consultas simultâneas a fatias separadas, o que permite executar várias consultas em paralelo em várias fatias. Por exemplo, se uma consulta for um agregado simples com um predicado na chave de distribuição, os dados da consulta estarão localizados em uma única fatia.

Um exemplo de WLM manual

Este exemplo é um cenário simples de WLM manual para mostrar como os slots e a memória podem ser alocados. Implemente o WLM manual com três filas, que são as seguintes:

- fila de ingestão de dados: configurada para ingerir dados. Recebe 20% da memória do cluster e tem 5 slots. Consequentemente, 5 consultas podem ser executadas simultaneamente na fila e cada uma recebe 4% da memória.
- fila de cientistas de dados: projetada para consultas que consomem muita memória. Recebe 40% da memória do cluster e tem 5 slots. Consequentemente, 5 consultas podem ser executadas simultaneamente e cada uma recebe 8% da memória.
- fila padrão: projetada para a maioria dos usuários na organização. Isso inclui grupos de vendas e contabilidade, que normalmente têm consultas de curta ou média duração que não são complicadas. Recebe 40% da memória do cluster e tem 40 slots. Essa fila pode executar 40 consultas simultaneamente, e cada consulta recebe 1% da memória. Esse é o número máximo de slots que essa fila pode ter, pois o limite total entre todas as filas é 50.

Se você estiver realizando WLM automático e sua workload exigir execução simultânea de mais de 15 consultas, recomendamos ativar a escalabilidade da simultaneidade. Isso ocorre porque o aumento da contagem de slots de consulta acima de 15 pode criar contenção de recursos do sistema e limitar a throughput geral de um único cluster. Com a escalabilidade da simultaneidade, você pode executar centenas de consultas simultâneas até um número configurado de clusters de escalabilidade da simultaneidade. O número de clusters de escalabilidade da simultaneidade é controlado por max_concurrency_scaling_clusters. Para obter mais informações sobre a escalabilidade da simultaneidade, consulte Escalabilidade simultânea..

Para obter mais informações, consulte Aprimoramento do desempenho das consultas.

User groups (Grupos de usuários)

Você pode atribuir um conjunto de grupos de usuários a uma fila especificando o nome de cada grupo de usuários ou usando curingas. Quando um membro de um grupo de usuários listado executa uma consulta, esta é executada na fila correspondente. Não há limite definido quanto ao número de grupos de usuários que podem ser atribuídos a uma fila. Para obter mais informações, consulte Atribuir consultas a filas com base em grupos de usuários.

Funções de usuário

Você pode atribuir um conjunto de perfis de usuário a uma fila especificando o nome de cada perfil de usuário ou usando curingas. Quando um membro de um perfil de usuário listado executa uma consulta, esta é executada na fila correspondente. Não há limite definido quanto ao número de perfis de usuário que podem ser atribuídos a uma fila. Para obter mais informações, consulte Atribuir consultas a filas com base em perfis de usuário.

Grupos de consultas

Você pode atribuir um conjunto de grupos de consultas a uma fila especificando o nome de cada grupo de consultas ou usando curingas. Um grupo de consultas é apenas um rótulo. No tempo de execução, é possível atribuir o rótulo do grupo de consultas a uma série de consultas. Todas as consultas atribuídas a um grupo de consultas listado são executadas na fila correspondente. Não há limite definido para o número de grupos de consultas que podem ser atribuídos a uma fila. Para obter mais informações, consulte Atribuir uma consulta a um grupo de consultas.

Curingas

Se os curingas forem permitidos na configuração de fila do WLM, você poderá atribuir grupos de usuários e consultas a uma fila individualmente ou usando curingas em estilo shell do Unix. A comparação de padrões não diferencia maiúsculas de minúsculas.

Por exemplo, o caractere curinga "*" corresponde a qualquer número de caracteres. Portanto, se você adicionar `dba_*` à lista de grupos de usuários para uma fila, qualquer consulta executada por usuário, que pertença a um grupo com um nome que começa com `dba_`, será atribuída a essa fila. Os exemplos são `dba_admin` ou `DBA_primary`. O caractere curinga "?" corresponde a qualquer caractere único. Portanto, se a fila incluir o grupo de usuários `dba?1`, os grupos de usuários chamados `dba11` e `dba21` corresponderão, mas `dba12` não corresponderá.

Os curingas são desativados por padrão.

Porcentagem de memória do WLM a ser usada

Em uma configuração de WLM automático, a porcentagem de memória é definida como **`auto`**. Para obter mais informações, consulte Implementar o WLM automático.

Em uma configuração de WLM manual, para especificar o valor de memória disponível alocado para uma consulta, você pode definir o parâmetro `WLM Memory Percent to Use`. Por padrão, cada

fila definida pelo usuário recebe uma parte igual da memória disponível para consultas definidas pelo usuário. Por exemplo, se você tiver quatro filas definidas pelo usuário, cada fila receberá 25 por cento da memória disponível. A fila de superusuários tem a própria memória alocada e não pode ser modificada. Para alterar a alocação, você atribui uma porcentagem de memória em inteiro a cada fila, até um total de 100 por cento. Qualquer memória não alocada é gerenciada pelo Amazon Redshift e pode ser temporariamente fornecida a uma fila se a fila solicitar memória adicional para processamento.

Por exemplo, se configurar quatro filas, você poderá alocar a memória da seguinte maneira: 20 por cento, 30 por cento, 15 por cento, 15 por cento. Os 20 por cento restantes permanecem desalocados e são gerenciados pelo serviço.

Tempo limite do WLM

O tempo limite do WLM (`max_execution_time`) está obsoleto. Em vez disso, crie uma regra de monitoramento de consulta (QMR) usando `query_execution_time` para limitar o tempo de execução decorrido para uma consulta. Para obter mais informações, consulte Regras de monitoramento de consulta do WLM.

Para limitar o valor de tempo em que as consultas em uma determinada fila do WLM têm permissão para serem usadas, você pode definir o valor de tempo limite do WLM para cada fila. O parâmetro tempo limite especifica a quantidade de tempo, em milissegundos, durante o qual o Amazon Redshift aguarda a execução de uma consulta antes de cancelar ou ignorar a consulta. O tempo limite se baseia no tempo de execução da consulta e não inclui o tempo gasto esperando em uma fila.

O WLM tenta saltar as instruções CREATE TABLE AS (CTAS) e as consultas somente leitura, como instruções SELECT. Consultas que não podem ser saltadas são canceladas. Para obter mais informações, consulte Salto na fila de consultas do WLM.

O tempo limite do WLM não se aplica a uma consulta que tenha atingido o estado de retorno. Para exibir o estado de uma consulta, consulte a tabela de sistema STV_WLM_QUERY_STATE. As instruções COPY e as operações de manutenção, como ALTER, ANALYZE e VACUUM, não estão sujeitas ao tempo limite do WLM.

A função de tempo limite do WLM é semelhante ao parâmetro de configuração statement_timeout. A diferença é que, enquanto o parâmetro de configuração `statement_timeout` se aplica a todo o cluster, o tempo limite do WLM é específico para uma única fila na configuração do WLM.

Se statement_timeout também for especificado, o menor statement_timeout e tempo limite do WLM (max_execution_time) serão usados.

Regras de monitoramento de consulta

As regras de monitoramento de consultas definem limites de performance baseados em métricas para filas do WLM e especificam qual ação tomar quando uma consulta vai além desses limites. Por exemplo, para uma fila dedicada a consultas rápidas, convém criar uma regra que cancele consultas executadas por mais de 60 segundos. Para acompanhar consultas mal projetadas, convém ter outra regra que registre consultas que contenham loops aninhados. Para obter mais informações, consulte Regras de monitoramento de consulta do WLM.

Salto na fila de consultas do WLM

Com o Amazon Redshift, é possível gerenciar a simultaneidade de workloads e a alocação de recursos habilitando o salto na fila de consultas do gerenciamento de workloads (WLM). Esse recurso permite que as consultas “saltem” temporariamente de uma fila atribuída para uma fila de maior prioridade quando os recursos estão disponíveis, melhorando o desempenho geral da consulta e a utilização do sistema. As seções a seguir apresentam orientações detalhadas sobre como configurar e utilizar o salto na fila de consultas do WLM no Amazon Redshift.

Uma consulta pode ser saltada devido a um tempo limite do WLM ou uma ação de salto da regra de monitoramento de consultas (QMR). Só é possível saltar consultas em uma configuração manual do WLM.

Quando uma consulta é saltada, o WLM tenta rotear a consulta para a próxima fila correspondente com base nas regras de atribuição de filas do WLM. Se a consulta não corresponder a qualquer outra definição de fila, ela será cancelada. Ela não é atribuída à fila padrão.

Ações de tempo limite do WLM

A tabela a seguir resume o comportamento dos diferentes tipos de consultas com um tempo limite do WLM.

Tipo da consulta	Ação
INSERT, UPDATE e DELETE	Cancelar
Funções definidas pelo usuário (UDFs)	Cancelar
UNLOAD	Cancelar
COPY	Continuar a execução

Tipo da consulta	Ação
Operações de manutenção	Continuar a execução
Consultas somente leitura em estado `returning`	Continuar a execução
Consultas somente leitura em estado `running`	Reatribuir ou reiniciar
CREATE TABLE AS (CTAS), SELECT INTO	Reatribuir ou reiniciar

Salto na fila de tempo limite do WLM

O WLM salta os seguintes tipos de consultas quando seus tempos limite são expirados:

- Consultas somente leitura, como instruções SELECT que estejam em estado no WLM `running`. Para encontrar o estado do WLM de uma consulta, exiba a coluna STATE na tabela do sistema STV_WLM_QUERY_STATE.
- Instruções CREATE TABLE AS (CTAS). O salto de filas do WLM oferece suporte para as instruções CTAS definidas pelo usuário e para as geradas pelo sistema.
- Instruções SELECT INTO.

As consultas que não estão sujeitas ao tempo limite do WLM continuam a ser executadas na fila original até que sejam concluídas. Os seguintes tipos de consultas não estão sujeitos ao tempo limite do WLM:

- Instruções COPY
- Operações de manutenção, como ALTER, ANALYZE e VACUUM
- Consultas somente leitura, como instruções SELECT que estejam em estado no WLM `returning`. Para encontrar o estado do WLM de uma consulta, exiba a coluna STATE na tabela do sistema STV_WLM_QUERY_STATE.

Consultas que não estão qualificadas para a operação de salto por tempo limite do WLM são canceladas quando seus tempos limite expiram. Os seguintes tipos de consultas não estão qualificados para a operação de salto por tempo limite do WLM:

- Instruções INSERT, UPDATE e DELETE
- Instruções UNLOAD
- Funções definidas pelo usuário (UDFs)

Consultas reatribuídas e reiniciadas devido ao tempo limite do WLM

Quando uma consulta é saltada e nenhuma fila correspondente é encontrada, a consulta é cancelada.

Quando uma consulta é saltada e uma fila correspondente é encontrada, o WLM tenta reatribuir a consulta à nova fila. Caso não seja possível reatribuir uma consulta, ela é reiniciada em uma fila nova, como descrito a seguir.

Uma consulta somente será reatribuída se todas as afirmações a seguir forem verdadeiras:

- Uma fila correspondente foi encontrada.
- A fila nova tem slots livres suficientes para executar a consulta. Uma consulta pode exigir vários slots se o parâmetro wlm_query_slot_count foi definido com um valor maior que 1.
- A fila nova tem, no mínimo, a mesma quantidade de memória disponível usada pela consulta no momento.

Se a consulta for reatribuída, ela continuará sendo executada na fila nova. Os resultados intermediários são conservados, de forma que o efeito sobre o tempo total de execução é mínimo.

Se a consulta não puder ser reatribuída, ela será cancelada e reiniciada na nova fila. Os resultados intermediários são excluídos. A consulta espera na fila e inicia a execução quando há slots suficientes disponíveis.

Ações de salto do QMR

A tabela a seguir resume o comportamento dos diferentes tipos de consultas com uma ação de salto do QMR.

Tipo da consulta	Ação
COPY	Continuar a execução

Tipo da consulta	Ação
Operações de manutenção	Continuar a execução
Funções definidas pelo usuário (UDFs)	Continuar a execução
UNLOAD	Reatribuir ou continuar a execução
INSERT, UPDATE e DELETE	Reatribuir ou continuar a execução
Consultas somente leitura em estado `returning`	Reatribuir ou continuar a execução
Consultas somente leitura em estado `running`	Reatribuir ou reiniciar
CREATE TABLE AS (CTAS), SELECT INTO	Reatribuir ou reiniciar

Para saber se uma consulta que foi saltada pelo QMR foi reatribuída, reiniciada ou cancelada, consulte a STL_WLM_RULE_ACTIONtabela de log do sistema.

Consultas reatribuídas e reiniciadas devido à ação de salto do QMR

Quando uma consulta é saltada e nenhuma fila correspondente é encontrada, a consulta é cancelada.

Quando uma consulta é saltada e uma fila correspondente é encontrada, o WLM tenta reatribuir a consulta à nova fila. Caso não seja possível reatribuir uma consulta, ela será reiniciada na fila nova ou continuará a ser executada na fila original, como descrito a seguir.

Uma consulta somente será reatribuída se todas as afirmações a seguir forem verdadeiras:

- Uma fila correspondente foi encontrada.
- A fila nova tem slots livres suficientes para executar a consulta. Uma consulta pode exigir vários slots se o parâmetro wlm_query_slot_count foi definido com um valor maior que 1.
- A fila nova tem, no mínimo, a mesma quantidade de memória disponível usada pela consulta no momento.

Se a consulta for reatribuída, ela continuará sendo executada na fila nova. Os resultados intermediários são conservados, de forma que o efeito sobre o tempo total de execução é mínimo.

Caso não seja possível reatribuir uma consulta, ela será reiniciada ou continuará a ser executada na fila original. Se a consulta for reiniciada, ela será cancelada e reiniciada na nova fila. Os resultados intermediários são excluídos. A consulta espera na fila e inicia a execução quando há slots suficientes disponíveis.

Tutorial: Configuração de filas de gerenciamento do workload (WLM) manual

Com o Amazon Redshift, é possível configurar filas do gerenciamento de workloads (WLM) manual para priorizar e alocar recursos a diferentes tipos de consulta e usuário. As filas do WLM manual permitem que você controle as configurações de memória e simultaneidade de filas específicas, garantindo que as workloads essenciais recebam os recursos necessários e evitando que consultas de baixa prioridade monopolizem o sistema. As seções a seguir orientam você no processo de criação e configuração de filas do WLM manual no Amazon Redshift para atender aos requisitos de gerenciamento de workloads.

Visão geral

Recomendamos configurar o gerenciamento automático de workload (WLM) no Amazon Redshift. Para obter mais informações sobre o WLM automático, consulte Gerenciamento do workload. No entanto, se você precisar de várias filas WLM, este tutorial o orienta no processo de configuração do gerenciamento de workload manual (WLM) no Amazon Redshift. Ao configurar o WLM manual, é possível melhorar a performance de consulta e a alocação de recursos no cluster.

O Amazon Redshift roteia as consultas do usuário para filas para processamento. O WLM define como essas consultas são roteadas para as filas. Por padrão, o Amazon Redshift tem duas filas disponíveis para consultas: uma para superusuários e outra para usuários. A fila de superusuários não pode ser configurada e processa somente uma consulta por vez. Você deve reservar essa fila somente para fins de solução de problemas. A fila de usuários pode processar até cinco consultas por vez, mas você pode configurá-la alterando o nível de simultaneidade da fila, se necessário.

Quando tem diversos usuários executando consultas no banco de dados, você pode achar outra configuração mais eficiente. Por exemplo, se executarem operações que exijam muitos recursos, como VACUUM, alguns usuários poderão ter um impacto negativo sobre consultas menos intensivas, como relatórios. Convém considerar adicionar filas e configurá-las para workloads diferentes.

Tempo estimado: 75 minutos

Custo estimado: 50 centavos

Pré-requisitos

Você precisa de um cluster do Amazon Redshift, do banco de dados TICKIT de amostra e da ferramenta cliente Amazon Redshift RSQL. Se você ainda não fez essa configuração, consulte o Guia de conceitos básicos do Amazon Redshift e o Amazon Redshift RSQL.

Seções

- Seção 1: Compreender o comportamento do processamento de filas padrão
- Seção 2: Modificar a configuração da fila de consultas do WLM
- Seção 3: Rotear consultas para filas com base em grupos de usuários e grupos de consultas
- Seção 4: Usar wlm_query_slot_count para substituir temporariamente o nível de simultaneidade em uma fila
- Seção 5: Limpar os recursos

Seção 1: Compreender o comportamento do processamento de filas padrão

Antes de começar a configurar o WLM manual, é útil entender o comportamento padrão do processamento de fila no Amazon Redshift. Nesta seção, crie duas visualizações de banco de dados que retornam informações de diversas tabelas do sistema. Depois, execute algumas consultas de teste para saber como as consultas são roteadas por padrão. Para obter mais informações sobre tabelas de sistema, consulte Referência de visualizações e tabelas do sistema.

Etapa 1: Criar a visualização WLM_QUEUE_STATE_VW

Nesta etapa, crie uma visualização chamada WLM_QUEUE_STATE_VW. Essa visualização retorna informações das tabelas de sistema a seguir.

- STV_WLM_CLASSIFICATION_CONFIG
- STV_WLM_SERVICE_CLASS_CONFIG
- STV_WLM_SERVICE_CLASS_STATE

Essa visualização será usada ao longo do tutorial para monitorar o que acontecerá com as filas depois que alterar a configuração do WLM. A tabela a seguir descreve os dados retornados pela visualização WLM_QUEUE_STATE_VW.

Coluna	Descrição
queue (fila)	O número associado à linha que representa uma fila. O número da fila determina a ordem das filas no banco de dados.
descrição	Um valor que descreve se a fila está disponível somente para determina dos grupos de usuários, determinados grupos de consultas ou todos os tipos de consultas.
slots	O número de slots alocados para a fila.
mem	O valor de memória, em MB, por slot, alocado para a fila.
max_execution_time	O valor de tempo em que uma consulta tem permissão para ser executada antes de ser encerrada.
user_*	Um valor que indique se os caracteres curinga são permitidos na configuração do WLM para corresponder aos grupos de usuários.
query_*	Um valor que indique se os caracteres curinga são permitidos na configuração do WLM para corresponder aos grupos de consultas.
queued	O número de consultas que estão aguardando na fila para serem processadas.
executing	O número de consultas que estão em execução no momento.
executed	O número de consultas que já foram executadas.

Como criar a visualização WLM_QUEUE_STATE_VW

1. Abra o Amazon Redshift RSQL e conecte-se ao seu banco de dados de amostra TICKIT. Se você não tiver esse banco de dados, consulte Pré-requisitos.
2. Execute a consulta a seguir para criar a visualização WLM_QUEUE_STATE_VW.

```
create view WLM_QUEUE_STATE_VW as
select (config.service_class-5) as queue
, trim (class.condition) as description
, config.num_query_tasks as slots
```

```
, config.query_working_mem as mem
, config.max_execution_time as max_time
, config.user_group_wild_card as "user_*"
, config.query_group_wild_card as "query_*"
, state.num_queued_queries queued
, state.num_executing_queries executing
, state.num_executed_queries executed
from
STV_WLM_CLASSIFICATION_CONFIG class,
STV_WLM_SERVICE_CLASS_CONFIG config,
STV_WLM_SERVICE_CLASS_STATE state
where
class.action_service_class = config.service_class
and class.action_service_class = state.service_class
and config.service_class > 4
order by config.service_class;
```

3. Execute a consulta a seguir para ver as informações contidas pela visualização.

```
select * from wlm_queue_state_vw;
```

Este é um resultado de exemplo.

```
query | description                                | slots | mem | max_time | user_* |
 query_* | queued | executing | executed
------+--------------------------------------------+-------+-----+----------+--------
+---------+--------+-----------+----------
    0 | (super user) and (query group: superuser) |     1 | 357 |        0 |  false |
 false   |      0 |         0 |        0
    1 | (querytype:any)                           |     5 | 836 |        0 |  false |
 false   |      0 |         1 |      160
```

Etapa 2: Criar a visualização WLM_QUERY_STATE_VW

Nesta etapa, crie uma visualização chamada WLM_QUERY_STATE_VW. Essa visualização retorna informações da tabela de sistema STV_WLM_QUERY_STATE.

Essa visualização será usada ao longo do tutorial para monitorar as consultas em execução. A tabela a seguir descreve os dados retornados pela visualização WLM_QUERY_STATE_VW.

Coluna	Descrição
consultar	O ID da consulta.
queue (fila)	O número da fila.
slot_count	O número de slots alocados para a consulta.
start_time	A hora em que a consulta foi iniciada.
estado	O estado da consulta, como em execução.
queue_time	O número de microssegundos em que a consulta passou na fila.
exec_time	O número de microssegundos transcorridos desde que a consulta foi executada.

Como criar a visualização WLM_QUERY_STATE_VW

1. Em RSQL, execute a consulta a seguir para criar a visualização WLM_QUERY_STATE_VW.

```
create view WLM_QUERY_STATE_VW as
select query, (service_class-5) as queue, slot_count, trim(wlm_start_time) as
 start_time, trim(state) as state, trim(queue_time) as queue_time, trim(exec_time) as
 exec_time
from stv_wlm_query_state;
```

2. Execute a consulta a seguir para ver as informações contidas pela visualização.

```
select * from wlm_query_state_vw;
```

   Este é um resultado de exemplo.

```
query | queue | slot_count | start_time          | state     | queue_time | exec_time
------+-------+------------+---------------------+-----------+------------
+-----------
 1249 |     1 |          1 | 2014-09-24 22:19:16 | Executing | 0          | 516
```

Etapa 3: Executar consultas de teste

Nesta etapa, execute consultas de várias conexões em RSQL e examine as tabelas de sistema para determinar como as consultas foram roteadas para processamento.

Para esta etapa, são necessárias duas janelas RSQL abertas:

- Na janela RSQL 1, execute consultas que monitoram o estado das filas e das consultas usando as visualizações já criadas neste tutorial.
- Na janela RSQL 2, execute consultas demoradas para alterar os resultados encontrados na janela RSQL 1.

Como executar as consultas de teste

1. Abra duas janelas RSQL. Se já tiver uma janela aberta, você precisará somente abrir uma segunda. Você pode usar a mesma conta de usuário para ambas as conexões.
2. Na janela RSQL 1, execute a consulta a seguir.

   ```
   select * from wlm_query_state_vw;
   ```

   Este é um resultado de exemplo.

   ```
   query | queue | slot_count | start_time          | state     | queue_time | exec_time
   ------+-------+------------+---------------------+-----------+------------
   +-----------
    1258 |     1 |          1 | 2014-09-24 22:21:03 | Executing | 0          | 549
   ```

   Essa consulta retorna um resultado autorreferencial. A consulta em execução no momento é a instrução SELECT nesta visualização. Uma consulta nessa visualização sempre retorna pelo menos um resultado. Compare esse resultado com o resultado ocorrido depois de iniciar a consulta demorada na próxima etapa.
3. Na janela RSQL 2, execute uma consulta no banco de dados de exemplo TICKIT. Essa consulta deve ser executada por aproximadamente um minuto, de maneira que você tenha tempo de explorar os resultados das visualizações WLM_QUEUE_STATE_VW e WLM_QUERY_STATE_VW criadas anteriormente. Em alguns casos, você poderá descobrir que a consulta não é executada por tempo suficiente para consultar ambas as visualizações. Nesses casos, aumente o valor do filtro em `l.listid` para fazer com seja excutada por mais tempo.

> **Note**
>
> Para reduzir o tempo de execução da consulta e melhorar a performance do sistema, o Amazon Redshift armazena em cache os resultados de certos tipos de consultas na memória no nó líder. Quando o cache de resultados estiver habilitado, as consultas subsequentes serão executadas mais rapidamente. Para impedir que a consulta seja executada muito rapidamente, desabilite o cache de resultados para a sessão atual.

Para desabilitar o cache de resultados da sessão atual, defina o parâmetro enable_result_cache_for_session como `off`, conforme mostrado a seguir.

```
set enable_result_cache_for_session to off;
```

Na janela RSQL 2, execute a consulta a seguir.

```
select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <
 100000;
```

4. Na janela RSQL 1, as consultas WLM_QUEUE_STATE_VW e WLM_QUERY_STATE_VW comparam os resultados com os resultados anteriores.

```
select * from wlm_queue_state_vw;
select * from wlm_query_state_vw;
```

Estes são resultados de exemplo.

```
query | description                                | slots | mem | max_time | user_* |
 query_* | queued | executing | executed
------+--------------------------------------------+-------+-----+----------+--------
+---------+--------+-----------+----------
    0 | (super user) and (query group: superuser) |     1 | 357 |        0 |  false |
 false   |      0 |         0 |        0
    1 | (querytype:any)                           |     5 | 836 |        0 |  false |
 false   |      0 |         2 |      163

query | queue | slot_count | start_time          | state     | queue_time | exec_time
------+-------+------------+---------------------+-----------+------------
+-----------
```

```
1267 |     1 |          1 | 2014-09-24 22:22:30 | Executing | 0          | 684
1265 |     1 |          1 | 2014-09-24 22:22:36 | Executing | 0          | 4080859
```

Observe as seguintes diferenças entre as consultas anteriores e os resultados nesta etapa:

- Agora existem duas linhas em WLM_QUERY_STATE_VW. Um resultado é a consulta autorreferencial para executar uma operação SELECT nessa visualização. O segundo resultado é a consulta demorada da etapa anterior.
- A coluna em execução em WLM_QUEUE_STATE_VW aumentou de 1 a 2. Essa entrada de coluna significa que existem duas consultas em execução na fila.
- A coluna executada é incrementada sempre que você executa uma consulta na fila.

A visualização WLM_QUEUE_STATE_VW é útil para ter uma visualização geral das filas e quantas consultas estão sendo processadas em cada fila. A visualização WLM_QUERY_STATE_VW é útil para obter uma visão mais detalhada das consultas individuais em execução no momento.

Seção 2: Modificar a configuração da fila de consultas do WLM

Agora que compreende como as filas funcionam por padrão, aprenda como configurar filas de consultas usando WLM manual. Nesta seção, crie e configure um novo grupo de parâmetros para o cluster. Você cria duas filas de usuários adicionais e configura-as para aceitar consultas com base no grupo de usuários das consultas ou nos rótulos dos grupos de consultas. Todas as consultas que não forem roteadas para uma dessas duas filas serão roteadas para a fila padrão em tempo de execução.

Para criar uma configuração manual de WLM em um grupo de parâmetros

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.
2. No menu de navegação, escolha Configurations (Configurações) e Workload management (Gerenciamento de workload) para exibir a página Workload management (Gerenciamento de workload).
3. Escolha Create (Criar) para exibir a janela Create parameter group (Criar grupo de parâmetros).
4. Insira **`WLMTutorial`** tanto para Nome do grupo de parâmetros como para Descrição e depois escolha Criar para criar o grupo de parâmetros.

> **Note**
>
> O Parameter group name (Nome do grupo de parâmetros) é convertido para todas letras minúsculas quando criado.

5. Na página Workload management (Gerenciamento do workload), escolha o grupo de parâmetros **`wlmtutorial`** para exibir a página de detalhes com guias para Parameters (Parâmetros) e Workload management (Gerenciamento do workload).
6. Confirme se você está na página Workload management (Gerenciamento do workload) e escolha Switch WLM mode (Alternar modo WLM) para exibir a janela Concurrency settings (Configurações de simultaneidade).
7. Escolha Manual WLM (WLM manual) e escolha Save (Salvar) para alternar para o WLM manual.
8. Escolha Edit workload queues (Editar filas de workload).
9. Escolha Add queue (Adicionar fila) duas vezes para adicionar duas filas. Agora existem três filas: Queue 1 (Fila 1), Queue 2 (Fila 2) e Default queue (Fila padrão).
10. Insira informações para cada fila da seguinte maneira:

    - Em Queue 1 (Fila 1), insira **`30`** para Memory (%) (Memória (%)), **`2`** para Concurrency on main (Simultaneidade em principal) e **`test`** para Query groups (Grupos de consultas). Deixe as outras configurações com os valores padrão.
    - Em Queue 2 (Fila 2), insira **`40`** para Memory (%) (Memória (%)), **`3`** para Concurrency on main (Simultaneidade em principal) e **`admin`** para User groups (Grupos de usuários). Deixe as outras configurações com os valores padrão.
    - Defina o valor de Simultaneidade na principal para a fila padrão como um valor maior ou igual a 1. Não faça nenhuma outra alteração na Fila padrão. O WLM atribui a memória não alocada à fila padrão.
11. Para salvar suas configurações, escolha Save (Salvar).

Depois, associe o grupo de parâmetros que tem a configuração de WLM manual com um cluster.

Para associar um grupo de parâmetros com uma configuração de WLM manual a um cluster.

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Redshift em https://console.aws.amazon.com/redshiftv2/.

2. No menu de navegação, escolha Clusters e, depois, Clusters para exibir uma lista de seus clusters.
3. Escolha seu cluster, como `examplecluster`, para exibir os detalhes do cluster. Em seguida, escolha a guia Properties para exibir as propriedades desse cluster.
4. Na seção Configurações do banco de dados, escolha Editar e Editar grupo de parâmetros para exibir a janela de grupos de parâmetros.
5. Para o Grupos de parâmetros escolha o grupo de parâmetros **`wlmtutorial`** que você criou anteriormente.
6. Selecione Salvar alterações para associar o grupo de parâmetros.

   O cluster é modificado com o grupo de parâmetros. Contudo, você precisa reinicializar o cluster para que as alterações também sejam aplicadas ao banco de dados.
7. Escolha seu cluster e, em seguida, selecione Reinicializar para Ações.

Depois que o cluster for reinicializado, seu status retornará para Available (Disponível).

Seção 3: Rotear consultas para filas com base em grupos de usuários e grupos de consultas

Agora, o cluster está associado a um novo grupo de parâmetros e o WLM está configurado. Em seguida, execute algumas consultas para ver como o Amazon Redshift roteia as consultas em filas para processamento.

Etapa 1: Visualizar a configuração da fila de consulta no banco de dados

Primeiro, verifique se o banco de dados tem a configuração do WLM esperada.

Como visualizar a configuração da fila de consultas

1. Abra RSQL e execute a consulta a seguir. A consulta usa a visualização WLM_QUEUE_STATE_VW criada em Etapa 1: Criar a visualização WLM_QUEUE_STATE_VW. Se já tiver uma sessão conectada ao banco de dados antes da reinicialização do cluster, é necessário reconectar.

   ```
   select * from wlm_queue_state_vw;
   ```

   Este é um resultado de exemplo.

```
query | description                                | slots | mem | max_time | user_* |
 query_* | queued | executing | executed
------+--------------------------------------------+-------+-----+----------+--------
+---------+--------+-----------+----------
    0 | (super user) and (query group: superuser) |     1 | 357 |        0 |  false |
 false   |      0 |         0 |        0
    1 | (query group: test)                       |     2 | 627 |        0 |  false |
 false   |      0 |         0 |        0
    2 | (suser group: admin)                      |     3 | 557 |        0 |  false |
 false   |      0 |         0 |        0
    3 | (querytype:any)                           |     5 | 250 |        0 |  false |
 false   |      0 |         1 |        0
```

Compare esses resultados com os resultados recebidos por você em Etapa 1: Criar a visualização WLM_QUEUE_STATE_VW. Observe que agora existem duas filas adicionais. Agora a fila 1 é a fila do grupo de consultas de teste, e a fila 2 é a fila do grupo de usuários administradores.

Agora a fila 3 é a padrão. A última fila na lista sempre é a fila padrão. Essa é a fila para a qual consultas são roteadas por padrão caso nenhum grupo de usuários ou de consultas seja especificado em uma consulta.

2. Execute a consulta a seguir para confirmar que a consulta agora é executada na fila 3.

```
select * from wlm_query_state_vw;
```

Este é um resultado de exemplo.

```
query | queue | slot_count | start_time          | state     | queue_time | exec_time
------+-------+------------+---------------------+-----------+------------
+-----------
 2144 |     3 |          1 | 2014-09-24 23:49:59 | Executing | 0          | 550430
```

Etapa 2: Executar uma consulta usando a fila de grupos de consultas

Como executar uma consulta usando a fila de grupos de consultas

1. Execute a consulta a seguir a fim de roteá-la para o grupo de consultas `test`.

```
set query_group to test;
```

```
select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <40000;
```

2. Na outra janela RSQL, execute a consulta a seguir.

```
select * from wlm_query_state_vw;
```

Este é um resultado de exemplo.

```
query | queue | slot_count | start_time          | state     | queue_time | exec_time
------+-------+------------+---------------------+-----------+------------
+-----------
 2168 |     1 |          1 | 2014-09-24 23:54:18 | Executing | 0          | 6343309
 2170 |     3 |          1 | 2014-09-24 23:54:24 | Executing | 0          | 847
```

A consulta foi roteada para o grupo de consultas de teste, que é a fila 1 agora.

3. Selecione tudo na visualização de estado da fila.

```
select * from wlm_queue_state_vw;
```

Você verá um resultado semelhante ao seguinte.

```
query | description                                | slots | mem | max_time | user_* |
 query_* | queued | executing | executed
------+--------------------------------------------+-------+-----+----------+--------
+---------+--------+-----------+----------
    0 | (super user) and (query group: superuser) |     1 | 357 |        0 |  false |
 false   |      0 |         0 |        0
    1 | (query group: test)                       |     2 | 627 |        0 |  false |
 false   |      0 |         1 |        0
    2 | (suser group: admin)                      |     3 | 557 |        0 |  false |
 false   |      0 |         0 |        0
    3 | (querytype:any)                           |     5 | 250 |        0 |  false |
 false   |      0 |         1 |        0
```

4. Agora redefina o grupo de consultas e reexecute a consulta longa:

```
reset query_group;
select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <40000;
```

5. Execute as consultas em relação às visualizações para ver os resultados.

```
select * from wlm_queue_state_vw;
select * from wlm_query_state_vw;
```

Estes são resultados de exemplo.

```
query | description                                | slots | mem | max_time | user_* |
 query_* | queued | executing | executed
------+--------------------------------------------+-------+-----+----------+--------
+---------+--------+-----------+----------
    0 | (super user) and (query group: superuser) |     1 | 357 |        0 |  false |
 false   |      0 |         0 |        0
    1 | (query group: test)                       |     2 | 627 |        0 |  false |
 false   |      0 |         0 |        1
    2 | (suser group: admin)                      |     3 | 557 |        0 |  false |
 false   |      0 |         0 |        0
    3 | (querytype:any)                           |     5 | 250 |        0 |  false |
 false   |      0 |         2 |        5

query | queue | slot_count | start_time          | state     | queue_time | exec_time
------+-------+------------+---------------------+-----------+------------
+-----------
 2186 |     3 |          1 | 2014-09-24 23:57:52 | Executing | 0          | 649
 2184 |     3 |          1 | 2014-09-24 23:57:48 | Executing | 0          | 4137349
```

O resultado deve ser a consulta em execução na fila 3 novamente.

Etapa 3: Criar um grupo e um usuário de banco de dados

Para executar todas as consultas nessa fila, você precisa criar o grupo de usuários no banco de dados e adicionar um usuário ao grupo. Depois, faça logon com o RSQL usando as credenciais do novo usuário e execute consultas. É necessário executar consultas como um superusuário, como usuário administrador, para criar usuários de banco de dados.

Como criar um usuário do banco de dados e um grupo de usuários

1. No banco de dados, crie um nome do usuário do banco de dados `adminwlm` executando o comando a seguir em uma janela RSQL.

```
create user adminwlm createuser password '123Admin';
```

2. Em seguida, execute os comandos a seguir para criar o novo grupo de usuários e adicionar o novo usuário `adminwlm` a ele.

```
create group admin;
alter group admin add user adminwlm;
```

Etapa 4: Executar uma consulta usando a fila de grupos de usuários

Na sequência, execute uma consulta e roteie-a para a fila do grupo de usuários. Você faz isso quando quiser rotear a consulta para uma fila configurada para processar o tipo de consulta que deseja executar.

Como executar uma consulta usando a fila de grupos de usuários

1. Na janela RSQL 2, execute as consultas a seguir a fim de alternar para a conta `adminwlm` e executar uma consulta desse usuário.

```
set session authorization 'adminwlm';
select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <40000;
```

2. Na janela RSQL 1, execute a consulta a seguir para ver a fila de consultas para a qual as consultas são roteadas.

```
select * from wlm_query_state_vw;
select * from wlm_queue_state_vw;
```

   Estes são resultados de exemplo.

```
query | description                                | slots | mem | max_time | user_* |
 query_* | queued | executing | executed
------+--------------------------------------------+-------+-----+----------+--------
+---------+--------+-----------+----------
    0 | (super user) and (query group: superuser) |     1 | 357 |        0 |  false |
 false   |      0 |         0 |        0
    1 | (query group: test)                       |     2 | 627 |        0 |  false |
 false   |      0 |         0 |        1
    2 | (suser group: admin)                      |     3 | 557 |        0 |  false |
 false   |      0 |         1 |        0
    3 | (querytype:any)                           |     5 | 250 |        0 |  false |
 false   |      0 |         1 |        8
```

```
query | queue | slot_count | start_time          | state     | queue_time | exec_time
------+-------+------------+---------------------+-----------+------------
+-----------
 2202 |     2 |          1 | 2014-09-25 00:01:38 | Executing | 0          | 4885796
 2204 |     3 |          1 | 2014-09-25 00:01:43 | Executing | 0          | 650
```

A fila em que essa consulta foi executada está na fila 2, a fila de usuários `admin`. Sempre que você executar consultas conectado como esse usuário, elas serão executadas na fila 2, a menos que especifique um grupo de consultas diferente a ser usado. A fila escolhida depende das regras de atribuição de fila. Para obter mais informações, consulte Regras de atribuição de fila do WLM.

3. Agora execute a consulta a seguir na janela RSQL 2.

```
set query_group to test;
select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <40000;
```

4. Na janela RSQL 1, execute a consulta a seguir para ver a fila de consultas para a qual as consultas são roteadas.

```
select * from wlm_queue_state_vw;
select * from wlm_query_state_vw;
```

Estes são resultados de exemplo.

```
query | description                                | slots | mem | max_time | user_* |
 query_* | queued | executing | executed
------+--------------------------------------------+-------+-----+----------+--------
+---------+--------+-----------+----------
    0 | (super user) and (query group: superuser) |     1 | 357 |        0 |  false |
 false   |      0 |         0 |        0
    1 | (query group: test)                       |     2 | 627 |        0 |  false |
 false   |      0 |         1 |        1
    2 | (suser group: admin)                      |     3 | 557 |        0 |  false |
 false   |      0 |         0 |        1
    3 | (querytype:any)                           |     5 | 250 |        0 |  false |
 false   |      0 |         1 |       10

query | queue | slot_count | start_time          | state     | queue_time | exec_time
------+-------+------------+---------------------+-----------+------------
+-----------
 2218 |     1 |          1 | 2014-09-25 00:04:30 | Executing | 0          | 4819666
```

```
 2220 |     3 |          1 | 2014-09-25 00:04:35 | Executing | 0           | 685
```

5. Quando terminar, redefina o grupo de consultas.

```
reset query_group;
```

Seção 4: Usar wlm_query_slot_count para substituir temporariamente o nível de simultaneidade em uma fila

Às vezes, os usuários podem precisar temporariamente de mais recursos para uma consulta específica. Em caso afirmativo, elas poderão usar a definição de configuração wlm_query_slot_count para substituir temporariamente a maneira como os slots são alocados em uma fila de consultas. Slots são unidades de memória e CPU usadas para processar consultas. Convém substituir a contagem de slots quando você tiver consultas ocasionais utilizando muitos recursos no cluster, como acontece quando você realiza uma operação VACUUM no banco de dados.

Você poderá perceber que os usuários geralmente precisarão definir wlm_query_slot_count para determinados tipos de consultas. Se esse for o caso, considerar ajustar a configuração de WLM e conceder aos usuários uma fila mais adequada às necessidades de suas consultas. Para obter mais informações sobre como substituir temporariamente o nível de simultaneidade usando a contagem de slots, consulte wlm_query_slot_count.

Etapa 1: Substituir o nível de simultaneidade usando wlm_query_slot_count

Para os fins deste tutorial, execute a mesma consulta SELECT de longa duração. Execute-a como o usuário `adminwlm` usando wlm_query_slot_count para aumentar o número de slots disponíveis para a consulta.

Como substituir o nível de simultaneidade usando wlm_query_slot_count

1. Aumente o limite na consulta para verificar se você tem tempo suficiente para consultar a visualização WLM_QUERY_STATE_VW e ver um resultado.

```
set wlm_query_slot_count to 3;
select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <40000;
```

2. Agora, consulte WLM_QUERY_STATE_VW com o usuário administrador para ver como a consulta está sendo executada.

```
select * from wlm_query_state_vw;
```

Este é um resultado de exemplo.

```
query | queue | slot_count | start_time          | state     | queue_time | exec_time
------+-------+------------+---------------------+-----------+------------
+-----------
 2240 |     2 |          1 | 2014-09-25 00:08:45 | Executing | 0          | 3731414
 2242 |     3 |          1 | 2014-09-25 00:08:49 | Executing | 0          | 596
```

A contagem de slots para a consulta é 3. Essa contagem significa que a consulta está usando todos os três slots para processar a consulta, alocando todos os recursos na fila para essa consulta.

3. Agora execute a consulta a seguir.

```
select * from WLM_QUEUE_STATE_VW;
```

Este é um resultado de exemplo.

```
query | description                                | slots | mem | max_time | user_* |
 query_* | queued | executing | executed
------+--------------------------------------------+-------+-----+----------+--------
+---------+--------+-----------+----------
    0 | (super user) and (query group: superuser) |     1 | 357 |        0 |  false |
 false   |      0 |         0 |        0
    1 | (query group: test)                       |     2 | 627 |        0 |  false |
 false   |      0 |         0 |        4
    2 | (suser group: admin)                      |     3 | 557 |        0 |  false |
 false   |      0 |         1 |        3
    3 | (querytype:any)                           |     5 | 250 |        0 |  false |
 false   |      0 |         1 |       25
```

A definição de configuração wlm_query_slot_count é válida somente para a sessão atual. Se a sessão expirar, ou outro usuário executar uma consulta, a configuração do WLM será usada.

4. Redefina a contagem de slots e reexecute o teste.

```
reset wlm_query_slot_count;
select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <40000;
```

Estes são resultados de exemplo.

```
query | description                                | slots | mem | max_time | user_* |
 query_* | queued | executing | executed
------+--------------------------------------------+-------+-----+----------+--------
+---------+--------+-----------+----------
    0 | (super user) and (query group: superuser) |     1 | 357 |        0 |  false |
 false   |      0 |         0 |        0
    1 | (query group: test)                       |     2 | 627 |        0 |  false |
 false   |      0 |         0 |        2
    2 | (suser group: admin)                      |     3 | 557 |        0 |  false |
 false   |      0 |         1 |        2
    3 | (querytype:any)                           |     5 | 250 |        0 |  false |
 false   |      0 |         1 |       14

query | queue | slot_count | start_time          | state     | queue_time | exec_time
------+-------+------------+---------------------+-----------+------------
+-----------
 2260 |     2 |          1 | 2014-09-25 00:12:11 | Executing | 0          | 4042618
 2262 |     3 |          1 | 2014-09-25 00:12:15 | Executing | 0          | 680
```

Etapa 2: Executar consultas em sessões diferentes

Em seguida, execute consultas em sessões diferentes.

Como executar consultas em sessões diferentes

1. Nas janelas RSQL 1 e 2, execute o seguinte para usar o grupo de consultas de teste.

```
set query_group to test;
```

2. Na janela RSQL 1, execute a consulta demorada a seguir.

```
select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <40000;
```

3. Enquanto a consulta demorada ainda estiver em execução na janela RSQL 1, execute o seguinte. Esses comandos aumentam a contagem de slots a fim de usar todos os slots para a fila e começa a executar a consulta demorada.

```
set wlm_query_slot_count to 2;
```

```
select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <40000;
```

4. Abra uma terceira janela RSQL e consulte as visualizações para ver os resultados.

```
select * from wlm_queue_state_vw;
select * from wlm_query_state_vw;
```

Estes são resultados de exemplo.

```
query | description                                | slots | mem | max_time | user_* |
 query_* | queued | executing | executed
------+--------------------------------------------+-------+-----+----------+--------
+---------+--------+-----------+----------
    0 | (super user) and (query group: superuser) |     1 | 357 |        0 |  false |
 false   |      0 |         0 |        0
    1 | (query group: test)                       |     2 | 627 |        0 |  false |
 false   |      1 |         1 |        2
    2 | (suser group: admin)                      |     3 | 557 |        0 |  false |
 false   |      0 |         0 |        3
    3 | (querytype:any)                           |     5 | 250 |        0 |  false |
 false   |      0 |         1 |       18

query | queue | slot_count | start_time          | state         | queue_time |
 exec_time
------+-------+------------+---------------------+---------------+------------
+-----------
 2286 |     1 |          2 | 2014-09-25 00:16:48 | QueuedWaiting | 3758950    | 0
 2282 |     1 |          1 | 2014-09-25 00:16:33 | Executing     | 0          |
 19335850
 2288 |     3 |          1 | 2014-09-25 00:16:52 | Executing     | 0          | 666
```

Observe que a primeira consulta usa um dos slots alocados à fila 1 para executar a consulta. Além disso, observe que há uma consulta que está aguardando na fila (onde `queued` é `1` e `state` é `QueuedWaiting`). Depois que a primeira consulta for concluída, a segunda será executada. Essa execução acontece porque ambas as consultas são roteadas para o grupo de consultas `test`, e a segunda consulta deve aguardar slots suficientes para começar o processamento.

Seção 5: Limpar os recursos

O cluster continua acumulando cobranças enquanto está em execução. Ao concluir este tutorial, retorne seu ambiente ao estado anterior seguindo as etapas em Encontrar recursos adicionais e redefinir seu ambiente no Guia de conceitos básicos do Amazon Redshift.

Para obter mais informações sobre WLM, consulte Gerenciamento do workload.

Escalabilidade simultânea.

Com o recurso de escalabilidade da simultaneidade, é possível oferecer suporte a milhares de usuários e consultas simultâneos, com performance de consulta consistentemente rápida. Ao ativar a escalabilidade de simultaneidade, o Amazon Redshift adicionará automaticamente capacidade de cluster para processar um aumento em consultas de leitura e de gravação. Os usuários veem os dados mais atuais, sejam as consultas executadas no cluster principal ou em um cluster de escalabilidade de simultaneidade.

É possível gerenciar quais consultas são enviadas ao cluster de escalabilidade simultânea configurando filas WLM. Quando você ativa a escalabilidade de simultaneidade, as consultas elegíveis são enviadas para o cluster de escalabilidade de simultaneidade em vez de esperar em uma fila.

Você é cobrado por clusters de escalabilidade de simultaneidade apenas pelo tempo em que estão executando ativamente as consultas. Para obter mais informações sobre preços, incluindo como as cobranças são acumuladas e cobranças mínimas, consulte Preço da escalabilidade de simultaneidade.

Tópicos

- Recursos de escalabilidade de simultaneidade
- Limitações para a escalabilidade de simultaneidade
- Regiões da AWSEscalabilidade de simultaneidade de
- Candidatos da escalabilidade da simultaneidade
- Configurar filas da escalabilidade da simultaneidade
- Monitoramento da escalabilidade da simultaneidade
- Visualizações do sistema de escalabilidade da simultaneidade

Recursos de escalabilidade de simultaneidade

Quando você ativa a escalabilidade de simultaneidade para uma fila do WLM, ele funciona para operações de leitura, como consultas de painel. Ele também funciona para operações de gravação comumente usadas, como instruções para ingestão e processamento de dados.

Recursos de escalabilidade de simultaneidade para operações de gravação

A escalabilidade de simultaneidade oferece suporte a operações de gravação usadas com frequência, como instruções de extrair, transformar e carregar (ETL). A escalabilidade de simultaneidade para operações de gravação é especialmente útil quando você deseja manter tempos de resposta consistentes quando o cluster recebe um grande número de solicitações. Ele melhora a taxa de transferência para operações de gravação disputando recursos no cluster principal.

A escalabilidade de simultaneidade dá suporte a instruções COPY, INSERT, DELETE, UPDATE, CREATE TABLE AS (CTAS) e VACUUM. Além disso, a escalabilidade simultânea aceita a atualização manual de visões materializadas (MVs) e operações de vácuo automático. Outras instruções Data Manipulation Language (DML) e Data Definition Language (DDL) não são compatíveis. Quando instruções de gravação não compatíveis, como CREATE sem TABLE AS, são incluídas em uma transação explícita antes das instruções de gravação compatíveis, nenhuma das instruções de gravação será executada em clusters de escalabilidade de simultaneidade.

Quando você provisiona crédito para escalabilidade de simultaneidade, essa provisão de crédito se aplica a operações de leitura e gravação.

Limitações para a escalabilidade de simultaneidade

Veja a seguir limitações para usar a escalabilidade de simultaneidade do Amazon Redshift:

- Ela não oferece suporte a consultas em tabelas que usam chaves de classificação intercaladas.
- Ela não oferece suporte a consultas em tabelas temporárias.
- Ela não oferece suporte a consultas que acessam recursos externos protegidos por configurações restritivas de rede ou Virtual Private Cloud (VPC).
- O recurso não é compatível com consultas que contêm funções definidas pelo usuário (UDFs) em Python e UDFs em Lambda.
- Ela não suporta consultas que acessam tabelas de sistema, tabelas de catálogo PostgreSQL ou tabelas sem backup.

- Ela não permite consultas COPY ou UNLOAD que acessam um recurso externo quando permissões da política do IAM estão em vigor. Isso inclui permissões aplicadas a um recurso, como um bucket do Amazon S3 ou uma tabela do DynamoDB, ou à origem. As origens do IAM podem incluir o seguinte:
  - `aws:sourceVpc`: uma VPC de origem.
  - `aws:sourceVpce`: um endpoint da VPC de origem.
  - `aws:sourceIp`: um endereço IP de origem.

  Em alguns casos, pode ser necessário remover permissões que restringem o recurso ou a origem, para que as consultas COPY e UNLOAD que acessam o recurso sejam enviadas ao cluster de escalabilidade simultânea.

  Para obter mais informações sobre políticas de recursos, consulte Tipos de políticas no guia do usuário do AWS Identity and Access Management e Controlar o acesso a partir de VPC endpoints com políticas de bucket.
- A escalabilidade de simultaneidade do Amazon Redshift para operações de gravação não é compatível com operações DDL, como CREATE TABLE.
- Não oferece suporte a ANALYZE para o comando COPY.
- Ela não suporta operações de gravação em uma tabela de destino onde DISTSTYLE está definido como ALL.
- Ela não suporta operações de gravação em tabelas com colunas de identidade.
- O Amazon Redshift oferece escalabilidade simultânea para operações de gravação somente em nós RG e RA3 do Amazon Redshift. A escalabilidade de simultaneidade para operações de gravação não é aceita em outros tipos de nó.

Regiões da AWSEscalabilidade de simultaneidade de

Com o Amazon Redshift, é possível usar a escalabilidade simultânea para gerenciar demandas simultâneas de workload em todos os clusters do Redshift. Este tópico detalha em quais regiões é possível usar a escalabilidade simultânea com o Amazon Redshift.

A escalabilidade de simultaneidade está disponível nestas regiões da AWS:

- Região Leste dos EUA (Norte da Virgínia) (us-east-1)
- Região Leste dos EUA (Ohio) (us-east-2)
- Região Oeste dos EUA (Norte da Califórnia) us-west-1

- Região Oeste dos EUA (Oregon) us-west-2
- Região da África (Cidade do Cabo) (af-south-1)
- Região da Ásia-Pacífico (Mumbai) (ap-south-1)
- Região Ásia-Pacífico (Haiderabade) (ap-south-2)
- Região da Ásia-Pacífico (Seul) (ap-northeast-2)
- Região da Ásia-Pacífico (Osaka) (ap-northeast-3)
- Região da Ásia-Pacífico (Singapura) (ap-southeast-1)
- Região da Ásia-Pacífico (Sydney) (ap-southeast-2)
- Região da Ásia-Pacífico (Jakarta) (ap-southeast-3)
- Região Ásia-Pacífico (Malásia) (ap-southeast-5)
- Região Ásia-Pacífico (Nova Zelândia) (ap-southeast-6)
- Região Ásia-Pacífico (Tailândia) (ap-southeast-7)
- Região da Ásia-Pacífico (Hong Kong) (ap-east-1)
- Região Ásia-Pacífico (Taipei) (ap-east-2)
- Região da Ásia-Pacífico (Tóquio) (ap-northeast-1)
- Região do Canadá (Central) (ca-central-1)
- Região Oeste do Canadá (Calgary) (ca-west-1)
- Região da China (Pequim) (cn-north-1)
- Região da China (Ningxia) (cn-northwest-1)
- Região da Europa (Frankfurt) (eu-central-1)
- Região da Europa (Irlanda) (eu-west-1)
- Região da Europa (Londres) (eu-west-2)
- Região da Europa (Paris) (eu-west-3)
- Região da Europa (Estocolmo) (eu-north-1)
- Região Europa (Zurique) (eu-central-2)
- Região da Europa (Milão) (eu-south-1)
- Região Europa (Espanha) (eu-south-2)
- Região de Israel (Tel Aviv) (il-central-1)
- Região do Oriente Médio (Bahrein) (me-south-1)

- Região México (Centro) (mx-central-1)
- Região da América do Sul (São Paulo) (sa-east-1)
- AWS GovCloud (Leste dos EUA)
- AWS GovCloud (Oeste dos EUA)

Candidatos da escalabilidade da simultaneidade

Com o Amazon Redshift, é possível aumentar a escala horizontalmente do processamento de consultas para acelerar a execução de consultas simultâneas. O tópico a seguir descreve os critérios que o Amazon Redshift usa para determinar quais consultas devem ser encaminhadas para a escalabilidade simultânea.

As consultas são roteadas para o cluster de escalabilidade da simultaneidade somente quando o cluster principal atende aos seguintes requisitos:

- Plataforma EC2-VPC.
- O tipo de nó deve ser dc2.8xlarge, dc2.large, rg.large, rg.xlarge, rg.4xlarge, rg.12xlarge, ra3.large, ra3.xlplus, ra3.4xlarge ou ra3.16xlarge. A escalabilidade simultânea para operações de gravação é possível somente em nós RG e RA3 do Amazon Redshift.
- Máximo de 32 nós de computação para clusters com os tipos de nó rg.large, rg.xlarge, rg.4xlarge, rg.12xlarge, ra3.xlplus, ra3.4xlarge ou ra3.16xlarge. Além disso, o número de nós do cluster principal não pode ser maior que 32 nós quando o cluster foi originalmente criado. Por exemplo, mesmo que um cluster tenha atualmente 20 nós, mas tenha sido originalmente criado com 40, ele não atende aos requisitos para escalabilidade da simultaneidade. Por outro lado, se, no momento, um cluster DC2 tiver quarenta nós, mas tiver sido originalmente criado com vinte, ele atenderá aos requisitos para escalabilidade simultânea.
- Não um cluster de nó único.

Configurar filas da escalabilidade da simultaneidade

Com o Amazon Redshift, você pode gerenciar a simultaneidade e os recursos do sistema configurando a escalabilidade simultânea. Com as filas de escalabilidade simultânea, é possível definir limites para o número de consultas ou sessões de usuário que podem ser executadas simultaneamente. A seção a seguir fornece instruções sobre como habilitar as filas de escalabilidade simultânea no Amazon Redshift, para que seja possível lidar com eficácia com consultas e sessões de usuário simultâneas.

Roteie consultas para clusters de escalabilidade da simultaneidade habilitando essa escalabilidade em uma fila do gerenciamento do workload (WLM). Para habilitar a escalabilidade de simultaneidade para uma fila, defina o valor do Modo de escalabilidade de simultaneidade como Automático.

Quando o número de consultas roteadas para uma fila com a escalabilidade de simultaneidade habilitada excede a capacidade de simultaneidade da fila, seja a capacidade configurada manualmente ou determinada automaticamente, filas elegíveis são enviadas ao cluster de escalabilidade de simultaneidade. Quando os slots de fila ficam disponíveis no cluster principal, as consultas são roteadas e executadas no cluster principal. Assim como com qualquer fila do WLM, roteie consultas para uma fila de escalabilidade da simultaneidade em grupos de usuários, ou rotulando consultas com rótulos de grupo de consultas, ou concordando em atender às condições definidas em Atribuir consultas a filas. Você também pode rotear consultas definindo Regras de monitoramento de consulta do WLM. Por exemplo, você pode rotear todas as consultas que demoram mais que 5 segundos para uma fila da escalabilidade da simultaneidade. Lembre-se de que o comportamento do enfileiramento pode variar, dependendo se você está usando o WLM automático ou o WLM manual. Consulte mais informações em Implementar o WLM automático ou em Implementar o WLM manual.

O número padrão de clusters de escalabilidade da simultaneidade é um. O número de clusters de escalabilidade de simultaneidade que podem ser usados é controlado por max_concurrency_scaling_clusters.

Monitoramento da escalabilidade da simultaneidade

Com o Amazon Redshift, é possível monitorar e gerenciar a escalabilidade simultânea para otimizar o desempenho e o custo-benefício das workloads de data warehouse. A escalabilidade simultânea permite que o Amazon Redshift aumente automaticamente a capacidade de cluster quando as demandas de workload aumentam e remova essa capacidade quando as demandas diminuem. A seção a seguir apresenta orientações sobre como monitorar a escalabilidade simultânea de clusters do Amazon Redshift.

Você pode ver se uma consulta está em execução no cluster principal ou em um cluster de escalabilidade de simultaneidade acessando Cluster no console do Amazon Redshift e escolhendo um cluster. Depois, escolha a guia Monitoramento de consultas e Simultaneidade de carga de trabalho para visualizar informações sobre a execução de consultas e consultas na fila.

Para encontrar tempos de execução, consulte a tabela STL_QUERY e filtre na coluna `concurrency_scaling_status`. A consulta a seguir compara o tempo da fila e o tempo de

execução para consultas executadas no cluster de escalabilidade da simultaneidade e consultas executadas no cluster principal.

```
SELECT w.service_class AS queue
, CASE WHEN q.concurrency_scaling_status = 1 THEN 'concurrency scaling cluster' ELSE
 'main cluster' END as concurrency_scaling_status
, COUNT( * ) AS queries
, SUM( q.aborted ) AS aborted
, SUM( ROUND( total_queue_time::NUMERIC / 1000000,2) ) AS queue_secs
, SUM( ROUND( total_exec_time::NUMERIC / 1000000,2) ) AS exec_secs
FROM stl_query q
JOIN stl_wlm_query w
USING (userid,query)
WHERE q.userid > 1
AND q.starttime > '2019-01-04 16:38:00'
AND q.endtime < '2019-01-04 17:40:00'
GROUP BY 1,2
ORDER BY 1,2;
```

Ajuste os valores de `starttime` e `endtime` de acordo com suas necessidades.

Visualizações do sistema de escalabilidade da simultaneidade

Com o Amazon Redshift, é possível usar visualizações do sistema de escalabilidade simultânea para monitorar e gerenciar a atividade de escalabilidade simultânea no cluster. A seção a seguir descreve a consulta a essas visualizações do sistema e a interpretação dos resultados para aproveitar eficazmente a escalabilidade simultânea em seu ambiente do Amazon Redshift.

Um conjunto de visualizações do sistema com o prefixo SVCS fornece detalhes das tabelas de log do sistema sobre consultas nos clusters principal e de escalabilidade de simultaneidade.

As seguintes visualizações têm informações semelhantes às visualizações STL ou SVL correspondentes:

- SVCS_ALERT_EVENT_LOG
- SVCS_COMPILE
- SVCS_EXPLAIN
- SVCS_PLAN_INFO
- SVCS_QUERY_SUMMARY

- SVCS_STREAM_SEGS

As visualizações a seguir são específicas para a escalabilidade de simultaneidade.

- SVCS_CONCURRENCY_SCALING_USAGE

Para obter mais informações sobre a escalabilidade de simultaneidade, consulte os seguintes tópicos no Guia de gerenciamento de clusters do Amazon Redshift.

- Visualizar dados da escalabilidade da simultaneidade
- Exibir a performance do cluster durante a execução da consulta
- Visualizar detalhes da consulta

Aceleração de consulta breve

A aceleração de consultas breves (SQA) prioriza as consultas de curta execução sobre as consultas de execução demorada. A SQA executa consultas breves em um espaço dedicado, de maneira que as consultas SQA não sejam forçadas a esperar em filas atrás de consultas mais demoradas. O SQA apenas prioriza consultas que são de execução curta e estão em uma fila definida pelo usuário. Com a SQA, consultas breves são iniciadas com mais rapidez e os usuários veem os resultados mais cedo.

Se você habilitar a SQA, poderá reduzir as filas de gerenciamento de workload (WLM) que são dedicadas à execução de consultas breves. Além disso, as consultas demoradas não precisam disputar slots em uma fila com as consultas breves, e assim você pode configurar suas filas do WLM usando menos slots de consulta. Quando você usa um nível de simultaneidade menor, a taxa de transferência de consultas aumenta e a performance geral do sistema melhora na maioria dos workloads.

As instruções CREATE TABLE AS (CTAS) e consultas somente leitura, como as instruções SELECT, são qualificadas para a SQA.

O Amazon Redshift usa um algoritmo de Machine Learning para analisar cada consulta elegível e prever o tempo de execução da consulta. Por padrão, o WLM atribui dinamicamente um valor para o tempo máximo de execução de SQA baseado na análise do workload do cluster. Como alternativa, você pode especificar um valor fixo de 1 a 20 segundos. Se o runtime previsto da consulta for menor que o runtime máximo da SQA definido ou atribuído dinamicamente e a consulta estiver aguardando

em uma fila de WLM, a SQA vai separar a consulta das filas de WLM e programá-la para execução prioritária. Se uma consulta for executada por um tempo maior do que o tempo de execução máximo da SQA, o WLM moverá a consulta para a primeira fila correspondente do WLM com base nas regras de atribuição de filas do WLM. Com o passar do tempo, as previsões serão melhores pois a SQA aprenderá com os padrões das suas consulta.

O SQA é habilitado por padrão no grupo de parâmetros padrão e para todos os novos grupos de parâmetros. Para desabilitar o SQA no console do Amazon Redshift, edite a configuração do WLM para um grupo de parâmetros e desmarque Habilitar aceleração de consulta curta. Como melhor prática, recomendamos o uso de 15 ou menos slots de consulta do WLM para manter a performance geral do sistema no nível ideal. Para obter informações sobre como modificar as configurações do WLM, consulte "Configurar o gerenciamento de workload" no Guia de gerenciamento de clusters do Amazon Redshift.

Tempo máximo de execução para consultas breves

Ao permitir o SQA, o WLM define o tempo máximo de execução de consultas breves para dinâmico, por padrão. Recomendamos manter a configuração dinâmica para o tempo de execução máximo de SQA. Você pode substituir a configuração padrão especificando um valor fixo de 1 a 20 segundos.

Em alguns casos, você pode considerar o uso de valores diferentes para os valores de tempo de execução máximos de SQA a fim de melhorar a performance do sistema. Nesses casos, analise seu workload para encontrar o tempo de execução máximo para a maioria das consultas de execução breve. A seguinte consulta retorna o tempo máximo de execução para consultas com aproximadamente 70 por cento.

```
select least(greatest(percentile_cont(0.7)
within group (order by total_exec_time / 1000000) + 2, 2), 20)
from stl_wlm_query
where userid >= 100
and final_state = 'Completed';
```

Após identificar o valor máximo de tempo de execução que funcione bem para seu workload, não será necessário alterá-lo a menos que seus workload sejam alterados de maneira significativa.

Monitoramento da SQA

Para verificar se a SQA está habilitada, execute a seguinte consulta. Se a consulta retornar uma linha, a SQA está habilitada.

```
select * from stv_wlm_service_class_config
where service_class = 14;
```

A consulta a seguir exibe o numero de consultas que passaram por cada fila de consulta (classe de serviço). Ele também mostra o tempo de execução médio, o número de consultas com tempo de espera no percentil 90º, e o tempo de espera médio. As consultas de SQA usam a classe em serviço 14.

```
select final_state, service_class, count(*), avg(total_exec_time),
percentile_cont(0.9) within group (order by total_queue_time), avg(total_queue_time)
from stl_wlm_query where userid >= 100 group by 1,2 order by 2,1;
```

Para localizar as consultas que foram selecionadas pela SQA e concluídas com êxito, execute a seguinte consulta.

```
select a.queue_start_time, a.total_exec_time, label, trim(querytxt)
from stl_wlm_query a, stl_query b
where a.query = b.query and a.service_class = 14 and a.final_state = 'Completed'
order by b.query desc limit 5;
```

Para localizar as consultas que SQA selecionou mas que ultrapassaram o tempo limite, execute a seguinte consulta.

```
select a.queue_start_time, a.total_exec_time, label, trim(querytxt)
from stl_wlm_query a, stl_query b
where a.query = b.query and a.service_class = 14 and a.final_state = 'Evicted'
order by b.query desc limit 5;
```

Para obter mais informações sobre consultas removidas e informações mais genéricas sobre ações baseadas em regras que podem ser realizadas em consultas, consulte Regras de monitoramento de consulta do WLM.

Regras de atribuição de fila do WLM

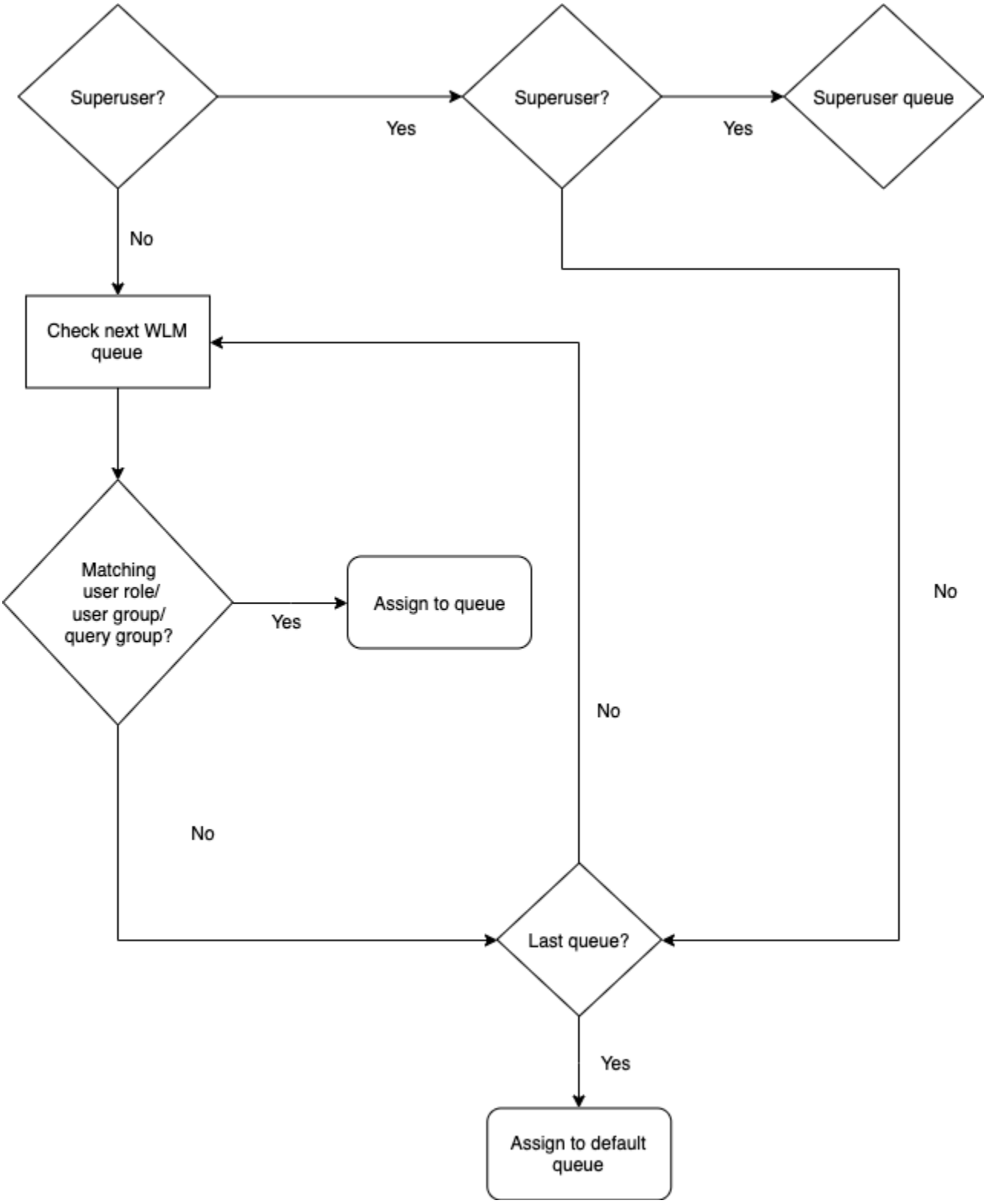
Com o Amazon Redshift, é possível controlar a alocação de recursos de memória e CPU para consultas de usuários definindo regras de atribuição de fila em uma configuração do gerenciamento de workloads (WLM). A seção a seguir descreve como criar e gerenciar regras de atribuição de fila

do WLM para obter uma alocação eficiente de recursos e atender a acordos de serviço referentes a diversas workloads no Amazon Redshift.

Quando um usuário executa uma consulta, o WLM a atribui à primeira fila correspondente, com base nas regras de atribuição de fila do WLM:

1. Se um usuário estiver conectado como superusuário e executar uma consulta no grupo de consultas identificado, a consulta será atribuída à fila de superusuários.
2. Se um usuário fizer parte de um perfil, pertencer a um grupo de usuários listado ou executar uma consulta dentro de um grupo de consultas listado, a consulta será atribuída à primeira fila correspondente.
3. Se não atender a nenhum critério, a consulta será atribuída à fila padrão, a última fila definida na configuração do WLM.

O diagrama a seguir ilustra como essas regras funcionam.

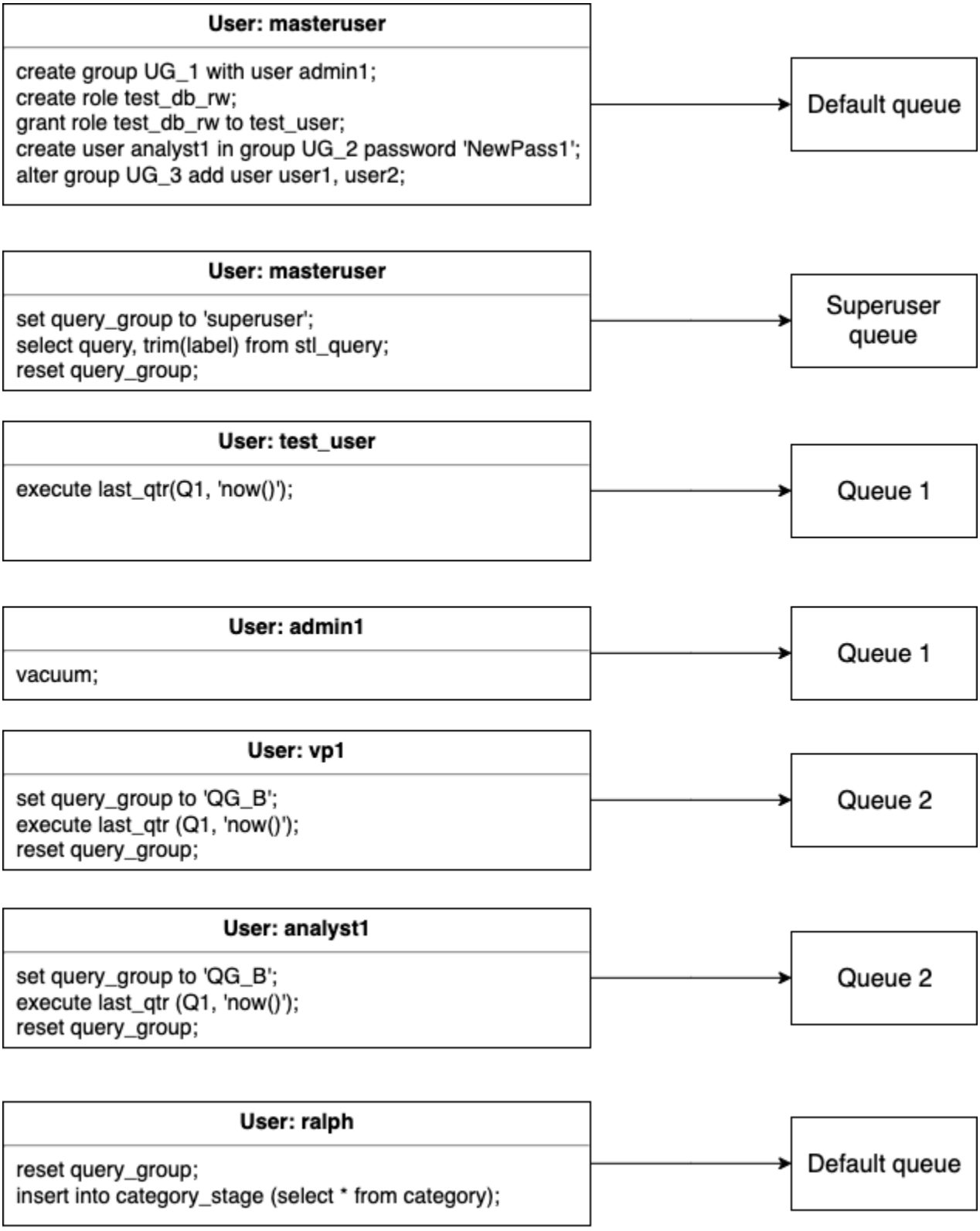

Superuser?
Yes
Superuser?
Yes
Superuser queue
No
Check next WLM queue
Matching user role/ user group/ query group?
Yes
Assign to queue
No
No
No
Last queue?
Yes
Assign to default queue

Exemplo das atribuições de fila

A tabela a seguir mostra uma configuração do WLM com a fila de superusuários e quatro filas definidas pelo usuário.

Fila	Simultaneidade	Funções de usuário	User Groups (Grupos de usuários)	Grupos de consultas
Superusuário	1			superusuário
1	5	test_db_rw	UG_1	
2	5			QG_B
3	5		UG_2	QG_C
Padrão	5			

A ilustração a seguir mostra como consultas são atribuídas às filas na tabela anterior de acordo com grupos de usuários e grupos de consultas. Para obter informações sobre como atribuir consultas a grupos de usuários e grupos de consultas no tempo de execução, consulte Atribuir consultas a filas posteriormente nesta seção.



Neste exemplo, WLM faz as seguintes atribuições:

1. O primeiro conjunto de instruções mostra três maneiras de atribuir usuários a grupos. As instruções são executadas pelo usuário `adminuser`, que não é membro de um grupo de usuários listado em nenhuma fila do WLM. Como nenhum grupo de consultas está definido, as instruções são roteadas para a fila padrão.
2. Como o usuário `adminuser` é um superusuário e o grupo de consultas está definido como `'superuser'`, a consulta está atribuída à fila de superusuários.
3. Como o usuário `test_user` recebeu o perfil `test_db_rw` listado na fila 1, a consulta é atribuída à fila 1.
4. Como o usuário `admin1` é membro do grupo de usuários listado na fila 1, a consulta é atribuída à fila 1.
5. O usuário `vp1` não é membro de nenhum grupo de usuários listado. Como o grupo de consultas é definido como `'QG_B'`, a consulta é atribuída à fila 2.
6. O usuário `analyst1` é membro do grupo de usuários listado na fila 3, mas `'QG_B'` corresponde à fila 2, logo, consulta é atribuída à fila 2.
7. Como o usuário `ralph` não é membro de nenhum grupo de usuários listado e o grupo de consultas não foi redefinido, não há fila correspondente. A consulta é atribuída à fila padrão.

Atribuir consultas a filas

Com o Amazon Redshift, é possível gerenciar a simultaneidade de workloads e priorizar as consultas atribuindo-as a filas. As filas permitem que você aloque recursos, como memória e CPU, a diferentes tipos de consulta ou usuário, garantindo que as consultas essenciais sejam priorizadas em detrimento das menos importantes. As seções a seguir descrevem como criar filas, configurar suas propriedades e atribuir consultas de entrada com base em critérios definidos por você.

Os exemplos a seguir atribuem consultas a filas de acordo com perfis de usuário, grupos de usuários e grupos de consultas.

Atribuir consultas a filas com base em perfis de usuário

Se um usuário for atribuído a um perfil e esse perfil estiver anexado a uma fila, as consultas executadas por esse usuário serão atribuídas a essa fila. O exemplo a seguir cria um perfil de usuário chamado `sales_rw` e atribui o usuário `test_user` a esse perfil.

```
create role sales_rw;
```

```
grant role sales_rw to test_user;
```

Você também pode combinar permissões de dois perfis ao conceder explicitamente um perfil a outro. A atribuição de um perfil aninhado a um usuário concede as permissões de ambos os perfis ao usuário.

```
create role sales_rw;
create role sales_ro;
grant role sales_ro to role sales_rw;
grant role sales_rw to test_user;
```

Para ver a lista de usuários que receberam perfis no cluster, consulte a tabela SVV_USER_GRANTS. Para ver a lista de perfis que receberam perfis no cluster, consulte a tabela SVV_ROLE_GRANTS.

```
select * from svv_user_grants;
select * from svv_role_grants;
```

Atribuir consultas a filas com base em grupos de usuários

Se o nome de um grupo de usuários estiver listado em uma definição de fila, as consultas executadas por membros desse grupo de usuários são atribuídas à fila correspondente. O exemplo a seguir cria grupos de usuários e adiciona usuários a grupos usando os comandos SQL CREATE USER, CREATE GROUP e ALTER GROUP.

```
create group admin_group with user admin246, admin135, sec555;
create user vp1234 in group ad_hoc_group password 'vpPass1234';
alter group admin_group add user analyst44, analyst45, analyst46;
```

Atribuir uma consulta a um grupo de consultas

É possível atribuir uma consulta a uma fila no tempo de execução atribuindo a consulta ao grupo de consultas apropriado. Use o comando SET para iniciar um grupo de consultas.

```
SET query_group TO group_label
```

Aqui, *group_label* é um rótulo de grupo de consulta listado na configuração do WLM.

Todas as consultas executadas depois do comando `SET query_group` são executadas como membros do grupo de consultas especificado até você redefinir o grupo de consultas ou encerrar a sessão de login atual. Para obter informações sobre como definir e redefinir objetos do Amazon Redshift, consulte SET e RESET na Referência de comandos SQL.

Os rótulos do grupo de consultas especificados devem ser incluídos na configuração do WLM atual; do contrário, o comando SET query_group não afeta filas de consultas.

O rótulo definido na cláusula TO é capturado nos logs de consulta, de maneira que você possa usá-lo na solução de problemas. Para obter informações sobre o parâmetro de configuração query_group, consulte query_group na Referência de configuração.

O exemplo a seguir executa duas consultas como parte de 'priority' do grupo de consultas e redefine o grupo de consultas.

```
set query_group to 'priority';
select count(*)from stv_blocklist;
select query, elapsed, substring from svl_qlog order by query desc limit 5;
reset query_group;
```

Atribuir consultas à fila de superusuários

Para atribuir uma consulta à fila de superusuário, faça logon no Amazon Redshift como superusuário e execute a consulta no grupo de superusuário. Quando você tiver terminado, redefina o grupo de consultas, de maneira que as consultas subsequentes não sejam executadas na fila de superusuários.

O exemplo a seguir atribui dois comandos a serem executados na fila de superusuários.

```
set query_group to 'superuser';

analyze;
vacuum;
reset query_group;
```

Para exibir uma lista de superusuários, consulte a tabela de catálogo do sistema PG_USER.

```
select * from pg_user where usesuper = 'true';
```

Propriedades de configuração dinâmicas e estáticas do WLM

As propriedades de configuração do WLM são dinâmicas ou estáticas. É possível aplicar propriedades dinâmicas ao banco de dados sem uma reinicialização do cluster, mas as propriedades estáticas exigem uma reinicialização do cluster para que as alterações entrem em vigor. No entanto, se você alterar propriedades dinâmicas e estáticas ao mesmo tempo, será necessário reinicializar o cluster para que todas as alterações feitas na propriedade entrem em vigor. Isso ocorrerá independentemente se as propriedades alteradas forem dinâmicas ou estáticas.

Enquanto as propriedades dinâmicas estiverem sendo aplicadas, o status do cluster será `modifying`. A alternância entre o WLM automático e o WLM manual é uma alteração estática e exige que a reinicialização do cluster entre em vigor.

A tabela a seguir indica quais propriedades do WLM são dinâmicas ou estáticas ao usar o WLM automático ou o WLM manual

Propriedade do WLM	WLM automático	WLM manual
Grupos de consultas	Dinâmico	Estático
Curinga do grupo de consultas	Dinâmico	Estático
User groups (Grupos de usuários)	Dinâmico	Estático
Curinga do grupo de usuários	Dinâmico	Estático
Funções de usuário	Dinâmico	Estático
Caractere curinga de perfil de usuário	Dinâmico	Estático
Simultaneidade no principal	Não aplicável	Dinâmico
Modo de escalabilidade da simultaneidade	Dinâmico	Dinâmico
Habilitar a aceleração de consultas breves	Não aplicável	Dinâmico

Propriedade do WLM	WLM automático	WLM manual
Tempo máximo de execução para consultas breves	Dinâmico	Dinâmico
Porcentagem de memória a ser usada	Não aplicável	Dinâmico
Tempo limite	Não aplicável	Dinâmico
Prioridade	Dinâmico	Não aplicável
Adicionar ou remover filas	Dinâmico	Estático

Se você adicionar uma regra de monitoramento de consultas (QMR) ou modificar ou excluir uma QMR existente, a alteração ocorrerá automaticamente, sem a necessidade de reiniciar o cluster.

> **Note**
>
> Ao usar o WLM manual, se o valor de tempo limite for alterado, o valor novo será aplicado a qualquer consulta que iniciar a execução depois que o valor for alterado. Se a simultaneidade ou o percentual de memória a ser usado forem alterados, o Amazon Redshift muda para a nova configuração dinamicamente. Portanto, as consultas em execução no momento não serão afetadas pela alteração. Para obter mais informações, consulte Alocação de memória dinâmica do WLM.

Tópicos

- Alocação de memória dinâmica do WLM
- Exemplo do WLM dinâmico

Alocação de memória dinâmica do WLM

Em cada fila, o WLM cria vários slots de consulta iguais ao nível de simultaneidade da fila. O valor de memória alocada a um slot de consulta equivale à porcentagem de memória alocada à fila dividida pela contagem de slots. Se você alterar a alocação de memória ou simultaneidade, o Amazon Redshift gerencia dinamicamente a transição para a nova configuração WLM. Portanto, as consultas

ativas poderão ser executadas até a conclusão usando a quantidade memória alocada atualmente. Ao mesmo tempo, o Amazon Redshift garante que o uso total da memória nunca exceda 100 por cento da memória disponível.

O gerenciador do workload usa o processo a seguir para gerenciar a transição:

1. O WLM recalcula a alocação de memória para cada novo slot de consulta.
2. Se um slot de consulta não estiver sendo usado ativamente por uma consulta em execução, o WLM removerá o slot, o que disponibiliza essa memória para novos slots.
3. Se um slot de consulta estiver ativamente em uso, o WLM aguardará a conclusão da consulta.
4. À medida que as consultas ativas são concluídas, os slots vazios são removidos e a memória associada é liberada.
5. À medida que a memória suficiente é disponibilizada para adicionar um ou mais slots, novos slots são adicionados.
6. Quando todas as consultas que estavam em execução no momento da alteração são concluídas, a contagem de slots iguala o novo nível de simultaneidade, e a transição para a nova configuração do WLM é concluída.

Na verdade, as consultas que estão em execução quando a alteração acontecer continuam a usar a alocação de memória original. As consultas enfileiradas quando a alteração acontecer são roteadas para novos slots à medida que se tornarem disponíveis.

Se as propriedades dinâmicas do WLM forem alteradas durante o processo de transição, o WLM começará imediatamente a transição para a nova configuração, começando pelo estado atual. Para exibir o status da transição, consulte a tabela do sistema STV_WLM_SERVICE_CLASS_CONFIG.

Exemplo do WLM dinâmico

Com o Amazon Redshift, é possível gerenciar automaticamente a distribuição de workloads e a alocação de recursos nos clusters do Amazon Redshift usando o gerenciamento de workloads dinâmico (WLM dinâmico). O WLM dinâmico é um exemplo de configuração do gerenciamento de workloads (WLM) que ajusta dinamicamente as alocações de memória com base nas demandas de workload, permitindo simultaneidade e desempenho ideais. A seção a seguir apresenta detalhes sobre a implementação e configuração do WLM dinâmico para clusters do Amazon Redshift.

Suponhamos que o WLM do cluster seja configurado com duas filas usando as propriedades dinâmicas a seguir.

Fila	Simultaneidade	% de memória a ser usada
1	4	50%
2	4	50%

Agora suponhamos que o cluster tenha 200 GB de memória disponível para processamento de consulta. (Esse número é arbitrário e usado somente para fins ilustrativos.) Como a equação a seguir mostra, cada slot recebe 25 GB.

```
(200 GB * 50% ) / 4 slots  = 25 GB
```

Em seguida, você altera o WLM para usar as propriedades dinâmicas a seguir.

Fila	Simultaneidade	% de memória a ser usada
1	3	75%
2	4	25%

Como a equação a seguir mostra, a nova alocação de memória para cada slot na fila 1 é 50 GB.

```
(200 GB * 75% ) / 3 slots = 50 GB
```

Suponhamos que as consultas A1, A2, A3 e A4 estejam em execução quando a nova configuração for aplicada e as consultas B1, B2, B3 e B4 estejam enfileiradas. O WLM reconfigura dinamicamente os slots de consulta da maneira a seguir.

Etapa	Execução de consultas	Contagem de slots atual	Contagem de slots de destino	Memória alocada	Memória disponível
1	A1, A2, A3, A4	4	0	100 GB	50 GB
2	A2, A3, A4	3	0	75 GB	75 GB

Etapa	Execução de consultas	Contagem de slots atual	Contagem de slots de destino	Memória alocada	Memória disponível
3	A3, A4	2	0	50 GB	100 GB
4	A3, A4, B1	2	1	100 GB	50 GB
5	A4, B1	1	1	75 GB	75 GB
6	A4, B1, B2	1	2	125 GB	25 GB
7	B1, B2	0	2	100 GB	50 GB
8	B1, B2, B3	0	3	150 GB	0 GB

1. O WLM recalcula a alocação de memória para cada slot de consulta. Originalmente, a fila 1 recebeu 100 GB. Como a nova fila tem uma alocação total de 150 GB, a nova fila tem imediatamente 50 GB disponíveis. Agora a fila 1 está usando quatro slots, e o novo nível de simultaneidade tem três slots, logo, nenhum slot novo é adicionado.
2. Quando uma consulta é concluída, o slot é removido e 25 GB são liberados. Agora a fila 1 tem três slots e 75 GB de memória disponíveis. A nova configuração precisa de 50 GB para cada slot novo, mas o novo nível de simultaneidade tem três slots, logo, nenhum slot é adicionado.
3. Quando uma segunda consulta é concluída, o slot é removido e 25 GB são liberados. Agora a fila 1 tem dois slots e 100 GB de memória livre.
4. Um novo slot é adicionado usando-se 50 GB de memória livre. Agora a fila 1 tem três slots e 50 GB de memória livre. Agora as consultas enfileiradas podem ser roteadas para o novo slot.
5. Quando uma terceira consulta é concluída, o slot é removido e 25 GB são liberados. Agora a fila 1 tem dois slots e 75 GB de memória livre.
6. Um novo slot é adicionado usando-se 50 GB de memória livre. Agora a fila 1 tem três slots e 25 GB de memória livre. Agora as consultas enfileiradas podem ser roteadas para o novo slot.
7. Quando uma quarta consulta é concluída, o slot é removido e 25 GB são liberados. Agora a fila 1 tem dois slots e 50 GB de memória livre.
8. Um novo slot é adicionado usando-se os 50 GB de memória livre. Agora a fila 1 tem três slots com 50 GB cada, e toda a memória disponível foi alocada.

A transição foi concluída e todos os slots de consulta estão disponíveis para consultas enfileiradas.

Regras de monitoramento de consulta do WLM

No gerenciador de workload (WLM) do Amazon Redshift, as regras de monitoramento de consulta definem limites de performance baseados em métricas para filas WLM e especificam qual ação tomar quando uma consulta ultrapassa esses limites. Por exemplo, para uma fila dedicada a consultas rápidas, convém criar uma regra que cancele consultas executadas por mais de 60 segundos. Para acompanhar consultas mal projetadas, convém ter outra regra que registre consultas que contenham loops aninhados.

Você define regras de monitoramento de consultas como parte da configuração do Workload Management (WLM – Gerenciamento do workload). É possível definir até 25 regras para cada fila, com um limite de 25 regras para todas as filas. Cada regra inclui até três condições, ou predicados, e uma ação. Um predicado consiste em uma métrica, uma condição de comparação (=, < ou >) e um valor. Se todos os predicados para qualquer regra forem atendidos, a ação dessa regra será disparada. As ações de regra possíveis são registrar em log, saltar e anular, conforme abordado a seguir.

As regras em uma determinada fila se aplicam somente a consultas em execução nessa fila. A regra independe de outras regras.

O WLM avalia métricas a cada 10 segundos. O Amazon Redshift aplica regras de monitoramento de consultas no nível de consulta secundária quando as consultas são reescritas automaticamente. Se mais de uma regra for acionada, o WLM escolherá a regra com a ação mais grave. Se a ação para duas regras tiver a mesma gravidade, o WLM executará as regras em ordem alfabética, com base no nome da regra. Se a ação for saltar ou anular, a ação será registrada, e a consulta será removida da fila. Se a ação for registrar em log, a consulta continuará sendo executada na fila. O WLM inicia somente uma ação de registrar em log por consulta por regra. Se a fila contiver outras regras, essas regras permanecerão em vigor. Se a ação for saltar e a consulta for roteada para outra fila, as regras para a nova fila se aplicarão. Para obter mais informações sobre ações de monitoramento e rastreamento realizadas em consultas específicas, consulte o conjunto de exemplos em Aceleração de consulta breve.

Quando todos os predicados de uma regra são atendidos, o WLM grava uma linha na tabela do sistema STL_WLM_RULE_ACTION. Além disso, o Amazon Redshift registra métricas de consulta para consultas em execução no momento em STV_QUERY_METRICS. As métricas para consultas concluídas são armazenadas em STL_QUERY_METRICS.

> **Note**
>
> Para o Amazon Redshift sem servidor, é possível configurar filas de consultas e regras de monitoramento usando o parâmetro `wlm_json_configuration`. Isso permite criar várias filas com diferentes perfis de usuário, grupos de consultas e regras de monitoramento. Para ter mais informações sobre como configurar filas de consultas sem servidor, consulte Configurar filas de consultas no Guia de gerenciamento do Amazon Redshift.

Definir uma regra de monitoramento de consultas

Você cria regras de monitoramento de consulta como parte da configuração do WLM, estabelecida como parte da definição do grupo de parâmetro do cluster.

Você pode criar regras usando o Console de gerenciamento da AWS ou programaticamente usando JSON.

> **Note**
>
> Se você optar por criar regras programaticamente, será altamente recomendável usar o console para gerar o JSON incluído por você na definição do parameter group. Para ter mais informações, consulte Criar uma regra de monitoramento de consulta e Configuring Parameter Values Using the AWS CLI no Guia de gerenciamento do Amazon Redshift.

Para definir uma regra de monitoramento da consulta, você especifica os seguintes elementos:

- Um nome de regra – Os nomes de regra devem ser exclusivos na configuração do WLM. Os nomes de regra podem ter até 32 caracteres alfanuméricos ou sublinhados e não podem conter espaços ou aspas. Pode haver até 25 regras por fila e o limite total para todas as filas é de 25 regras.
- Um ou mais predicados – Você pode ter até três predicados por regra. Se todos os predicados para qualquer regra forem atendidos, a ação associada será disparada. Um predicado é definido por um nome de métrica, um operador (=, < ou >) e um valor. Um exemplo é `query_cpu_time > 100000`. Para obter uma lista de métricas e exemplos de valores para métricas diferentes, consulte Métricas de monitoramento de consultas para o Amazon Redshift provisionado posteriormente nesta seção.

- Uma ação – Se mais de uma regra for acionada, o WLM escolherá a regra com a ação mais severa. As ações possíveis, em ordem crescente de gravidade, são:
  - Log – Registra informações sobre a consulta na tabela de sistema STL_WLM_RULE_ACTION. Use a ação de registrar em log quando você quiser gravar somente um registro de log. O WLM cria no máximo um log por consulta, por regra. Depois de uma ação de registrar em log, outras regras permanecerão em vigor e o WLM continuará monitorando a consulta.
  - Saltar (disponível apenas com WLM manual) – Registra a ação e salta a consulta para a próxima fila correspondente. Se não houver outra fila correspondente, a consulta será cancelada. O QMR salta somente instruções CREATE TABLE AS (CTAS) e consultas somente leitura, como instruções SELECT. Para obter mais informações, consulte Salto na fila de consultas do WLM.
  - Abort (Anular): registra a ação e encerra a consulta. A QMR não interrompe as instruções COPY e as operações de manutenção, como ALTER, ANALYZE e VACUUM.
  - Alterar prioridade (disponível apenas com WLM automático) – Altera a prioridade de uma consulta.

Para limitar o tempo de execução de consultas, recomendamos a criação de uma regra de monitoramento de consulta em vez de usar o tempo limite do WLM. Por exemplo, é possível definir `max_execution_time` como 50.000 milissegundos, conforme mostrado no trecho de JSON a seguir.

```
"max_execution_time": 50000
```

Mas, em vez disso, recomendamos que você defina uma regra de monitoramento de consultas equivalente. O seguinte exemplo demonstra uma regra de monitoramento de consultas que define `query_execution_time` como 50 segundos:

```
"rules":
[
    {
        "rule_name": "rule_query_execution",
        "predicate": [
            {
                "metric_name": "query_execution_time",
                "operator": ">",
                "value": 50
            }
        ],
        "action": "abort"
```

```
    }
]
```

Para ver as etapas de como criar ou modificar uma regra de monitoramento de consultas, consulte Criar uma regra de monitoramento de consulta e Propriedades referentes ao parâmetro de configuração do WLM no Guia de gerenciamento do Amazon Redshift.

Você pode encontrar mais informações sobre regras de monitoramento de consulta nos seguintes tópicos:

- Métricas de monitoramento de consultas para o Amazon Redshift provisionado
- Modelos de regras de monitoramento de consulta
- Criar uma regra de monitoramento de consulta
- Configurar gerenciamento do workload
- Tabelas de sistema e visualizações para regras de monitoramento de consultas

Métricas de monitoramento de consultas para o Amazon Redshift provisionado

A tabela a seguir descreve as métricas usadas em regras de monitoramento de consulta. (Essas métricas são diferentes das métricas armazenadas nas tabelas de sistema STV_QUERY_METRICS e STL_QUERY_METRICS.)

Para uma determinada métrica, o limite de performance é acompanhado no nível de consulta ou no nível de segmento. Para obter mais informações sobre segmentos e etapas, consulte Planejamento de consulta e fluxo de trabalho de execução.

> **Note**
>
> O parâmetro Tempo limite do WLM é diferente das regras de monitoramento de consulta.

Métrica	Name (Nome)	Descrição
Tempo de CPU da consulta	`query_cpu_time`	O tempo de CPU usado pela consulta (em segundos). `CPU time` é diferente de `Query execution time`.

Métrica	Name (Nome)	Descrição
		Os valores válidos são 0–999.999.
Leitura de blocos	query_blocks_read	O número de blocos de dados de 1 MB lidos pela consulta. Os valores válidos são 0–1.048.575.
Contagem de linhas da verificação	scan_row_count	O número de linhas em uma etapa de varredura. A contagem de linhas é o número total de linhas emitidas antes da filtragem das linhas marcadas para exclusão (linhas fantasmas) e antes da aplicação dos filtros de consulta definidos pelo usuário. Os valores válidos são 0–999.999.999.999.999.
Tempo de execução da consulta	query_execution_time	O tempo de execução decorrido para uma consulta (em segundos). O tempo de execução não inclui o tempo gasto esperando em uma fila. Os valores válidos são 0–86.399.
Tempo de fila de consulta	query_queue_time	Tempo gasto esperando em uma fila, em segundos. Os valores válidos são 0–86.399.
Uso da CPU	query_cpu_usage_percent	A porcentagem da capacidade de CPU usada pela consulta. Os valores válidos são 0–6.399.
Da memória para o disco	query_temp_blocks_to_disk	O espaço em disco temporário usado para gravar resultados intermediários, em blocos de 1 MB. Os valores válidos são 0–319.815.679.

Métrica	Name (Nome)	Descrição
Distorção da CPU	`cpu_skew`	A taxa de utilização máxima da CPU para uma fatia em relação à média de utilização da CPU para todas as fatias. Essa métrica é definida no nível do segmento. Os valores válidos são 0–99.
Distorção de E/S	`io_skew`	A taxa máxima de leitura de blocos (E/S) para uma fatia em relação à média de leitura de blocos para todas as fatias. Essa métrica é definida no nível do segmento. Os valores válidos são 0–99.
Linhas unidas	`join_row_count`	O número de linhas processadas em uma etapa de junção. Os valores válidos são 0–999.999.999.999.999.
Contagem de linhas unidas do loop aninhado	`nested_loop_join_row_count`	O número de linhas em uma união de loop aninhado. Os valores válidos são 0–999.999.999.999.999.
Contagem de linhas de retorno	`return_row_count`	O número de linhas retornado pela consulta. Os valores válidos são 0–999.999.999.999.999.
Tempo de execução do segmento	`segment_execution_time`	O tempo de execução decorrido para um único segmento, em segundos. Para evitar ou reduzir erros de amostragem, inclua `segment_execution_time > 10` nas regras. Os valores válidos são 0–86.388.

Métrica	Name (Nome)	Descrição
Contagem de linhas de verificação do Spectrum	`spectrum_scan_row_count`	O número de linhas de dados no Amazon S3 verificados por uma consulta do Amazon Redshift Spectrum. Os valores válidos são 0–999.999.999.999.999.
Tamanho de verificação do Spectrum	`spectrum_scan_size_mb`	O tamanho dos dados no Amazon S3, em MB, verificados por uma consulta do Amazon Redshift Spectrum. Os valores válidos são 0–999.999.999.999.999.
Prioridade da consulta	`query_priority`	A prioridade da consulta. Os valores válidos são `HIGHEST`, `HIGH`, `NORMAL`, `LOW` e `LOWEST`. Ao comparar `query_priority` usando operadores maior que (>) e menor que (<), `HIGHEST` será maior que `HIGH`, `HIGH` será maior que `NORMAL` e assim por diante.

Note

- Não há suporte para a ação de salto com o predicado `query_queue_time`. Ou seja, as regras definidas para saltar quando um predicado `query_queue_time` é atendido são ignoradas.
- Os tempos de execução de segmento curto podem resultar em erros de amostragem com algumas métricas, como `io_skew` e `query_cpu_usage_percent`. Para evitar ou reduzir erros de amostragem, inclua o tempo de execução do segmento nas regras. Um bom ponto de partida é `segment_execution_time > 10`.

A exibição SVL_QUERY_METRICS mostra as métricas de consultas concluídas. A exibição SVL_QUERY_METRICS_SUMMARY mostra os valores máximos de métricas de consultas

concluídas. Use os valores nessas exibições como um auxílio para determinar os valores limite para definir regras de monitoramento de consultas.

Métricas de monitoramento de consultas para o Amazon Redshift Serverless

A tabela a seguir descreve as métricas usadas em regras de monitoramento de consulta para o Amazon Redshift Serverless.

Métrica	Nome de predicado do WLM	Name (Nome)	Descrição
Leitura de blocos	`query_blocks_read`	`max_query_blocks_read`	O número de blocos de dados de 1 MB lidos pela consulta. Os valores válidos são 0–1.048.575.
Contagem de linhas da verificação	`scan_row_count`	`max_scan_row_count`	O número de linhas em uma etapa de varredura. A contagem de linhas é o número total de linhas emitidas antes da filtragem das linhas marcadas para exclusão (linhas fantasmas) e antes da aplicação dos filtros de consulta definidos pelo usuário. Os valores válidos são 0–999.999 .999.999.999.
Tempo de execução da consulta	`query_execution_time`	max_query _execution_time	O tempo de execução decorrido para uma consulta (em segundos) . O tempo de execução não inclui o tempo gasto esperando em uma fila. Se uma consulta exceder o tempo de execução definido, o Amazon Redshift Serverless interromperá a consulta.

Métrica	Nome de predicado do WLM	Name (Nome)	Descrição
			Os valores válidos são 0–86.399.
Tempo de fila de consulta	`query_queue_time`	`max_query_queue_time`	Tempo gasto esperando em uma fila, em segundos. Os valores válidos são 0–86.399.
Da memória para o disco	`query_temp_blocks_to_disk`	`max_query_temp_blocks_to_disk`	O espaço em disco temporário usado para gravar resultados intermediários, em blocos de 1 MB. Os valores válidos são 0–319.815.679.
Linhas unidas	`join_row_count`	`max_join_row_count`	O número de linhas processadas em uma etapa de junção. Os valores válidos são 0–999.999.999.999.
Contagem de linhas unidas do loop aninhado	`nested_loop_join_row_count`	`max_nested_loop_join_row_count`	O número de linhas em uma união de loop aninhado. Os valores válidos são 0–999.999.999.999.

Note

- Não há suporte para a ação de salto com o predicado `max_query_queue_time`. Ou seja, as regras definidas para saltar quando um predicado `max_query_queue_time` é atendido são ignoradas.
- Os tempos de execução de segmento curto podem resultar em erros de amostragem com algumas métricas, como `max_io_skew` e `max_query_cpu_usage_percent`.

Para o Amazon Redshift sem servidor, é possível configurar filas de consultas e regras de monitoramento usando o parâmetro `wlm_json_configuration`. Isso permite criar várias filas com diferentes perfis de usuário, grupos de consultas e regras de monitoramento usando as métricas listadas acima. Para ter mais informações sobre como configurar filas de consultas sem servidor, consulte Regras de monitoramento de consulta do WLM no Guia de gerenciamento do Amazon Redshift.

Modelos de regras de monitoramento de consulta

Ao adicionar uma regra usando o console do Amazon Redshift, você pode optar por criar uma regra com base em um modelo predefinido. O Amazon Redshift cria uma nova regra com um conjunto de predicados e preenche os predicados com valores padrão. A ação padrão é registrar em log. Você pode modificar os predicados e a ação para atender ao caso de uso.

A tabela a seguir lista os modelos disponíveis.

Nome do modelo	Predicados	Descrição
Junção de loop aninhado	`nested_loop_join_row_count > 100`	Uma união de loop aninhado pode indicar um predicado de união incompleto, o que normalmente resulta em um conjunto de retorno muito grande (um produto cartesiano). Use uma contagem de linhas baixa para encontrar uma consulta potencialmente perdida com antecedência.
Consulta retorna um número elevado de itens	`return_row_count > 1000000`	Se dedicar uma fila a consultas simples, rápidas, você poderá incluir uma regra que encontra consultas que retornam uma contagem de linhas elevada. O modelo usa um padrão de 1 milhão de linhas. Para alguns sistemas, convém levar em consideração um milhão de linhas um número elevado ou, em um sistema maior, um bilhão ou mais de linhas pode ser alto.

Nome do modelo	Predicados	Descrição
União com um número elevado de itens	`join_row_count > 1000000000`	Uma etapa de união que envolve um número excepcionalmente elevado de linhas pode indicar uma necessidade de filtros mais restritivos. O modelo usa um padrão de 1 bilhão de linhas. Para uma fila ad hoc (única) destinada a consultas rápidas e simples, você pode usar um número menor.
Uso elevado do disco durante a gravação de resultados intermediários	`query_temp_blocks_to_disk > 100000`	Ao executar consultas usando mais do que a RAM do sistema disponível, o mecanismo de execução de consulta grava resultados intermediários no disco (memória derramada). Normalmente, essa condição é o resultado de uma consulta fictícia, que normalmente também é a consulta que usa o maior espaço em disco. O limite aceitável para uso do disco varia com base no tipo de nó do cluster e no número de nós. O modelo usa um padrão de 100.000 blocos, ou 100 GB. Para um cluster pequeno, convém usar um número menor.
Consulta demorada com distorção de E/S elevada	`segment_execution_time > 120` e da `io_skew > 1.30`	A distorção de E/S ocorre quando uma fatia de nó apresenta uma taxa de E/S muito mais alta do que as outras fatias. Como ponto de partida, uma distorção de 1,30 (1,3 vez a média) é considerada alta. A distorção de E/S elevada nem sempre é um problema, mas quando combinada com um tempo de consulta demorado, pode indicar um problema no estilo de distribuição ou na chave de classificação.

Tabelas de sistema e visualizações para regras de monitoramento de consultas

Quando todos os predicados de uma regra são atendidos, o WLM grava uma linha na tabela do sistema STL_WLM_RULE_ACTION. Essa linha contém detalhes sobre a consulta que acionou a regra a ação resultante.

Além disso, o Amazon Redshift registra as métricas de consulta nas tabelas e visualizações do sistema a seguir.

- A tabela STV_QUERY_METRICS exibe as métricas de consultas em execução no momento.
- A tabela STL_QUERY_METRICS registra as métricas de consultas concluídas.
- A exibição SVL_QUERY_METRICS mostra as métricas de consultas concluídas.
- A exibição SVL_QUERY_METRICS_SUMMARY mostra os valores máximos de métricas de consultas concluídas.

Tabelas de sistema e visualizações do WLM

O WLM configura filas de consultas de acordo com classes de serviço do WLM, que são definidas internamente. O Amazon Redshift cria várias filas internas de acordo com essas classes de serviço com as filas definidas na configuração do WLM. Os termos fila e classe de serviço costumam ser usados alternadamente nas tabelas de sistema. A fila de superusuários usa classe de serviço 5. As filas definidas pelo usuário usam classe de serviço 6 e maiores.

Você pode exibir o status de consultas, filas e classes de serviço usando tabelas de sistema específicas do WLM. Consulte as seguintes tabelas de sistema para fazer o seguinte:

- Exiba quais consultas estão sendo acompanhadas e quais recursos são alocados pelo gerenciador do workload.
- Consulte a qual fila uma consulta foi atribuída.
- Exiba o status de uma consulta que esteja sendo acompanhada no momento pelo gerenciador do workload.

Nome da tabela	Descrição
STL_WLM_ERROR	Contém um log de eventos de erro relacionados ao WLM.

Nome da tabela	Descrição
STL_WLM_QUERY	Lista consultas acompanhadas pelo WLM.
STV_WLM_CLASSIFICATION_CONFIG	Mostra as regras de classificação atuais do WLM.
STV_WLM_QUERY_QUEUE_STATE	Registra o estado atual das filas de consulta.
STV_WLM_QUERY_STATE	Fornece um snapshot do estado atual de consultas acompanhadas pelo WLM.
STV_WLM_QUERY_TASK_STATE	Contém o estado atual de tarefas de consulta.
STV_WLM_SERVICE_CLASS_CONFIG	Registra as configurações da classe de serviço para o WLM.
STV_WLM_SERVICE_CLASS_STATE	Contém o estado atual das classes de serviço.
STL_WLM_RULE_ACTION	Registra detalhes sobre as ações resultantes das regras de monitoramento de consultas de WLM associadas às filas definidas pelo usuário.
STV_WLM_QMR_CONFIG	Registra a configuração das regras de monitoramento de consultas (QMR) para o WLM.

Você usa o ID da tarefa para acompanhar uma consulta nas tabelas de sistema. O seguinte exemplo mostra como obter o ID da tarefa da consulta de usuário enviada mais recentemente:

```
select task from stl_wlm_query where exec_start_time =(select max(exec_start_time) from
 stl_wlm_query);

task
------
137
(1 row)
```

O exemplo a seguir exibe consultas atualmente em execução ou esperando em diversas classes de serviço (filas). Esta consulta é útil para rastrear o workload simultâneo geral do Amazon Redshift:

```
select * from stv_wlm_query_state order by query;


xid |task|query|service_| wlm_start_  |  state  |queue_ | exec_
    |    |     |class   | time        |         |time   | time
----+----+-----+--------+-------------+---------+-------+--------
2645| 84 | 98  | 3      | 2010-10-... |Returning|   0   | 3438369
2650| 85 | 100 | 3      | 2010-10-... |Waiting  |   0   | 1645879
2660| 87 | 101 | 2      | 2010-10-... |Executing|   0   | 916046
2661| 88 | 102 | 1      | 2010-10-... |Executing|   0   | 13291
(4 rows)
```

IDs da classe de serviço do WLM

A tabela a seguir lista os IDs atribuídos a classes de serviço.

ID	Classe de serviço
1-4	Reservada para uso do sistema.
5	Usado pela fila de usuários avançados.
6–13	Usados por filas de WLM manual que estão definidas na configuração do WLM.
14	Usado pela aceleração de consultas breves.
15	Reservado para atividades de manutenção executadas pelo Amazon Redshift.
100–107	Usado pela fila de WLM automático quando auto_wlm for true.

Segurança do banco de dados

Você gerencia a segurança do banco de dados controlando quais usuários têm acesso a quais objetos do banco de dados. É possível atribuir funções ou grupos aos usuários, e as permissões que você concede a usuários, funções ou grupos determinam quais objetos do banco de dados eles podem acessar.

Tópicos

- Visão geral da segurança do Amazon Redshift
- Permissões de usuário padrão do banco de dados
- Superusuários
- Usuários
- Grupos
- Esquemas
- Regras de controle de acesso com base em função (RBAC)
- Segurança por linha
- Segurança de metadados
- Mascaramento dinâmico de dados
- Permissões em escopo

O acesso a objetos de banco de dados depende das permissões que você concede a usuários ou grupos. As seguintes diretrizes resumem como a segurança de banco de dados funciona:

- Por padrão, as permissões são concedidas somente ao proprietário do objeto.
- Os usuários do banco de dados Amazon Redshift são usuários nomeados que podem se conectar a um banco de dados. Um usuário recebe permissões de duas maneiras: explicitamente, com essas permissões atribuídas diretamente à conta, ou implicitamente, sendo um membro de um grupo que recebe permissões.
- Os grupos são coleções de usuários que podem receber permissões coletivamente tendo em vista uma manutenção de segurança mais simplificada.
- Esquemas são coleções de tabelas de bancos de dados e outros objetos de banco de dados. Os esquemas são semelhantes a diretórios de sistema de arquivos, exceto pelo fato de que

esquemas não podem ser aninhados. Os usuários podem receber acesso a um único esquema ou a vários.

Além disso, o Amazon Redshift emprega os seguintes recursos para oferecer um controle mais preciso sobre quais usuários têm acesso a quais objetos do banco de dados:

- Com o controle de acesso baseado em perfis (RBAC), é possível atribuir permissões a perfis que podem ser aplicados aos usuários, permitindo controlar permissões para grandes grupos de usuários. Ao contrário dos grupos, perfis podem herdar permissões de outros perfis.

  Com a segurança por linha (RLS), é possível definir políticas que restringem o acesso às linhas de sua escolha e, em seguida, aplicar essas políticas a usuários ou grupos.

  O mascaramento dinâmico de dados (DDM) protege ainda mais seus dados ao transformá-los no runtime de consulta para que você possa permitir que os usuários acessem os dados sem expor detalhes confidenciais.

Para obter exemplos de implementação de segurança, consulte Exemplo para controlar acesso de usuário e grupo.

Para obter mais informações sobre como proteger seus dados, consulte “Segurança no Amazon Redshift” no Guia de gerenciamento de clusters do Amazon Redshift.

Visão geral da segurança do Amazon Redshift

A segurança do banco de dados do Amazon Redshift é diferente de outros tipos de segurança do Amazon Redshift. Além da segurança do banco de dados, que é descrita nesta seção, o Amazon Redshift fornece estes recursos para gerenciar a segurança:

- Credenciais de login: o acesso ao Console de Gerenciamento da AWS do Amazon Redshift é controlado pelos privilégios da sua conta da AWS. Para obter mais informações, consulte Credenciais de login.
- Gerenciamento de acesso — Para controlar o acesso a recursos específicos do Amazon Redshift, defina contas do AWS Identity and Access Management (IAM). (Para obter mais informações, consulte Controlar o acesso aos recursos do Amazon Redshift.
- Grupos de segurança do cluster — Para conceder acesso de entrada de outros usuários a um cluster do Amazon Redshift, você define um grupo de segurança de cluster e o associa a um

cluster. Para obter mais informações, consulte Grupos de segurança do cluster do Amazon Redshift.

- VPC — Para proteger o acesso ao seu cluster usando um ambiente de rede virtual, você pode iniciar seu cluster em uma Amazon Virtual Private Cloud (VPC). Para obter mais informações, consulte Gerenciar clusters na Virtual Private Cloud (VPC).
- Criptografia de cluster: para criptografar os dados em todas as tabelas criadas pelo usuário, você pode ativar a criptografia de cluster ao iniciar o cluster. Para obter mais informações, consulte Clusters do Amazon Redshift.
- Conexões SSL — Para criptografar a conexão entre o cliente SQL e o cluster, você pode usar a criptografia Secure Sockets Layer (SSL). Para obter mais informações, consulte Conectar-se ao cluster usando SSL.
- Criptografia dos dados de carga — para criptografar seus arquivos de dados de carga de tabela ao carregá-los no Amazon S3, você pode usar criptografia do lado do servidor ou criptografia do lado do cliente. Quando você carrega dados criptografados do lado do servidor, o Amazon S3 lida com a descriptografia de forma transparente. Quando você carrega a partir de dados criptografados do lado do cliente, o comando Amazon Redshift COPY descriptografa os dados conforme carrega a tabela. Para obter mais informações, consulte Carregamento de dados criptografados para Amazon S3.
- Dados em trânsito: para proteger seus dados em trânsito na nuvem da AWS, o Amazon Redshift usa SSL acelerado por hardware para se comunicar com o Amazon S3 ou Amazon DynamoDB para operações de COPY, UNLOAD, backup e restauração.
- Controle de acesso em nível de coluna: para ter controle de acesso em nível de coluna para dados no Amazon Redshift, use as declarações de concessão e revogação em nível de coluna sem ter que implementar controle de acesso baseado em exibições ou usar outro sistema.
- Controle de segurança por linha: para ter controle de segurança por linha para dados no Amazon Redshift, crie e anexe políticas a perfis ou usuários que restrinjam o acesso às linhas definidas na política.

Permissões de usuário padrão do banco de dados

Ao criar um objeto de banco de dados, você é o proprietário. Por padrão, somente um superusuário ou o proprietário de um objeto pode consultar, modificar ou conceder permissões em relação ao objeto. Para que os usuários utilizem um objeto, você deve conceder as permissões necessárias ao o usuário ou ao grupo que contém o usuário. Os superusuários do banco de dados têm as mesmas permissões de proprietários do banco de dados.

O Amazon Redshift oferece suporte às seguintes permissões: SELECT, INSERT, UPDATE, DELETE, REFERENCES, CREATE, TEMPORARY e USAGE. As permissões diferentes são associadas a tipos de objeto diferentes. Para obter informações sobre permissões do objeto de banco de dados compatíveis com o Amazon Redshift, consulte o comando GRANT.

Somente o proprietário tem permissão para modificar ou destruir um objeto.

Por padrão, todos os usuários têm permissões CREATE e USAGE no esquema PUBLIC de um banco de dados. Para impedir que usuários criem objetos no esquema PUBLIC de um banco de dados, use o comando REVOKE para remover essa permissão.

Para revogar uma permissão que tenha sido concedida anteriormente, use o comando REVOKE. As permissões do proprietário do objeto, como DROP, GRANT e REVOKE, são implícitas e não podem ser concedidas ou revogadas. Os proprietários de objeto podem revogar as próprias permissões fundamentais, por exemplo, para tornar uma tabela somente leitura para eles e para os outros. Os superusuários têm todas as permissões, independentemente de comandos GRANT e REVOKE.

Superusuários

Os superusuários do banco de dados têm as mesmas permissões dos proprietários de todos os bancos de dados.

O usuário administrador, que é o usuário que você criou ao iniciar o cluster, é um superusuário.

Você deve ser um superusuário para criar outro.

As tabelas e visualizações do sistema do Amazon Redshift são visíveis apenas para superusuários ou para todos os usuários. Somente superusuários podem consultar tabelas e exibições do sistema designadas como “visíveis para superusuários”. Para mais informações, consulte Visualizações de monitoramento de SYS.

Os superusuários podem visualizar todas as tabelas do catálogo. Para mais informações, consulte Tabelas de catálogo do sistema.

Um superusuário de banco de dados ignora todas as verificações de permissão. Os superusuários têm todas as permissões, independentemente de comandos GRANT e REVOKE. Tenha cuidado ao usar um perfil de superusuário. Recomendamos que faça a maior parte do seu trabalho com uma função que não a de superusuário. É possível criar um perfil de administrador com permissões mais restritivas. Para obter mais informações sobre como criar funções, consulte Regras de controle de acesso com base em função (RBAC).

Para criar um novo superusuário do banco de dados, faça login no banco de dados como superusuário e emita um comando CREATE USER ou um comando ALTER USER com a permissão CREATEUSER.

```
CREATE USER adminuser CREATEUSER PASSWORD '1234Admin';
ALTER USER adminuser CREATEUSER;
```

Para criar, alterar ou soltar um superusuário, use os mesmos comandos para gerenciar usuários. Para obter mais informações, consulte Criar, alterar e excluir usuários.

Usuários

Você pode criar e gerenciar usuários de banco de dados usando os comandos de SQL do Amazon Redshift CREATE USER e ALTER USER. Ou você pode configurar seu cliente SQL com drivers JDBC ou ODBC personalizados do Amazon Redshift. Eles gerenciam o processo de criação de usuários de banco de dados e senhas temporárias como parte do processo de logon do banco de dados.

Os drivers autenticam os usuários de banco de dados com base na autenticação do AWS Identity and Access Management (IAM). Se você já gerencia identidades de usuário fora da AWS, pode usar um provedor de identidades (IdP) compatível com SAML 2.0 para gerenciar o acesso aos recursos do Amazon Redshift. Use uma função do IAM para configurar o IdP e a AWS para permitir que os usuários federados gerem credenciais de banco de dados temporárias e façam logon nos bancos de dados do Amazon Redshift. Para obter mais informações, consulte Uso da autenticação do IAM para gerar credenciais do usuário do banco de dados.

Os usuários do Amazon Redshift podem ser criados e descartados somente por um superusuário do banco de dados. Os usuários são autenticados quando fazem login no Amazon Redshift. Eles podem possuir bancos de dados e objetos de banco de dados (por exemplo, tabelas). Eles também podem conceder permissões nesses objetos a usuários, grupos e esquemas para controlar quem tem acesso a qual objeto. Os usuários com direitos CREATE DATABASE podem criar bancos de dados e conceder permissões a esses bancos de dados. Os superusuários têm permissões de propriedade em todos os bancos de dados.

Criar, alterar e excluir usuários

Os usuários do banco de dados são globais em um cluster de data warehouse (e não em cada banco de dados individual).

- Para criar um usuário, utilize o comando CREATE USER.
- Para criar um superusuário, utilize o comando CREATE USER com a opção CREATEUSER.
- Para remover um usuário existente, use o comando DROP USER.
- Para alterar um usuário, por exemplo, alterar uma senha, use o comando ALTER USER.
- Para exibir uma lista de usuários, consulte a tabela do catálogo PG_USER.

```
select * from pg_user;

  usename   | usesysid | usecreatedb | usesuper | usecatupd |  passwd  | valuntil |
 useconfig
------------+----------+-------------+----------+-----------+----------+----------
+-----------
 rdsdb      |        1 | t           | t        | t         | ******** |          |
 masteruser |      100 | t           | t        | f         | ******** |          |
 dwuser     |      101 | f           | f        | f         | ******** |          |
 simpleuser |      102 | f           | f        | f         | ******** |          |
 poweruser  |      103 | f           | t        | f         | ******** |          |
 dbuser     |      104 | t           | f        | f         | ******** |          |
(6 rows)
```

Bloqueio de usuários de banco de dados

Com o Amazon Redshift, você pode proteger os usuários de banco de dados baseados em senha contra ataques de uso forçado de senha. O Amazon Redshift monitora tentativas consecutivas de login com senha com falha para cada usuário criado. Quando o número de tentativas com falha atinge um limite, o Amazon Redshift bloqueia o usuário e rejeita outras tentativas de conexão até que um superusuário ou usuário com o privilégio ALTER USER desbloqueie o usuário. Um login bem-sucedido zera o contador. Por padrão, essa proteção está habilitada.

Um superusuário define o limite para o cluster com o parâmetro de configuração max_failed_login_attempts. O padrão são cinco tentativas consecutivas com falha, e você pode definir um valor de 2 a 50. Defina esse parâmetro usando um comando ALTER SYSTEM SET, conforme mostrado no exemplo a seguir.

```
ALTER SYSTEM SET max_failed_login_attempts = 10;
```

O bloqueio de usuário se aplica somente a usuários de banco de dados baseados em senha. Os usuários federados não são afetados, incluindo usuários que se autenticam por meio do AWS

Identity and Access Management (IAM), do AWS IAM Identity Center ou de um provedor de identidades (IdP) nativo. O Amazon Redshift também não bloqueia automaticamente o usuário administrador.

Bloqueio e desbloqueio de usuários

Além do bloqueio automático, um superusuário ou um usuário com o privilégio ALTER USER pode bloquear e desbloquear um usuário manualmente.

- Para bloquear um usuário manualmente, use ALTER USER com a opção NOLOGIN. Em seguida, o Amazon Redshift rejeita as tentativas de conexão desse usuário. O bloqueio não encerra as sessões existentes do usuário em questão.

```
ALTER USER data_analyst NOLOGIN;
```

- Para desbloquear um usuário, use ALTER USER com a opção LOGIN PASSWORD e forneça uma nova senha. O desbloqueio de um usuário redefine o contador de logins com falha desse usuário.

```
ALTER USER data_analyst LOGIN PASSWORD 'NewStr0ngP@ssword';
```

Para saber mais sobre essas opções, consulte ALTER USER.

Auditoria de usuários bloqueados

Para verificar quais usuários estão bloqueados no momento, ver se cada bloqueio foi automático ou manual e revisar a atividade recente de login com falha, um superusuário pode executar o comando SHOW USER LOCKOUT. Para obter mais informações, consulte SHOW USER LOCKOUT.

Recuperação do acesso se o usuário administrador estiver bloqueado

Um superusuário ou um usuário com o privilégio ALTER USER desbloqueia um usuário bloqueado com o comando ALTER USER LOGIN PASSWORD. Se o usuário administrador estiver bloqueado e nenhum outro superusuário puder fazer login, você poderá redefinir a senha do administrador para recuperar o acesso. A redefinição da senha do administrador restaura a capacidade do usuário administrador de fazer login.

- Para um cluster provisionado, use a operação da API ModifyCluster, ou o comando equivalente do Console de gerenciamento da AWS ou da AWS CLI, para definir uma nova senha de administrador.

- Para o Amazon Redshift sem servidor, use a operação da API UpdateNamespace, ou o comando equivalente do Console de gerenciamento da AWS ou da AWS CLI, para definir uma nova senha de administrador. Você deve fornecer `adminUsername` e `adminUserPassword` na mesma solicitação.

Grupos

Grupos são coleções de usuários que recebem todas as permissões associadas ao grupo. Você pode usar grupos para atribuir permissões. Por exemplo, você pode criar grupos diferentes para vendas, administração e suporte e dar aos usuários em cada grupo o acesso apropriado aos dados necessários para o trabalho. Você pode conceder ou revogar permissões no nível do grupo, e essas alterações serão aplicadas a todos os membros do grupo, exceto superusuários.

Para exibir todos os grupos de usuários, consulte a lista de catálogo do sistema PG_GROUP:

```
select * from pg_group;
```

Por exemplo, para listar todos os usuários do banco de dados por grupo, execute o SQL a seguir.

```
SELECT u.usesysid
,g.groname
,u.usename
FROM pg_user u
LEFT JOIN pg_group g ON u.usesysid = ANY (g.grolist)
```

Criar, alterar e excluir grupos

Somente um superusuário pode criar, alterar ou descartar grupos.

Você pode realizar as seguintes ações:

- Para criar um grupo, use o comando CREATE GROUP.
- Para adicionar ou remover usuários de um grupo existente, use o comando ALTER GROUP.
- Para excluir um grupo, use o comando DROP GROUP. Esse comando descarta somente o grupo, e não os usuários membros.

Exemplo para controlar acesso de usuário e grupo

Esse exemplo cria usuários e grupos de usuários e, depois, concede a eles várias permissões para um banco de dados do Amazon Redshift que se conecta a um cliente de aplicação web. Este exemplo pressupõe três grupos de usuários: usuários regulares de uma aplicação Web, usuários avançados de uma aplicação Web e desenvolvedores Web.

Para obter informações sobre como remover um usuário de um grupo, consulte ALTER GROUP.

1. Crie os grupos aos quais os usuários serão atribuídos. O seguinte conjunto de comandos criam três grupos de usuários diferentes:

```
create group webappusers;

create group webpowerusers;

create group webdevusers;
```

2. Crie diversos usuários de banco de dados com permissões diferentes e os adicione aos grupos.

   a. Crie dois usuários e os adicione ao grupo WEBAPPUSERS:

```
create user webappuser1 password 'webAppuser1pass'
in group webappusers;

create user webappuser2 password 'webAppuser2pass'
in group webappusers;
```

   b. Crie um usuário de desenvolvedor da web e adicione-o ao grupo WEBDEVUSERS:

```
create user webdevuser1 password 'webDevuser2pass'
in group webdevusers;
```

   c. Crie um superusuário. Esse usuário terá direitos administrativos para criar outros usuários:

```
create user webappadmin  password 'webAppadminpass1'
createuser;
```

3. Crie um esquema a ser associado às tabelas de bancos de dados usadas pela aplicação Web e conceda acesso a este esquema aos diversos grupos de usuários:

   a. Crie o esquema WEBAPP:

```
create schema webapp;
```

b. Conceda permissões USAGE ao grupo WEBAPPUSERS:

```
grant usage on schema webapp to group webappusers;
```

c. Conceda permissões USAGE ao grupo WEBPOWERUSERS:

```
grant usage on schema webapp to group webpowerusers;
```

d. Conceda permissões ALL ao grupo WEBDEVUSERS:

```
grant all on schema webapp to group webdevusers;
```

Agora os usuários e os grupos básicos estão configurados. Você já pode alterar os usuários e os grupos.

4. Por exemplo, o comando a seguir altera o parâmetro search_path do WEBAPPUSER1.

```
alter user webappuser1 set search_path to webapp, public;
```

SEARCH_PATH especifica a ordem de pesquisa do esquema de objetos de banco de dados, como tabelas e funções, quando o objeto é referenciado por um nome simples sem esquema especificado.

5. Você também poderá adicionar usuários a um grupo depois ter criado o grupo, como adicionar WEBAPPUSER2 ao grupo WEBPOWERUSERS:

```
alter group webpowerusers add user webappuser2;
```

Esquemas

Um banco de dados contém um ou mais esquemas nomeados. Cada esquema em um banco de dados contém tabelas e outros tipos de objetos nomeados. Por padrão, um banco de dados tem um único esquema, chamado PUBLIC. Você pode usar esquemas para agrupar objetos de banco de dados sob um mesmo nome. Os esquemas são semelhantes a diretórios de sistema de arquivos, exceto pelo fato de que esquemas não podem ser aninhados.

É possível usar nomes de objeto de banco de dados idênticos em esquemas diferentes no mesmo banco de dados sem conflito. Por exemplo, MY_SCHEMA e YOUR_SCHEMA podem conter uma tabela chamada MYTABLE. Os usuários com as permissões necessáriss podem acessar objetos em vários esquemas em um banco de dados.

Por padrão, um objeto é criado dentro do primeiro esquema no caminho de pesquisa do banco de dados. Para obter informações, consulte Caminho de pesquisa posteriormente nesta seção.

Os esquemas podem ajudar em problemas de organização e simultaneidade em um ambiente de vários usuários das seguintes maneiras:

- Para permitir que muitos desenvolvedores trabalhem no mesmo banco de dados sem interferência entre si.
- Para organizar objetos de banco de dados em grupos lógicos para torná-los mais gerenciáveis.
- Para permitir que as aplicações tenham a possibilidade de colocar objetos em esquemas separados, de maneira que os nomes não colidam com nomes de objetos usados por outros aplicativos.

Caminho de pesquisa

O caminho de pesquisa é definido no parâmetro search_path com uma lista de nomes de esquema separados por vírgulas. O caminho de pesquisa especifica a ordem na qual os esquemas são pesquisados quando um objeto, como uma tabela ou uma função, é referenciado por um nome simples que não inclui um qualificador de esquema.

Se for criado sem especificar um esquema de destino, o objeto será adicionado ao primeiro esquema listado no caminho de pesquisa. Quando houver objetos com nomes idênticos em esquemas diferentes, um nome de objeto que não especificar um esquema irá se referir ao primeiro esquema no caminho de pesquisa que contenha um objeto com esse nome.

Para alterar o esquema padrão da sessão atual, use o comando SET.

Para obter mais informações, consulte a descrição search_path na Referência de configuração.

Criar, alterar e excluir esquemas

Qualquer usuário pode criar esquemas e alterar ou descartar esquemas próprios.

Você pode realizar as seguintes ações:

- Para criar um esquema, use o comando CREATE SCHEMA.
- Para alterar o proprietário de um esquema, use o comando ALTER SCHEMA.
- Para excluir um esquema e os objetos, use o comando DROP SCHEMA.
- Para criar uma tabela dentro de um esquema, crie a tabela com o formato nome_do_esquema.nome_da_tabela.

Para exibir uma lista de todos os esquemas, consulte a tabela de catálogo do sistema PG_NAMESPACE:

```
select * from pg_namespace;
```

Para exibir uma lista de tabelas que pertençam a um esquema, consulte a tabela de catálogo do sistema PG_TABLE_DEF. Por exemplo, a consulta a seguir retorna uma lista de tabelas no esquema PG_CATALOG.

```
select distinct(tablename) from pg_table_def
where schemaname = 'pg_catalog';
```

Permissões baseadas em esquemas

As permissões baseadas em esquema são determinadas pelo proprietário do esquema:

- Por padrão, todos os usuários têm permissões CREATE e USAGE no esquema PUBLIC de um banco de dados. Para impedir que usuários criem objetos no esquema PUBLIC de um banco de dados, use o comando REVOKE para remover essa permissão.
- A menos que recebam a permissão USAGE pelo proprietário do objeto, os usuários não podem acessar objetos em esquemas que não possuam.
- Se tiverem concedido a permissão CREATE a um esquema criado por outro usuário, os usuários poderão criar objetos nesse esquema.

Regras de controle de acesso com base em função (RBAC)

Com o uso do controle de acesso baseado em função (RBAC) para gerenciar permissões de banco de dados no Amazon Redshift, você pode simplificar o gerenciamento de permissões de segurança no Amazon Redshift. Você pode proteger o acesso a dados confidenciais controlando o que os usuários podem fazer tanto em um nível amplo quanto mais preciso. Você também pode controlar o

acesso do usuário a tarefas que normalmente são restritas a superusuários. Ao atribuir permissões diferentes a funções diferentes e atribuí-las a usuários diferentes, você pode ter um controle mais granular do acesso do usuário.

Os usuários com uma função atribuída podem executar somente as tarefas especificadas pela função atribuída com a qual estão autorizados. Por exemplo, um usuário com a função atribuída que tem as permissões CREATE TABLE e DROP TABLE só está autorizado a executar essas tarefas. Você pode controlar o acesso de usuários, concedendo diferentes níveis de permissões de segurança a diferentes usuários para acessar os dados necessários para o trabalho.

O RBAC aplica o princípio de permissões mínimas aos usuários com base em seus requisitos de função, independentemente dos tipos de objetos envolvidos. A concessão e revogação de permissões é realizada no nível da função, sem a necessidade de atualizar permissões em objetos de banco de dados individuais.

Com o RBAC, você pode criar funções com permissões para executar comandos que costumavam exigir permissões de superusuário. Os usuários podem executar esses comandos, desde que sejam autorizados com uma função que inclua essas permissões. Da mesma forma, você também pode criar funções para limitar o acesso a determinados comandos e atribuir a função a superusuários ou usuários que foram autorizados com a função.

Para saber mais sobre como o RBAC funciona no Amazon Redshift, assista ao seguinte vídeo: Introducing Role-based access control (RBAC) in Amazon Redshift (Apresentação do controle de acesso baseado em função (RBAC) no Amazon Redshift).

Hierarquia de funções

Funções são coleções de permissões que você pode atribuir a um usuário ou outra função. Você pode atribuir permissões de sistema ou banco de dados a uma função. Um usuário herda permissões de uma função atribuída.

No RBAC, os usuários podem ter funções aninhadas. Você pode conceder funções a usuários e funções. Ao conceder um perfil a um usuário, você autoriza o usuário com todas as permissões que esse perfil inclui. Ao conceder uma função r1 a um usuário, você autoriza o usuário com permissões de r1. O usuário agora tem permissões de r1 e também todas as permissões existentes que ele já possui.

Ao conceder uma função (r1) a uma outra função (r2), você autoriza r2 com todas as permissões de r1. Além disso, ao conceder r2 a outra função (r3), as permissões de r3 são a combinação das

permissões de r1 e r2. A hierarquia de funções tem permissões de herdar de r2 de r1. O Amazon Redshift propaga permissões com cada autorização de função. Conceder r1 para r2 e, em seguida, r2 para r3, autoriza r3 com todas as permissões das três funções. Assim, ao conceder r3 a um usuário, o usuário tem todas as permissões das três funções.

O Amazon Redshift não permite a criação de um ciclo de autorização de função. Um ciclo de autorização de função ocorre quando uma função aninhada é atribuída de volta a uma função anteriormente na hierarquia de funções, como r3 sendo atribuído de volta a r1. Para obter mais informações sobre como criar e gerenciar atribuições de perfis, consulte Gerenciamento de funções no RBAC.

Atribuição de função

Superusuários e usuários regulares com as permissões CREATE ROLE podem usar a instrução CREATE ROLE para criar funções. Superusuários e administradores de função podem usar a instrução GRANT ROLE para conceder uma função a outras pessoas. Eles podem usar a instrução REVOKE ROLE para revogar uma função de outras pessoas e a instrução DROP ROLE para descartar funções. Os administradores de perfis incluem proprietários de perfis e usuários que receberam o perfil com a permissão ADMIN OPTION.

Somente superusuários ou administradores de função podem conceder e revogar funções. Você pode conceder ou revogar uma ou mais funções para ou de uma ou mais funções ou usuários. Use a opção WITH ADMIN OPTION na instrução GRANT ROLE para fornecer as opções de administração para todas as funções concedidas a todos os beneficiários.

O Amazon Redshift oferece suporte a diferentes combinações de atribuições de função, como conceder várias funções ou ter vários beneficiários. WITH ADMIN OPTION só se aplica a usuários, e não a funções. Da mesma forma, use a opção WITH ADMIN OPTION na instrução REVOKE ROLE para remover a função e a autorização administrativa do beneficiário. Quando usado com ADMIN OPTION, somente a autorização administrativa é revogada da função.

O exemplo a seguir revoga a autorização administrativa da função `sample_role2` de `user2`.

```
REVOKE ADMIN OPTION FOR sample_role2 FROM user2;
```

Para obter mais informações sobre como criar e gerenciar atribuições de perfis, consulte Gerenciamento de funções no RBAC.

Funções definidas pelo sistema do Amazon Redshift

O Amazon Redshift fornece algumas funções definidas pelo sistema que são definidas com permissões específicas. Funções específicas do sistema começam com um prefixo `sys:`. Somente usuários com acesso apropriado podem alterar funções definidas pelo sistema ou criar funções personalizadas definidas pelo sistema. Não é possível usar o prefixo `sys:` para uma função definida pelo sistema personalizada.

A tabela a seguir resume as funções e suas permissões.

Nome do perfil	Descrição
`sys:monitor`	Essa função tem permissão para acessar tabelas de catálogo ou sistema.
`sys:operator`	Essa função tem as permissões para acessar tabelas de catálogo ou sistema, analisar, efetuar vacuum ou cancelar consultas.
`sys:dba`	Essa função tem as permissões para criar esquemas, criar tabelas, descartar esquemas, descartar tabelas e truncar tabelas. Ela tem permissões para criar ou substitui r procedimentos armazenados, descartar procedime ntos, criar ou substitui

Nome do perfil	Descrição		
	r funções, criar ou substituir funções externas, criar exibições e descartar exibições. Além disso, essa função herda todas as permissões da função sys:opera tor.		
`sys:superuser`	Essa função tem todas as permissões de sistema compatíve is definidas em Permissões do sistema para RBAC.		

Nome do perfil	Descrição		
`sys:secadmin`	• Essa função tem as permissões para criar usuários, alterar usuários, descartar usuários, criar funções, descartar funções e conceder funções. • Esse perfil tem permissões para ativar ou desativar o RLS em uma relação e permissões para gerenciar políticas de RLS e DDM (CREATE, DROP, ATTACH, DETACH e ALTER). Além disso, as permissões EXPLAIN RLS, IGNORE RLS e EXPLAIN MASKING são concedidas a essa função por padrão. • Essa função só pode ter acesso às tabelas de usuários quando a permissão for explicitamente concedida à função.		

Usuários e perfis definidos pelo sistema para compartilhamento de dados

O Amazon Redshift cria perfis e usuários para uso interno que correspondem a unidades de compartilhamento de dados e os respectivos consumidores. O nome de cada perfil e usuário internos tem o prefixo de namespace reservado `ds:`. Eles têm o seguinte formato:

Name (Nome)	Descrição		
`ds:`*`sharename`*	Um perfil do sistema que corresponde a uma unidade de compartilhamento de dados.		
`ds:`*`sharename_consumer`*	Um usuário do sistema que corresponde a um consumidor de unidade de compartilhamento de dados.		

É criado um perfil de compartilhamento de dados para cada unidade de compartilhamento de dados. Ele contém todas as permissões atualmente concedidas à unidade de compartilhamento de dados. É criado um usuário de compartilhamento de dados para cada consumidor de uma unidade de compartilhamento de dados. Ele recebe permissão para um único perfil de compartilhamento de dados. Um consumidor adicionado a várias unidades de compartilhamento de dados terá um usuário criado para cada unidade de compartilhamento de dados.

Esses usuários e perfis são necessários para que o compartilhamento de dados funcione corretamente. Eles não podem ser modificados nem eliminados e não podem ser acessados nem usados para nenhuma tarefa realizada pelos clientes. É possível ignorá-los com segurança. Para ter mais informações, consulte Compartilhar dados entre clusters no Amazon Redshift.

> **Note**
>
> Não é possível usar o prefixo `ds:` para criar usuários ou perfis definidos pelo usuário.

Permissões do sistema para RBAC

A seguir, uma lista de permissões do sistema que podem ser concedidas ou revogadas de uma função.

Command	Você precisa ter permissão de uma das seguintes formas para executar o comando		
CREATE ROLE	• Superusuário. • Usuários com a permissão CREATE ROLE.		
DROP ROLE	• Superusuário. • Proprietário da função, que é o usuário que criou a função ou um usuário que recebeu a função com a permissão WITH ADMIN OPTION.		
CREATE USER	• Superusuário. • Usuários com a permissão CREATE USER. Esses usuários não podem criar superusuários.		
DROP USER	• Superusuário.		

Command	Você precisa ter permissão de uma das seguintes formas para executar o comando		
	• Usuários com a permissão DROP USER.		
ALTER USER	• Superusuário. • Usuários com a permissão ALTER USER. Esses usuários não podem alterar usuários para superusuários ou alterar superusuá rios para usuários. • Usuário atual que deseje alterar a própria senha.		
CREATE SCHEMA	• Superusuário. • Usuários com a permissão CREATE SCHEMA.		
DROP SCHEMA	• Superusuário. • Usuários com a permissão DROP SCHEMA. • Proprietário do esquema.		

Command	Você precisa ter permissão de uma das seguintes formas para executar o comando		
ALTER DEFAULT PRIVILEGES	• Superusuário. • Usuários com a permissão ALTER DEFAULT PRIVILEGES. • Usuários que alterem suas próprias permissões de acesso padrão. • Usuários definindo permissões para esquemas aos quais eles têm permissões de acesso.		
ACCESS CATALOG	• Superusuário. • Usuários com a permissão ACCESS CATALOG.		
ACCESS SYSTEM TABLE	• Superusuário. • Usuários com a permissão ACCESS SYSTEM TABLE.		

Command	Você precisa ter permissão de uma das seguintes formas para executar o comando		
CRIAR TABELA	• Superusuário. • Usuários com a permissão CREATE TABLE. • Usuários com a permissão CREATE para esquemas.		
DESCARTAR TABELA	• Superusuário. • Usuários com a permissão DROP TABLE. • Proprietário da tabela com a permissão USAGE no esquema.		
ALTER TABLE	• Superusuário. • Usuários com a permissão ALTER TABLE. • Proprietário da tabela com a permissão USAGE no esquema.		

Command	Você precisa ter permissão de uma das seguintes formas para executar o comando		

CREATE OR REPLACE FUNCTION	• Para CREATE FUNCTION:
	  • Superusuário.
	  • Usuários com a permissão CREATE OR REPLACE FUNCTION.
	  • Usuários com a permissão USAGE na linguagem.
	• Para REPLACE FUNCTION:
	  • Superusuário.
	  • Usuários com a permissão CREATE OR REPLACE FUNCTION.
	  • Proprietário da função.		

CREATE OR REPLACE EXTERNAL FUNCTION	• Superusuário. • Usuários com a permissão CREATE OR REPLACE EXTERNAL FUNCTION.		

Command	Você precisa ter permissão de uma das seguintes formas para executar o comando		
DROP FUNCTION	• Superusuário. • Usuários com a permissão DROP FUNCTION. • Proprietário da função.		

CREATE OR REPLACE PROCEDURE	• Para CREATE PROCEDURE:
	  • Superusuário.
	  • Usuários com a permissão CREATE OR REPLACE PROCEDURE.
	  • Usuários com a permissão USAGE na linguagem.
	• Para REPLACE PROCEDURE:
	  • Superusuário.
	  • Usuários com a permissão CREATE OR REPLACE PROCEDURE.
	  • Proprietário do procedimento.		

Command	Você precisa ter permissão de uma das seguintes formas para executar o comando		
DROP PROCEDURE	• Superusuário. • Usuários com a permissão DROP PROCEDURE. • Proprietário do procedimento.		

CREATE OR REPLACE VIEW	• Para CREATE VIEW:
	  • Superusuário.
	  • Usuários com a permissão CREATE OR REPLACE VIEW.
	  • Usuários com a permissão CREATE para esquemas.
	• Para REPLACE VIEW:
	  • Superusuário.
	  • Usuários com a permissão CREATE OR REPLACE VIEW.
	  • Proprietário da exibição.		

Command	Você precisa ter permissão de uma das seguintes formas para executar o comando		
DROP VIEW	• Superusuário. • Usuários com a permissão DROP VIEW. • Proprietário da exibição.		
CREATE MODEL	• Superusuário. • Usuários com a permissão do sistema CREATE MODEL, que devem ser capazes de ler a relação de CREATE MODEL. • Usuários com a permissão CREATE MODEL.		
DROP MODEL	• Superusuário. • Usuários com a permissão DROP MODEL. • Proprietário do modelo. • Proprietário do esquema.		

Command	Você precisa ter permissão de uma das seguintes formas para executar o comando		
CREATE DATASHARE	• Superusuário. • Usuários com a permissão CREATE DATESHARE. • Proprietário do banco de dados.		

Command	Você precisa ter permissão de uma das seguintes formas para executar o comando		
ALTER DATASHARE	• Superusuário. • Usuários com a permissão ALTER DATASHARE. • Usuários que tenham a permissão ALTER ou ALL na unidade de compartil hamento de dados. • Para adicionar objetos específicos a uma unidade de compartilhamento de dados, esses usuários devem ter a permissão nos objetos. Os usuários devem ser os proprietá rios dos objetos ou ter permissões SELECT, USAGE ou ALL nos objetos.		

Command	Você precisa ter permissão de uma das seguintes formas para executar o comando		
DROP DATASHARE	• Superusuário. • Usuários com a permissão DROP DATESHARE. • Proprietário do banco de dados.		
CREATE LIBRARY	• Superusuário. • Usuários com a permissão CREATE LIBRARY ou com a permissão da linguagem especific ada.		
DROP LIBRARY	• Superusuário. • Usuários com a permissão DROP LIBRARY. • Proprietário da biblioteca.		

Command	Você precisa ter permissão de uma das seguintes formas para executar o comando		
ANALYZE	• Superusuário. • Usuários com a permissão ANALYZE. • Proprietário da relação. • Proprietário do banco de dados com o qual a tabela é compartilhada.		
CANCEL	• Superusuário que cancele sua própria consulta. • Superusuário que cancele a consulta de um usuário. • Usuários com a permissão CANCEL que cancelem a consulta de um usuário. • Usuário que cancele sua própria consulta.		

Command	Você precisa ter permissão de uma das seguintes formas para executar o comando		
TRUNCATE TABLE	• Superusuário. • Usuários com a permissão TRUNCATE TABLE. • Proprietário da tabela.		
VACUUM	• Superusuário. • Usuários com a permissão VACUUM. • Proprietário da tabela. • Proprietário do banco de dados com o qual a tabela é compartilhada.		
IGNORE RLS	• Superusuário. • Usuários dentro da função `sys:secadmin` .		
EXPLAIN RLS	• Superusuário. • Usuários dentro da função `sys:secadmin` .		

Command	Você precisa ter permissão de uma das seguintes formas para executar o comando		
EXPLAIN MASKING	• Superusuário. • Usuários dentro da função `sys:secadmin` .		

Permissões de objetos do banco de dados

Além das permissões do sistema, o Amazon Redshift inclui permissões de objeto de banco de dados que definem as opções. Isso inclui opções como capacidade de leitura de dados em tabelas e exibições, gravação de dados, criação de tabelas e descarte de tabelas. Para obter mais informações, consulte GRANT.

Usando o RBAC, você pode atribuir permissões de objeto de banco de dados a funções, de forma semelhante a como você faz com as permissões do sistema. Em seguida, você pode atribuir funções a usuários, autorizar usuários com permissões do sistema e autorizar usuários com permissões do banco de dados.

ALTER DEFAULT PRIVILEGES para o RBAC

Use a instrução ALTER DEFAULT PRIVILEGES para definir o conjunto padrão de permissões de acesso a serem aplicadas a objetos criados no futuro pelo usuário especificado. Por padrão, os usuários podem alterar somente suas próprias permissões de acesso padrão. Com o RBAC, você pode definir as permissões de acesso padrão para funções. Para obter mais informações, consulte o comando ALTER DEFAULT PRIVILEGES.

Com o RBAC, é possível atribuir permissões de objetos de banco de dados a perfis, de maneira semelhante às permissões do sistema. Em seguida, você pode atribuir funções a usuários, autorizar usuários com permissões do sistema e/ou do banco de dados.

Considerações sobre o uso de funções no RBAC

Ao trabalhar com funções do RBAC, considere o seguinte:

- O Amazon Redshift não permite ciclos de autorizações de funções. Você não pode conceder r1 a r2 e, em seguida, conceder r2 a r1.
- O RBAC funciona tanto para objetos nativos do Amazon Redshift quanto para tabelas do Amazon Redshift Spectrum.
- Como administrador do Amazon Redshift, você pode ativar o RBAC atualizando seu cluster para o patch de manutenção mais recente para começar.
- Somente superusuários e usuários com a permissão do sistema CREATE ROLE podem criar funções.
- Somente superusuários ou administradores de função podem modificar e descartar funções.
- Um nome de função não pode ser o mesmo que um nome de usuário.
- Um nome de função não pode conter caracteres inválidos, como ":/\n."
- Um nome de função não pode ser uma palavra reservada, como PUBLIC.
- O nome da função não pode começar com o prefixo reservado para funções padrão, `sys:`.
- Você não pode descartar uma função que tenha o parâmetro RESTRICT quando ela estiver concedida a outra função. A configuração padrão é RESTRICT. O Amazon Redshift emite um erro quando você tenta eliminar uma função que tenha herdado outra função.
- Usuários que não têm permissões de administrador em uma função não podem conceder ou revogar uma função.
- O controle de acesso por perfil (RBAC) não é totalmente compatível com tabelas e visualizações do sistema. As permissões do RBAC para tabelas e visualizações do sistema não persistem por meio de atualizações, reversões para versões anteriores ou redimensionamentos. Recomendamos usar Funções definidas pelo sistema do Amazon Redshift para gerenciar a tabela do sistema e visualizar as permissões. Para ter mais informações sobre tabelas do sistema, acesse Referência de visualizações e tabelas do sistema.

Gerenciamento de funções no RBAC

Para executar as seguintes ações, use os seguintes comandos:

- Para criar uma função, use o comando CREATE ROLE.
- Para renomear uma função ou alterar o proprietário da função, use o comando ALTER ROLE.
- Para excluir uma função, use o comando DROP ROLE.
- Para conceder uma função a um usuário, use o comando GRANT.

- Para revogar uma função de um usuário, use o comando REVOKE.
- Para conceder permissões do sistema a uma função, use o comando GRANT.
- Para revogar permissões do sistema de uma função, use o comando REVOKE.

Para exibir uma lista de funções no cluster ou no grupo de trabalho, consulteSVV_ROLES.

Tutorial: Criar funções e consultar com o RBAC

Com o RBAC, você pode criar funções com permissões para executar comandos que costumavam exigir permissões de superusuário. Os usuários podem executar esses comandos, desde que sejam autorizados com uma função que inclua essas permissões.

Neste tutorial, você usa o controle de acesso por função (RBAC) para gerenciar permissões em um banco de dados que você cria. Em seguida, você se conecta ao banco de dados e o consulta usando duas funções diferentes para testar a funcionalidade de RBAC.

As duas funções que você cria e usa para consultar o banco de dados são `sales_ro` e `sales_rw`. Você cria a função `sales_ro` e consulta os dados como um usuário com a função `sales_ro`. O usuário `sales_ro` pode usar o comando SELECT, mas não pode usar o comando UPDATE. Em seguida, você cria a função `sales_rw` e consulta os dados como um usuário com a função `sales_rw`. O usuário `sales_rw` pode usar o comando SELECT e o comando UPDATE.

Além disso, você também pode criar funções para limitar o acesso a determinados comandos e atribuir a função a superusuários ou usuários.

Tarefas

- Pré-requisitos
- Etapa 1: Criar um usuário administrador
- Etapa 2: Configurar esquemas
- Etapa 3: Criar um usuário somente leitura
- Etapa 4: Consultar os dados como usuário somente leitura
- Etapa 5: Criar um usuário de leitura e gravação
- Etapa t: Consultar os dados como o usuário com a função somente leitura herdada
- Etapa 7: Conceder permissões de atualização e inserção à função de leitura e gravação

- Etapa 8: Consultar os dados como o usuário de leitura e gravação
- Etapa 9: Analisar e limpar tabelas em um banco de dados como o usuário administrador
- Etapa 10: Truncar tabelas como o usuário de leitura e gravação
- Funções do sistema para RBAC (opcional)
- Visualizações do sistema para RBAC (opcional)
- Usar a segurança por linha com o RBAC (opcional)

Pré-requisitos

- Crie um cluster ou grupo de trabalho sem servidor do Amazon Redshift carregado com o banco de dados TICKIT de exemplo. Para criar um grupo de trabalho de tecnologia sem servidor, consulte Criar um data warehouse com o Amazon Redshift sem servidor. Para criar um cluster, consulte Criar um cluster de amostra do Amazon Redshift. Para obter mais informações sobre os bancos de dados de amostra TICKIT, consulte Banco de dados de exemplo.
- Tenha acesso a um usuário com permissões de superusuário ou administrador de função. Somente superusuários ou administradores de função podem conceder ou revogar funções. Para obter mais informações sobre as permissões necessárias para RBAC, consulte Permissões do sistema para RBAC.
- Revise as Considerações sobre o uso de funções no RBAC.

Etapa 1: Criar um usuário administrador

Para configurar este tutorial, você cria uma função de administrador de banco de dados e a anexa a um usuário administrador de banco de dados nesta etapa. Você deve criar o administrador do banco de dados como superusuário ou administrador de função.

Execute todas as consultas no Editor de Consultas v2 do Amazon Redshift.

1. Para criar a função de administrador db_admin, use o exemplo a seguir.

```
CREATE ROLE db_admin;
```

2. Para criar um usuário do banco de dados chamado dbadmin, use o exemplo a seguir.

```
CREATE USER dbadmin PASSWORD 'Test12345';
```

3. Para conceder a função definida pelo sistema chamada sys:dba à função db_admin, use o exemplo a seguir. Quando a função sys:dba é concedida, o usuário dbadmin pode criar esquemas e tabelas. Para obter mais informações, consulte Funções definidas pelo sistema do Amazon Redshift.

Etapa 2: Configurar esquemas

Nesta etapa, você se conecta ao seu banco de dados como administrador do banco de dados. Em seguida, você cria dois esquemas e adiciona dados a eles.

1. Conecte-se ao banco de dados dev como o usuário dbadmin utilizando o editor de consultas v2. Para ter mais informações sobre como se conectar a um banco de dados, consulte Trabalhar com o editor de consultas v2.
2. Para criar esquemas de bancos de dados de vendas e marketing, use o exemplo a seguir.

```
CREATE SCHEMA sales;
CREATE SCHEMA marketing;
```

3. Para criar e inserir valores em tabelas em um esquema de vendas, use o exemplo a seguir.

```
CREATE TABLE sales.cat(
catid smallint,
catgroup varchar(10),
catname varchar(10),
catdesc varchar(50)
);
INSERT INTO sales.cat(SELECT * FROM category);

CREATE TABLE sales.dates(
dateid smallint,
caldate date,
day char(3),
week smallint,
month char(5),
qtr char(5),
year smallint,
holiday boolean
);
INSERT INTO sales.dates(SELECT * FROM date);

CREATE TABLE sales.events(
```

```
eventid integer,
venueid smallint,
catid smallint,
dateid smallint,
eventname varchar(200),
starttime timestamp
);
INSERT INTO sales.events(SELECT * FROM event);

 CREATE TABLE sales.sale(
salesid integer,
listid integer,
sellerid integer,
buyerid integer,
eventid integer,
dateid smallint,
qtysold smallint,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp
);
INSERT INTO sales.sale(SELECT * FROM sales);
```

4. Para criar e inserir valores em tabelas em um esquema de marketing, use o exemplo a seguir.

```
CREATE TABLE marketing.cat(
catid smallint,
catgroup varchar(10),
catname varchar(10),
catdesc varchar(50)
);
INSERT INTO marketing.cat(SELECT * FROM category);

CREATE TABLE marketing.dates(
dateid smallint,
caldate date,
day char(3),
week smallint,
month char(5),
qtr char(5),
year smallint,
holiday boolean
);
INSERT INTO marketing.dates(SELECT * FROM date);
```

```
CREATE TABLE marketing.events(
eventid integer,
venueid smallint,
catid smallint,
dateid smallint,
eventname varchar(200),
starttime timestamp
);
INSERT INTO marketing.events(SELECT * FROM event);

CREATE TABLE marketing.sale(
marketingid integer,
listid integer,
sellerid integer,
buyerid integer,
eventid integer,
dateid smallint,
qtysold smallint,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp
);
INSERT INTO marketing.sale(SELECT * FROM marketing);
```

Etapa 3: Criar um usuário somente leitura

Nesta etapa, você cria uma função somente leitura e um salesanalyst para a função somente leitura. O analista de vendas só precisa de acesso somente leitura às tabelas no esquema de vendas para realizar a tarefa atribuída de encontrar os eventos que geraram as comissões mais altas.

1. Conecte-se ao banco de dados como o usuário dbadmin.
2. Para criar a função sales_ro, use o exemplo a seguir.

```
CREATE ROLE sales_ro;
```

3. Para criar a função salesanalyst, use o exemplo a seguir.

```
CREATE USER salesanalyst PASSWORD 'Test12345';
```

4. Para conceder acesso de uso e seleção à função sales_ro aos objetos do esquema de vendas, use o exemplo a seguir.

```
GRANT USAGE ON SCHEMA sales TO ROLE sales_ro;
GRANT SELECT ON ALL TABLES IN SCHEMA sales TO ROLE sales_ro;
```

5. Para conceder ao usuário salesanalyst a função sales_ro, use o exemplo a seguir.

```
GRANT ROLE sales_ro TO salesanalyst;
```

Etapa 4: Consultar os dados como usuário somente leitura

Nesta etapa, o usuário salesanalyst consulta os dados do esquema de vendas. Em seguida, o usuário salesanalyst tenta atualizar uma tabela e ler tabelas no esquema de marketing.

1. Conecte-se ao banco de dados como o usuário salesanalyst.
2. Para encontrar as dez vendas com as comissões mais altas, use o exemplo a seguir.

```
SET SEARCH_PATH TO sales;
SELECT DISTINCT events.dateid, sale.commission, cat.catname
FROM sale, events, dates, cat
WHERE events.dateid=dates.dateid AND events.dateid=sale.dateid AND events.catid =
 cat.catid
ORDER BY 2 DESC LIMIT 10;

+--------+------------+----------+
| dateid | commission | catname  |
+--------+------------+----------+
|   1880 |     1893.6 | Pop      |
|   1880 |     1893.6 | Opera    |
|   1880 |     1893.6 | Plays    |
|   1880 |     1893.6 | Musicals |
|   1861 |       1500 | Plays    |
|   2003 |       1500 | Pop      |
|   1861 |       1500 | Opera    |
|   2003 |       1500 | Plays    |
|   1861 |       1500 | Musicals |
|   1861 |       1500 | Pop      |
+--------+------------+----------+
```

3. Para selecionar dez eventos na tabela de eventos no esquema de vendas, use o exemplo a seguir.

```
SELECT * FROM sales.events LIMIT 10;

+---------+---------+-------+--------+--------------------+---------------------+
| eventid | venueid | catid | dateid |     eventname      |      starttime      |
+---------+---------+-------+--------+--------------------+---------------------+
|    4836 |      73 |     9 |   1871 | Soulfest           | 2008-02-14 19:30:00 |
|    5739 |      41 |     9 |   1871 | Fab Faux           | 2008-02-14 19:30:00 |
|     627 |     229 |     6 |   1872 | High Society       | 2008-02-15 14:00:00 |
|    2563 |     246 |     7 |   1872 | Hamlet             | 2008-02-15 20:00:00 |
|    7703 |      78 |     9 |   1872 | Feist              | 2008-02-15 14:00:00 |
|    7903 |      90 |     9 |   1872 | Little Big Town    | 2008-02-15 19:30:00 |
|    7925 |     101 |     9 |   1872 | Spoon              | 2008-02-15 19:00:00 |
|    8113 |      17 |     9 |   1872 | Santana            | 2008-02-15 15:00:00 |
|     463 |     303 |     8 |   1873 | Tristan und Isolde | 2008-02-16 19:00:00 |
|     613 |     236 |     6 |   1873 | Pal Joey           | 2008-02-16 15:00:00 |
+---------+---------+-------+--------+--------------------+---------------------+
```

4. Para tentar atualizar eventname para eventid 1, execute o exemplo a seguir. Este exemplo gerará um erro de permissão negada porque o usuário salesanalyst só tem permissões SELECT na tabela de eventos no esquema de vendas. Para atualizar a tabela de eventos, você deve conceder permissões UPDATE à função sales_ro. Para obter mais informações sobre como conceder permissões para atualizar uma tabela, consulte o parâmetro UPDATE para GRANT. Para obter mais informações sobre o comando UPDATE, consulte UPDATE.

```
UPDATE sales.events
SET eventname = 'Comment event'
WHERE eventid = 1;

ERROR: permission denied for relation events
```

5. Para tentar selecionar todos os eventos na tabela no esquema de marketing, use o exemplo a seguir. Este exemplo gerará um erro de permissão negada porque o usuário salesanalyst só tem permissões SELECT para a tabela de eventos no esquema de vendas. Para selecionar dados da tabela de eventos no esquema de marketing, você deve conceder à função sales_ro as permissões SELECT na tabela de eventos do esquema de marketing.

```
SELECT * FROM marketing.events;
```

```
ERROR: permission denied for schema marketing
```

Etapa 5: Criar um usuário de leitura e gravação

Nesta etapa, o engenheiro de vendas responsável por criar o pipeline de extração, transformação e carregamento (ETL) para processamento de dados no esquema de vendas receberá acesso somente leitura, mas posteriormente receberá acesso de leitura e gravação para realizar suas tarefas.

1. Conecte-se ao banco de dados como o usuário dbadmin.
2. Para criar a função sales_rw no esquema de vendas, use o exemplo a seguir.

```
CREATE ROLE sales_rw;
```

3. Para criar o usuário salesengineer, use o exemplo a seguir.

```
CREATE USER salesengineer PASSWORD 'Test12345';
```

4. Para conceder acesso de uso e seleção à função sales_rw aos objetos do esquema de vendas atribuindo a função sales_ro a ela, use o exemplo a seguir. Para obter mais informações sobre como as funções herdam permissões no Amazon Redshift, consulte Hierarquia de funções.

```
GRANT ROLE sales_ro TO ROLE sales_rw;
```

5. Para atribuir a função sales_rw ao usuário salesengineer, use o exemplo a seguir.

```
GRANT ROLE sales_rw TO salesengineer;
```

Etapa t: Consultar os dados como o usuário com a função somente leitura herdada

Nesta etapa, o usuário salesengineer tenta atualizar a tabela de eventos antes de receber permissões de leitura.

1. Conecte-se ao banco de dados como o usuário salesengineer.
2. O usuário salesengineer pode ler com êxito os dados da tabela de eventos do esquema de vendas. Para selecionar o evento com eventid 1 na tabela de eventos no esquema de vendas, use o exemplo a seguir.

```
SELECT * FROM sales.events where eventid=1;

+---------+---------+-------+--------+----------------+---------------------+
| eventid | venueid | catid | dateid |    eventname    |      starttime      |
+---------+---------+-------+--------+----------------+---------------------+
|       1 |     305 |     8 |   1851 | Gotterdammerung | 2008-01-25 14:30:00 |
+---------+---------+-------+--------+----------------+---------------------+
```

3. Para tentar selecionar todos os eventos na tabela no esquema de marketing, use o exemplo a seguir. Como o usuário salesengineer não tem permissões para tabelas no esquema de marketing, essa consulta gerará um erro de permissão negada. Para selecionar dados da tabela de eventos no esquema de marketing, você deve conceder à função sales_rw as permissões SELECT na tabela de eventos do esquema de marketing.

```
SELECT * FROM marketing.events;

ERROR: permission denied for schema marketing
```

4. Para tentar atualizar eventname para eventid 1, execute o exemplo a seguir. Este exemplo gerará um erro de permissão negada porque o usuário salesengineer só tem permissões de seleção na tabela de eventos no esquema de vendas. Para atualizar a tabela de eventos, você deve conceder permissões de UPDATE à função sales_rw.

```
UPDATE sales.events
SET eventname = 'Comment event'
WHERE eventid = 1;

ERROR: permission denied for relation events
```

Etapa 7: Conceder permissões de atualização e inserção à função de leitura e gravação

Nesta etapa, você concede permissões de atualização e inserção à função sales_rw.

1. Conecte-se ao banco de dados como o usuário dbadmin.
2. Para conceder as permissões UPDATE, INSERT e DELETE à função sales_rw, use o exemplo a seguir.

```
GRANT UPDATE, INSERT, ON ALL TABLES IN SCHEMA sales TO role sales_rw;
```

Etapa 8: Consultar os dados como o usuário de leitura e gravação

Nesta etapa, o usuário salesengineer atualiza com êxito a tabela depois que sua função recebe as permissões de inserção e atualização. Em seguida, ele tenta analisar e limpar a tabela de eventos, mas não consegue fazer isso.

1. Conecte-se ao banco de dados como o usuário salesengineer.
2. Para atualizar eventname para eventid 1, execute o exemplo a seguir.

```
UPDATE sales.events
SET eventname = 'Comment event'
WHERE eventid = 1;
```

3. Para visualizar a alteração feita na consulta anterior, use o exemplo a seguir para selecionar o evento com eventid 1 na tabela de eventos no esquema de vendas.

```
SELECT * FROM sales.events WHERE eventid=1;

+---------+---------+-------+--------+--------------+---------------------+
| eventid | venueid | catid | dateid |   eventname  |      starttime      |
+---------+---------+-------+--------+--------------+---------------------+
|       1 |     305 |     8 |   1851 | Comment event | 2008-01-25 14:30:00 |
+---------+---------+-------+--------+--------------+---------------------+
```

4. Para analisar a tabela de eventos atualizada no esquema de vendas, use o exemplo a seguir. Este exemplo gerará um erro de permissão negada porque o usuário salesengineer não tem as permissões necessárias e não é o proprietário da tabela de eventos no esquema de vendas. Para analisar a tabela de eventos, você deve conceder permissões ANALYZE à função sales_rw usando o comando GRANT. Para obter mais informações sobre o comando ANALYZE, consulte ANALYZE.

```
ANALYZE sales.events;

               ERROR: skipping "events" --- only table or database owner can analyze
```

5. Para limpar a tabela de eventos atualizada, use o exemplo a seguir. Este exemplo gerará um erro de permissão negada porque o usuário salesengineer não tem as permissões necessárias e não é o proprietário da tabela de eventos no esquema de vendas. Para limpar a tabela de eventos, você deve conceder permissões VACUUM à função sales_rw usando o comando GRANT. Para obter mais informações sobre o comando VACUUM, consulte VACUUM.

```
VACUUM sales.events;

ERROR: skipping "events" --- only table or database owner can vacuum it
```

Etapa 9: Analisar e limpar tabelas em um banco de dados como o usuário administrador

Nesta etapa, o usuário dbadmin analisa e limpa todas as tabelas. O usuário tem permissões de administrador nesse banco de dados, então ele pode executar esses comandos.

1. Conecte-se ao banco de dados como o usuário dbadmin.
2. Para analisar a tabela de eventos no esquema de vendas, use o exemplo a seguir.

```
ANALYZE sales.events;
```

3. Para limpar a tabela de eventos no esquema de vendas, use o exemplo a seguir.

```
VACUUM sales.events;
```

4. Para analisar a tabela de eventos no esquema de marketing, use o exemplo a seguir.

```
ANALYZE marketing.events;
```

5. Para limpar a tabela de eventos no esquema de marketing, use o exemplo a seguir.

```
VACUUM marketing.events;
```

Etapa 10: Truncar tabelas como o usuário de leitura e gravação

Nesta etapa, o usuário salesengineer tenta truncar a tabela de eventos no esquema de vendas, mas só consegue quando o usuário dbadmin concede permissões para truncar.

1. Conecte-se ao banco de dados como o usuário salesengineer.
2. Para tentar excluir todas as linhas na tabela de eventos no esquema de vendas, use o exemplo a seguir. Este exemplo gerará um erro porque o usuário salesengineer não tem as permissões necessárias e não é o proprietário da tabela de eventos no esquema de vendas. Para truncar a tabela de eventos, você deve conceder permissões TRUNCATE à função sales_rw usando o comando GRANT. Para obter mais informações sobre o comando TRUNCATE, consulte TRUNCATE.

```
TRUNCATE sales.events;

ERROR: must be owner of relation events
```

3. Conecte-se ao banco de dados como o usuário dbadmin.
4. Para conceder à função sales_rw privilégios para truncar a tabela, use o exemplo a seguir.

```
GRANT TRUNCATE TABLE TO role sales_rw;
```

5. Conecte-se ao banco de dados como o usuário salesengineer utilizando o editor de consultas v2.
6. Para ler os dez primeiros eventos na tabela de eventos no esquema de vendas, use o exemplo a seguir.

```
SELECT * FROM sales.events ORDER BY eventid LIMIT 10;

+---------+---------+-------+--------+----------------------------
+---------------------+
| eventid | venueid | catid | dateid |          eventname          |      starttime
    |
+---------+---------+-------+--------+----------------------------
+---------------------+
|       1 |     305 |     8 |   1851 | Comment event               | 2008-01-25
 14:30:00 |
|       2 |     306 |     8 |   2114 | Boris Godunov               | 2008-10-15
 20:00:00 |
|       3 |     302 |     8 |   1935 | Salome                      | 2008-04-19
 14:30:00 |
|       4 |     309 |     8 |   2090 | La Cenerentola (Cinderella) | 2008-09-21
 14:30:00 |
|       5 |     302 |     8 |   1982 | Il Trovatore                | 2008-06-05
 19:00:00 |
|       6 |     308 |     8 |   2109 | L Elisir d Amore            | 2008-10-10
 19:30:00 |
```

```
|       7 |     309 |     8 |   1891 | Doctor Atomic               | 2008-03-06
 14:00:00 |
|       8 |     302 |     8 |   1832 | The Magic Flute             | 2008-01-06
 20:00:00 |
|       9 |     308 |     8 |   2087 | The Fly                     | 2008-09-18
 19:30:00 |
|      10 |     305 |     8 |   2079 | Rigoletto                   | 2008-09-10
 15:00:00 |
+---------+---------+-------+--------+-----------------------------
+---------------------+
```

7. Para truncar a tabela de eventos no esquema de vendas, use o exemplo a seguir.

```
TRUNCATE sales.events;
```

8. Para ler os dados da tabela de eventos atualizada no esquema de vendas, use o exemplo a seguir.

```
SELECT * FROM sales.events ORDER BY eventid LIMIT 10;

+---------+---------+-------+--------+-----------------------------
+---------------------+
| eventid | venueid | catid | dateid |          eventname          |      starttime
    |
+---------+---------+-------+--------+-----------------------------
+---------------------+
```

Criar funções somente leitura e leitura e gravação para o esquema de marketing (opcional)

Nesta etapa, você cria funções somente leitura e leitura e gravação para o esquema de marketing.

1. Conecte-se ao banco de dados como o usuário dbadmin.
2. Para criar funções somente leitura e leitura e gravação para o esquema de marketing, use o exemplo a seguir.

```
CREATE ROLE marketing_ro;

CREATE ROLE marketing_rw;

GRANT USAGE ON SCHEMA marketing TO ROLE marketing_ro, ROLE marketing_rw;
```

```
GRANT SELECT ON ALL TABLES IN SCHEMA marketing TO ROLE marketing_ro;

GRANT ROLE marketing_ro TO ROLE marketing_rw;

GRANT INSERT, UPDATE, DELETE ON ALL TABLES IN SCHEMA marketing TO ROLE marketing_rw;

CREATE USER marketinganalyst PASSWORD 'Test12345';

CREATE USER marketingengineer PASSWORD 'Test12345';

GRANT ROLE marketing_ro TO marketinganalyst;

GRANT ROLE marketing_rw TO marketingengineer;
```

Funções do sistema para RBAC (opcional)

O Amazon Redshift tem duas funções para fornecer informações do sistema sobre associação de usuários e funções em grupos ou funções adicionais: role_is_member_of e user_is_member_of. Essas funções estão disponíveis para superusuários e usuários regulares. Os superusuários podem verificar a associação de todas as funções. Os usuários regulares só podem verificar a associação de funções às quais tenham acesso.

Como usar a função role_is_member_of

1. Conecte-se ao banco de dados como o usuário salesengineer.
2. Para verificar se a função sales_rw é membro da função sales_ro, use o exemplo a seguir.

```
SELECT role_is_member_of('sales_rw', 'sales_ro');

+-------------------+
| role_is_member_of |
+-------------------+
| true              |
+-------------------+
```

3. Para verificar se a função sales_ro é membro da função sales_rw, use o exemplo a seguir.

```
SELECT role_is_member_of('sales_ro', 'sales_rw');

+-------------------+
| role_is_member_of |
```

```
+-------------------+
| false             |
+-------------------+
```

Como usar a função user_is_member_of

1. Conecte-se ao banco de dados como o usuário salesengineer.
2. O exemplo a seguir tenta verificar a associação do usuário salesanalyst. Essa consulta gera um erro porque o usuário salesengineer não tem acesso ao usuário salesanalyst. Para executar esse comando com êxito, conecte-se ao banco de dados como o usuário salesanalyst e use o exemplo.

```
SELECT user_is_member_of('salesanalyst', 'sales_ro');

ERROR
```

3. Conecte-se ao banco de dados como superusuário.
4. Para verificar a associação do usuário salesanalyst quando conectado como superusuário, use o exemplo a seguir.

```
SELECT user_is_member_of('salesanalyst', 'sales_ro');

+-------------------+
| user_is_member_of |
+-------------------+
| true              |
+-------------------+
```

5. Conecte-se ao banco de dados como o usuário dbadmin.
6. Para verificar a associação do usuário salesengineer quando conectado como superusuário, use o exemplo a seguir.

```
SELECT user_is_member_of('salesengineer', 'sales_ro');

+-------------------+
| user_is_member_of |
+-------------------+
| true              |
+-------------------+
```

```
SELECT user_is_member_of('salesengineer', 'marketing_ro');

+-------------------+
| user_is_member_of |
+-------------------+
| false             |
+-------------------+

SELECT user_is_member_of('marketinganalyst', 'sales_ro');

+-------------------+
| user_is_member_of |
+-------------------+
| false             |
+-------------------+
```

Visualizações do sistema para RBAC (opcional)

Para visualizar os perfis, a atribuição de perfis a usuários, a hierarquia de perfis e os privilégios de objetos de banco de dados por meio de perfis, use as visualizações do sistema do Amazon Redshift. Essas visualizações estão disponíveis para superusuários e usuários regulares. Os superusuários podem verificar todos os detalhes da função. Os usuários regulares só podem verificar detalhes das funções às quais tenham acesso.

1. Para visualizar uma lista de usuários aos quais se concedem funções no cluster explicitamente, use o exemplo a seguir.

   ```
   SELECT * FROM svv_user_grants;
   ```

2. Para visualizar uma lista de funções às quais se concedem funções no cluster explicitamente, use o exemplo a seguir.

   ```
   SELECT * FROM svv_role_grants;
   ```

Para obter a lista completa de visualizações do sistema, consulte Visualizações SVV de metadados.

Usar a segurança por linha com o RBAC (opcional)

Para ter controle de acesso detalhado sobre seus dados confidenciais, use a segurança por linha (RLS). Para obter mais informações sobre RLS, consulte Segurança por linha.

Nesta seção, você cria uma política de RLS que concede ao usuário `salesengineer` permissões para visualizar somente as linhas na tabela `cat` que tenham o valor `catdesc` da Major League Baseball. Em seguida, você consulta o banco de dados como o usuário `salesengineer`.

1. Conecte-se ao banco de dados como o usuário `salesengineer`.
2. Para visualizar as primeiras cinco entradas na tabela `cat`, use o exemplo a seguir.

```
SELECT *
FROM sales.cat
ORDER BY catid ASC
LIMIT 5;

+-------+----------+---------+---------------------------------+
| catid | catgroup | catname |             catdesc             |
+-------+----------+---------+---------------------------------+
|     1 | Sports   | MLB     | Major League Baseball           |
|     2 | Sports   | NHL     | National Hockey League          |
|     3 | Sports   | NFL     | National Football League        |
|     4 | Sports   | NBA     | National Basketball Association |
|     5 | Sports   | MLS     | Major League Soccer             |
+-------+----------+---------+---------------------------------+
```

3. Conecte-se ao banco de dados como o usuário `dbadmin`.
4. Para criar uma política de RLS para a coluna `catdesc` na tabela `cat`, use o exemplo a seguir.

```
CREATE RLS POLICY policy_mlb_engineer
WITH (catdesc VARCHAR(50))
USING (catdesc = 'Major League Baseball');
```

5. Para anexar a política de RLS à função `sales_rw`, use o exemplo a seguir.

```
ATTACH RLS POLICY policy_mlb_engineer ON sales.cat TO ROLE sales_rw;
```

6. Para alterar a tabela a fim de ativar a RLS, use o exemplo a seguir.

```
ALTER TABLE sales.cat ROW LEVEL SECURITY ON;
```

7. Conecte-se ao banco de dados como o usuário `salesengineer`.
8. Para tentar visualizar as primeiras cinco entradas na tabela `cat`, use o exemplo a seguir. Observe que as entradas só aparecem quando a coluna `catdesc` é `Major League Baseball`.

```
SELECT *
FROM sales.cat
ORDER BY catid ASC
LIMIT 5;

+-------+----------+---------+----------------------+
| catid | catgroup | catname |        catdesc       |
+-------+----------+---------+----------------------+
|     1 | Sports   | MLB     | Major League Baseball |
+-------+----------+---------+----------------------+
```

9. Conecte-se ao banco de dados como o usuário `salesanalyst`.
10. Para tentar visualizar as primeiras cinco entradas na tabela `cat`, use o exemplo a seguir. Observe que nenhuma entrada aparece porque a política padrão de negar tudo é aplicada.

```
SELECT *
FROM sales.cat
ORDER BY catid ASC
LIMIT 5;

+-------+----------+---------+----------------------+
| catid | catgroup | catname |        catdesc       |
+-------+----------+---------+----------------------+
```

11. Conecte-se ao banco de dados como o usuário `dbadmin`.
12. Para conceder a permissão IGNORE RLS à função `sales_ro`, use o exemplo a seguir. Isso concede ao usuário `salesanalyst` as permissões para ignorar as políticas de RLS, pois ele é membro da função `sales_ro`.

```
GRANT IGNORE RLS TO ROLE sales_ro;
```

13. Conecte-se ao banco de dados como o usuário `salesanalyst`.
14. Para visualizar as primeiras cinco entradas na tabela `cat`, use o exemplo a seguir.

```
SELECT *
FROM sales.cat
```

```
ORDER BY catid ASC
LIMIT 5;

+-------+----------+---------+--------------------------------+
| catid | catgroup | catname |             catdesc            |
+-------+----------+---------+--------------------------------+
|     1 | Sports   | MLB     | Major League Baseball          |
|     2 | Sports   | NHL     | National Hockey League         |
|     3 | Sports   | NFL     | National Football League       |
|     4 | Sports   | NBA     | National Basketball Association |
|     5 | Sports   | MLS     | Major League Soccer            |
+-------+----------+---------+--------------------------------+
```

15. Conecte-se ao banco de dados como o usuário `dbadmin`.
16. Para revogar a permissão IGNORE RLS da função `sales_ro`, use o exemplo a seguir.

```
REVOKE IGNORE RLS FROM ROLE sales_ro;
```

17. Conecte-se ao banco de dados como o usuário `salesanalyst`.
18. Para tentar visualizar as primeiras cinco entradas na tabela `cat`, use o exemplo a seguir. Observe que nenhuma entrada aparece porque a política padrão de negar tudo é aplicada.

```
SELECT *
FROM sales.cat
ORDER BY catid ASC
LIMIT 5;

+-------+----------+---------+----------------------+
| catid | catgroup | catname |        catdesc       |
+-------+----------+---------+----------------------+
```

19. Conecte-se ao banco de dados como o usuário `dbadmin`.
20. Para desanexar a política de RLS da tabela `cat`, use o exemplo a seguir.

```
DETACH RLS POLICY policy_mlb_engineer ON cat FROM ROLE sales_rw;
```

21. Conecte-se ao banco de dados como o usuário `salesanalyst`.
22. Para tentar visualizar as primeiras cinco entradas na tabela `cat`, use o exemplo a seguir. Observe que nenhuma entrada aparece porque a política padrão de negar tudo é aplicada.

```
SELECT *
```

```
FROM sales.cat
ORDER BY catid ASC
LIMIT 5;

+-------+----------+---------+---------------------------------+
| catid | catgroup | catname |             catdesc             |
+-------+----------+---------+---------------------------------+
|     1 | Sports   | MLB     | Major League Baseball           |
|     2 | Sports   | NHL     | National Hockey League          |
|     3 | Sports   | NFL     | National Football League        |
|     4 | Sports   | NBA     | National Basketball Association |
|     5 | Sports   | MLS     | Major League Soccer             |
+-------+----------+---------+---------------------------------+
```

23. Conecte-se ao banco de dados como o usuário `dbadmin`.

24. Para descartar a política de RLS, use o exemplo a seguir.

```
DROP RLS POLICY policy_mlb_engineer;
```

25. Para remover a RLS, use o exemplo a seguir.

```
ALTER TABLE cat ROW LEVEL SECURITY OFF;
```

Tópicos relacionados

Para obter mais informações sobre RBAC, consulte a seguinte documentação:

- Hierarquia de funções
- Atribuição de função
- Permissões de objetos do banco de dados
- ALTER DEFAULT PRIVILEGES para o RBAC

Segurança por linha

Usando a segurança no nível da linha (RLS) no Amazon Redshift, é possível ter controle de acesso granular sobre seus dados confidenciais. É possível decidir quais usuários ou funções podem acessar registros específicos de dados em esquemas ou tabelas com base nas políticas de segurança definidas no nível dos objetos do banco de dados. Além da segurança no nível da

coluna, na qual é possível conceder permissões aos usuários para um subconjunto de colunas, use as políticas de RLS para restringir ainda mais o acesso a linhas específicas das colunas visíveis. Para obter mais informações sobre regras de segurança no nível de coluna, consulte Observações de uso para controle de acesso no nível da coluna.

Quando você impuser políticas de RLS em tabelas, poderá restringir os conjuntos de resultados retornados quando os usuários executarem consultas.

Ao criar políticas de RLS, é possível especificar expressões que determinam se o Amazon Redshift retornará quaisquer linhas existentes em uma tabela de uma consulta. Ao criar políticas de RLS para limitar o acesso, você não precisa adicionar ou externalizar condições adicionais em suas consultas.

Ao criar políticas de RLS, recomendamos que você crie políticas simples e evite declarações complexas nas políticas. Ao definir políticas de RLS, não use junções de tabela excessivas na definição de política que sejam baseadas em políticas.

Quando uma política se refere a uma tabela de pesquisa, o Amazon Redshift verifica a tabela adicional, além da tabela na qual a política existe. Haverá diferenças de performance entre a mesma consulta para um usuário com uma política de RLS anexada e um usuário sem nenhuma política anexada.

Utilização de políticas de RLS em instruções SQL

Ao usar políticas de RLS em instruções SQL, o Amazon Redshift aplica as seguintes regras:

- O Amazon Redshift aplica políticas de RLS às instruções SELECT, UPDATE e DELETE por padrão.
- Para SELECT e UNLOAD, o Amazon Redshift filtra linhas de acordo com sua política definida.
- Para UPDATE, o Amazon Redshift atualiza somente as linhas que estão visíveis para você. Se uma política restringir um subconjunto das linhas em uma tabela, não será possível atualizá-las.
- Para DELETE, é possível excluir somente as linhas que estão visíveis para você. Se uma política restringir um subconjunto das linhas em uma tabela, não será possível excluí-las. Para TRUNCATE, ainda será possível truncar a tabela.
- Para CREATE TABLE LIKE, as tabelas criadas com as opções LIKE não herdarão as configurações de permissão da tabela de origem. Da mesma forma, a tabela de destino não herdará as políticas de RLS da tabela de origem.

Combinar várias políticas por usuário

O RLS no Amazon Redshift é compatível com a anexação de várias políticas por usuário e objeto. Quando há várias políticas definidas para um usuário, o Amazon Redshift aplica todas as políticas com a sintaxe AND ou OR dependendo da definição RLS CONJUNCTION TYPE da tabela. Para obter mais informações sobre o tipo de conjunção, consulte ALTER TABLE.

Várias políticas em uma tabela podem ser associadas a você. Várias políticas estão diretamente anexadas a você ou você pertence a várias funções, e as funções têm políticas diferentes anexadas a elas.

Quando as várias políticas precisam restringir o acesso a linhas em uma determinada relação, você pode definir RLS CONJUNCTION TYPE da relação como AND. Considere o exemplo a seguir. Alice só consegue ver um evento Sports que tenham um "catname" NBA de acordo com a política especificada.

```
-- Create an analyst role and grant it to a user named Alice.
CREATE ROLE analyst;
CREATE USER alice WITH PASSWORD 'Name_is_alice_1';
GRANT ROLE analyst TO alice;

-- Create an RLS policy that only lets the user see sports.
CREATE RLS POLICY policy_sports
WITH (catgroup VARCHAR(10))
USING (catgroup = 'Sports');

-- Create an RLS policy that only lets the user see NBA.
CREATE RLS POLICY policy_nba
WITH (catname VARCHAR(10))
USING (catname = 'NBA');

-- Attach both to the analyst role.
ATTACH RLS POLICY policy_sports ON category TO ROLE analyst;
ATTACH RLS POLICY policy_nba ON category TO ROLE analyst;

-- Activate RLS on the category table with AND CONJUNCTION TYPE.
ALTER TABLE category ROW LEVEL SECURITY ON CONJUNCTION TYPE AND;

-- Change session to Alice.
SET SESSION AUTHORIZATION alice;

-- Select all from the category table.
```

```
SELECT catgroup, catname
FROM category;

 catgroup | catname
---------+---------
 Sports   | NBA
(1 row)
```

Quando as várias políticas precisam permitir a usuários ver mais linhas em uma determinada relação, o usuário pode definir RLS CONJUNCTION TYPE da relação como OR. Considere o exemplo a seguir. Alice só consegue ver "Shows" e "Esportes" de acordo com a política especificada.

```
-- Create an analyst role and grant it to a user named Alice.
CREATE ROLE analyst;
CREATE USER alice WITH PASSWORD 'Name_is_alice_1';
GRANT ROLE analyst TO alice;

-- Create an RLS policy that only lets the user see concerts.
CREATE RLS POLICY policy_concerts
WITH (catgroup VARCHAR(10))
USING (catgroup = 'Concerts');

-- Create an RLS policy that only lets the user see sports.
CREATE RLS POLICY policy_sports
WITH (catgroup VARCHAR(10))
USING (catgroup = 'Sports');

-- Attach both to the analyst role.
ATTACH RLS POLICY policy_concerts ON category TO ROLE analyst;
ATTACH RLS POLICY policy_sports ON category TO ROLE analyst;

-- Activate RLS on the category table with OR CONJUNCTION TYPE.
ALTER TABLE category ROW LEVEL SECURITY ON CONJUNCTION TYPE OR;

-- Change session to Alice.
SET SESSION AUTHORIZATION alice;

-- Select all from the category table.
SELECT catgroup, count(*)
FROM category
GROUP BY catgroup ORDER BY catgroup;

 catgroup | count
```

```
---------+-------
 Concerts |  3
 Sports   |  5
(2 rows)
```

Propriedade e gerenciamento da política de RLS

Como superusuário, administrador de segurança ou usuário que tem a função sys:secadmin, você pode criar, modificar, anexar e desanexar políticas de RLS. As políticas de RLS podem ser anexadas a tabelas, visualizações, visualizações de vinculação tardia (LBVs) e visões materializadas (MVs). No nível do objeto, é possível ativar ou desativar a segurança no nível da linha sem modificar a definição do esquema para tabelas.

Para começar a usar a segurança no nível da linha, a seguir estão as instruções SQL que é possível usar:

- Use a instrução ALTER TABLE para ativar ou desativar a RLS em uma tabela, visualização ou visualização de vinculação tardia. Para obter mais informações, consulte ALTER TABLE.
- Use a instrução ALTER MATERIALIZED VIEW para ativar ou desativar a RLS em uma visão materializada (MV). Para obter mais informações, consulte ALTER MATERIALIZED VIEW.
- Use a instrução CREATE RLS POLICY para criar uma política de segurança para uma ou mais tabelas e especificar um ou mais usuários ou funções na política.

  Para obter mais informações, consulte CREATE RLS POLICY.
- Use a instrução ALTER RLS POLICY para alterar a política, como alterar a definição da política. Você pode usar a mesma política para várias tabelas ou visões.

  Para obter mais informações, consulte ALTER RLS POLICY.
- Use a instrução ATTACH RLS POLICY para anexar uma política a uma ou mais relações, a um ou mais usuários ou a funções.

  Para obter mais informações, consulte ATTACH RLS POLICY.
- Use a declaração DETACH RLS POLICY para desanexar uma política de uma ou mais relações, de um ou mais usuários ou de perfis.

  Para obter mais informações, consulte DETACH RLS POLICY.
- Use a instrução DROP RLS POLICY para descartar uma política.

  Para obter mais informações, consulte DROP RLS POLICY.

- Use as instruções GRANT e REVOKE para conceder e revogar explicitamente permissões SELECT para políticas de RLS que fazem referência a tabelas de pesquisa. Para obter mais informações, consulte GRANT e REVOKE.

Para monitorar as políticas criadas, sys:secadmin pode visualizar SVV_RLS_POLICY e SVV_RLS_ATTACHED_POLICY.

Para listar relações protegidas por RLS, sys:secadmin pode visualizar SVV_RLS_RELATION.

Para monitorar a aplicação de políticas de RLS em consultas que fazem referência a relações protegidas por RLS, um superusuário sys:operator ou qualquer usuário com a permissão do sistema ACCESS SYSTEM TABLE pode visualizar SVV_RLS_APPLIED_POLICY. Observe que sys:secadmin não recebe essas permissões por padrão.

Para permitir que os usuários tenham acesso total a uma relação protegida por RLS, você pode conceder a permissão IGNORE RLS. Superusuários ou sys:secadmin recebem automaticamente IGNORE RLS. Para obter mais informações, consulte GRANT.

Para explicar os filtros de política de RLS de uma consulta no plano EXPLAIN para solucionar problemas de consultas relacionadas à RLS, é possível conceder a permissão EXPLAIN RLS a qualquer usuário. Para obter mais informações, consulte GRANT e EXPLAIN.

Objetos e princípios dependentes de políticas

Para fornecer segurança a aplicações e evitar que objetos da política se tornem obsoletos ou inválidos, o Amazon Redshift não permite descartar ou alterar objetos referenciados por políticas de RLS.

Veja a seguir uma lista das dependências de objeto de esquema que o Amazon Redshift monitora para políticas de RLS.

- Ao monitorar a dependência do objeto de esquema para a tabela de destino, o Amazon Redshift segue estas regras:
  - O Amazon Redshift separa a política de uma relação, usuário, função ou público quando você descarta uma tabela de destino.
  - Quando você renomeia um nome de tabela de destino, não há impacto nas políticas anexadas.
  - Não é possível descartar as colunas da tabela de políticas referenciada dentro da definição de política, a menos que você descarte ou desanexe a política. Isso também se aplica quando a opção CASCADE é especificada. É possível descartar outras colunas na tabela de destino.

  - Não é possível renomear as colunas referenciadas da tabela de destino. Para renomear as colunas referidas, desanexe a política primeiro. Isso também se aplica quando a opção CASCADE é especificada.
  - Não é possível alterar o tipo da coluna referenciada, mesmo quando você especifica a opção CASCADE.
- Ao monitorar a dependência do objeto de esquema para a tabela de consulta, o Amazon Redshift segue estas regras:
  - Não é possível descartar uma tabela de pesquisa. Para descartar uma tabela de pesquisa, primeiro descarte a política na qual a tabela de pesquisa é referenciada.
  - Não é possível renomear uma tabela de pesquisa. Para renomear uma tabela de pesquisa, primeiro descarte a política na qual a tabela de pesquisa é referenciada. Isso também se aplica quando a opção CASCADE é especificada.
  - Não é possível descartar as colunas da tabela de pesquisa usadas na definição de política. Para descartar colunas de uma tabela de pesquisa usadas na definição da política, primeiro descarte a política na qual a tabela de pesquisa é referenciada. Isso também se aplica quando a opção CASCADE é especificada na instrução ALTER TABLE DROP COLUMN. É possível descartar outras colunas na tabela de consultas.
  - Não é possível renomear as colunas referidas da tabela de pesquisa. Para renomear colunas referenciadas, primeiro descarte a política na qual a tabela de pesquisa é referenciada. Isso também se aplica quando a opção CASCADE é especificada.
  - Não é possível alterar o tipo da coluna de referência.
- Ao descartar um usuário ou uma função, o Amazon Redshift desanexa todas as políticas anexadas ao usuário ou à função automaticamente.
- Quando você usa a opção CASCADE na instrução DROP SCHEMA, o Amazon Redshift também descarta as relações no esquema. Ele também descarta as relações em quaisquer outros esquemas que sejam dependentes das relações no esquema descartado. Para uma relação que seja uma tabela de pesquisa em uma política, o Amazon Redshift falhará no DROP SCHEMA DDL. Para todas as relações abandonadas pela instrução DROP SCHEMA, o Amazon Redshift desanexa todas as políticas anexadas a essas relações.
- Só é possível descartar uma função de pesquisa (uma função que é referenciada dentro de uma definição de política) quando você também descarta a política. Isso também se aplica quando a opção CASCADE é especificada.

- Quando uma política é anexada a uma tabela, o Amazon Redshift verifica se essa tabela é uma tabela de pesquisa em uma política diferente. Se esse for o caso, o Amazon Redshift não permitirá anexar uma política a essa tabela.
- Ao criar uma política de RLS, o Amazon Redshift verifica se essa tabela é uma tabela de destino para qualquer outra política de RLS. Se esse for o caso, o Amazon Redshift não permitirá criar uma política nessa tabela.

Exemplo

O exemplo a seguir ilustra como a dependência de esquema é monitorada.

```
-- The CREATE and ATTACH policy statements for `policy_events` references some
-- target and lookup tables.
-- Target tables are tickit_event_redshift and target_schema.target_event_table.
-- Lookup table is tickit_sales_redshift.
-- Policy `policy_events` has following dependencies:
--   table tickit_sales_redshift column eventid, qtysold
--   table tickit_event_redshift column eventid
--   table target_event_table column eventid
--   schema public and target_schema
CREATE RLS POLICY policy_events
WITH (eventid INTEGER)
USING (
    eventid IN (SELECT eventid FROM tickit_sales_redshift WHERE qtysold <3)
);

ATTACH RLS POLICY policy_events ON tickit_event_redshift TO ROLE analyst;

ATTACH RLS POLICY policy_events ON target_schema.target_event_table TO ROLE consumer;
```

Considerações e limitações ao usar políticas de RLS

Considerações

Veja as seguintes considerações para trabalhar com políticas de RLS:

- O Amazon Redshift aplica políticas de RLS às instruções SELECT, UPDATE e DELETE.
- O Amazon Redshift não aplica políticas de RLS a instruções INSERT, COPY e ALTER TABLE APPEND.

- As políticas de RLS podem ser anexadas a tabelas, visualizações, visualizações de vinculação tardia (LBVs) e visões materializadas (MVs).
- A segurança no nível da linha trabalha com a segurança no nível da coluna para proteger seus dados.
- Quando o RLS é ativado para a relação de origem, o Amazon Redshift oferece suporte à instrução ALTER TABLE APPEND para superusuários, usuários que receberam explicitamente a permissão de sistema IGNORE RLS ou o perfil sys:secadmin. Nesse caso, é possível executar a instrução ALTER TABLE APPEND para acrescentar linhas a uma tabela de destino movendo dados de uma tabela de origem existente. O Amazon Redshift move todas as tuplas da relação de origem para a relação de destino. O status RLS da relação de destino não afeta a instrução ALTER TABLE APPEND.
- Para facilitar a migração de outros sistemas de data warehouse, é possível definir e recuperar variáveis de contexto de sessão personalizadas para uma conexão especificando o nome e o valor da variável.

  O exemplo a seguir define variáveis de contexto de sessão para uma política de segurança no nível da linha (RLS).

```
-- Set a customized context variable.
SELECT set_config(‘app.category’, ‘Concerts’, FALSE);

-- Create a RLS policy using current_setting() to get the value of a customized
 context variable.
CREATE RLS POLICY policy_categories
WITH (catgroup VARCHAR(10))
USING (catgroup = current_setting('app.category', FALSE));

-- Set correct roles and attach the policy on the target table to one or more roles.
ATTACH RLS POLICY policy_categories ON tickit_category_redshift TO ROLE analyst, ROLE
 dbadmin;
```

  Para obter detalhes sobre como definir e recuperar variáveis de contexto de sessão personalizadas, acesse SET, SET_CONFIG, SHOW, CURRENT_SETTING e RESET. Para obter mais informações sobre como modificar a configuração do servidor em geral, acesse Modificar a configuração do servidor.

> **Important**
>
> Quando são usadas variáveis de contexto de sessão em políticas de RLS, a política de segurança depende do usuário ou da função que invoca a política. Tenha cuidado para evitar vulnerabilidades de segurança ao usar variáveis de contexto de sessão em políticas de RLS.

- Alterar o usuário da sessão usando SET SESSION AUTHORIZATION entre DECLARE e FETCH, ou entre instruções FETCH subsequentes, não atualizará o plano já preparado com base nas políticas do usuário no momento de DECLARE. Evite alterar o usuário da sessão quando os cursores são usados com tabelas protegidas por RLS.
- Quando os objetos de base dentro de um objeto de visualização são protegidos por RLS, as políticas anexadas ao usuário que executa a consulta são aplicadas nos respectivos objetos de base. Isso é diferente das verificações de permissão por objeto, onde as permissões do proprietário da visualização são comparadas com os objetos de base da visualização. Você pode visualizar as relações protegidas por RLS de uma consulta na saída do plano EXPLAIN.
- Quando uma função definida pelo usuário (UDF) é mencionada em uma política de RLS de uma relação anexada a um usuário, o usuário deve ter a permissão EXECUTE sobre a UDF para consultar a relação.
- A segurança no nível da linha pode limitar a otimização de consultas. É recomendável avaliar cuidadosamente o desempenho da consulta antes de implantar exibições protegidas por RLS em conjuntos de dados grandes.
- As políticas de segurança no nível da linha aplicadas às exibições de vinculação tardia podem ser inseridas em tabelas federadas. Essas políticas RLS podem estar visíveis em logs do mecanismo de processamento externos.

Limitações

Veja a seguir as limitações ao trabalhar com políticas de RLS:

- Não é possível anexar políticas de RLS a tabelas externas e a vários outros tipos de relação. Para obter mais informações, consulte ATTACH RLS POLICY.
- O Amazon Redshift é compatível com a instruções SELECT para determinadas políticas de RLS com pesquisas que têm junções complexas, mas não são compatíveis com instruções UPDATE ou

DELETE. Nos casos com instruções UPDATE ou DELETE, o Amazon Redshift retorna o seguinte erro:

```
ERROR: One of the RLS policies on target relation is not supported in UPDATE/DELETE.
```

- Sempre que uma função definida pelo usuário (UDF) é mencionada em uma política de RLS de uma relação anexada a um usuário, o usuário deve ter a permissão EXECUTE sobre a UDF para consultar a relação.
- Subconsultas correlacionadas não são compatíveis. O Amazon Redshift retorna o erro a seguir:

```
ERROR: RLS policy could not be rewritten.
```

- O Amazon Redshift não oferece suporte ao compartilhamento de dados com RLS. Se uma relação não tiver o RLS desativado para unidades de compartilhamento de dados, a consulta falhará no cluster de consumidores com o seguinte erro:

```
RLS-protected relation "rls_protected_table" cannot be accessed via datasharing
 query.
```

  É possível desativar o RLS para unidades de compartilhamento de dados usando o comando ALTER TABLE com o parâmetro ROW LEVEL SECURITY OFF FOR DATASHARES. Consulte mais informações sobre como usar ALTER TABLE para habilitar ou desabilitar o RLS em ALTER TABLE.
- Em consultas entre bancos de dados, o Amazon Redshift bloqueia as leituras de relações protegidas por RLS. Usuários com a permissão IGNORE RLS podem acessar a relação protegida usando consultas entre bancos de dados. Quando um usuário sem a permissão IGNORE RLS acessa uma relação protegida por RLS por meio de uma consulta entre bancos de dados, o seguinte erro é exibido:

```
RLS-protected relation "rls_protected_table" cannot be accessed via cross-database
 query.
```

- ALTER RLS POLICY só comporta a modificação de uma política de RLS por meio da cláusula USING (using_predicate_exp). Não é possível modificar uma política de RLS com uma cláusula WITH ao executar ALTER RLS POLICY.
- Você não poderá consultar relações que tenham a segurança por linha ativada se os valores de qualquer uma das opções de configuração abaixo não corresponderem ao valor padrão da sessão:
  - `enable_case_sensitive_super_attribute`

  - `enable_case_sensitive_identifier`
  - `downcase_delimited_identifier`

  Considere redefinir as opções de configuração da sessão se você tentar consultar uma relação com a segurança por linha ativada e vir a mensagem “RLS protected relation does not support session level config on case sensitivity being different from its default value”.

- Quando o cluster provisionado ou namespace sem servidor tem alguma política de segurança por linha, os seguintes comandos são bloqueados para usuários comuns:

  ```
  ALTER <current_user> SET enable_case_sensitive_super_attribute/
  enable_case_sensitive_identifier/downcase_delimited_identifier
  ```

  Ao criar políticas de RLS, recomendamos que você altere as definições de opções da configuração padrão para usuários comuns a fim de que correspondam às definições de opções da configuração da sessão no momento em que a política foi criada. Superusuários e usuários com o privilégio ALTER USER podem fazer isso usando as configurações do grupo de parâmetros ou o comando ALTER USER. Para obter informações sobre grupos de parâmetros, consulte Amazon Redshift parameter groups no Guia de gerenciamento do Amazon Redshift. Para obter mais informações sobre o comando ALTER USER, consulte ALTER USER.

- As exibições e as exibições de vinculação tardia com políticas de segurança no nível da linha não podem ser substituídas por usuários regulares usando o comando CREATE VIEW. Para substituir exibições ou LBVs por políticas RLS, primeiro desanexe todas as políticas RLS anexadas, substitua as exibições ou LBVs e reanexe as políticas. Os superusuários e usuários com o `sys:secadmin permission` podem usar CREATE VIEW em exibições ou LBVs com políticas RLS sem desanexar as políticas.
- As exibições com políticas de segurança no nível da linha não podem referenciar tabelas e exibições de sistema.
- Uma exibição de vinculação tardia referenciada por uma exibição normal não pode ser protegida por RLS.
- As relações protegidas por RLS e os dados aninhados de data lakes não podem ser acessados na mesma consulta.

Práticas recomendadas para desempenho de RLS

A seguir estão as práticas recomendadas para obter melhor desempenho do Amazon Redshift em tabelas protegidas pela RLS.

Segurança dos operadores e funções

Ao consultar tabelas protegidas por RLS, o uso de determinados operadores ou funções pode levar à degradação do desempenho. O Amazon Redshift classifica operadores e funções como seguros ou não seguros para consultar tabelas protegidas por RLS. Uma função ou operador é classificado como seguro para RLS quando não tem efeitos colaterais observáveis, dependendo das entradas. Em particular, uma função ou operador seguro para RLS não pode ser um dos seguintes:

- Produz um valor de entrada, ou qualquer valor que seja dependente do valor de entrada, com ou sem uma mensagem de erro.
- Falha ou retorna erros que são dependentes do valor de entrada.

Os operadores não seguros para RLS incluem:

- Operadores aritméticos: +, -, /, *, %.
- Operadores de texto: LIKE e SIMILAR TO.
- Alguns operadores de conversão. Observe que certas conversões são classificadas como seguras, incluindo conversões de data para carimbo de data/hora e de carimbo de data/hora para data, conversões de alargamento de inteiros (como de INT2 para INT8 ou de INT4 para INT8) e conversões de inteiro para texto.
- UDFs.

Use a seguinte instrução SELECT para verificar a segurança dos operadores e das funções.

```
SELECT proname, proc_is_rls_safe(oid) FROM pg_proc;
```

O Amazon Redshift impõe restrições na ordem de avaliação dos predicados do usuário que contêm operadores e funções não seguros para RLS ao planejar consultas em tabelas protegidas por RLS. Consultas que fazem referência a operadores ou funções não seguras para RLS podem causar degradação do desempenho ao consultar tabelas protegidas por RLS. O desempenho pode degradar significativamente quando o Amazon Redshift não pode enviar predicados não seguros

para RLS para varreduras de tabela base para aproveitar as chaves de classificação. Para obter um melhor desempenho, evite consultas usando predicados não seguros para RLS que tiram proveito de uma chave de classificação. Para verificar se o Amazon Redshift é capaz de enviar operadores e funções, é possível usar instruções EXPLAIN em combinação com a permissão de sistema EXPLAIN RLS.

Funções condicionalmente seguras

Algumas funções são classificadas como inseguras em geral, mas tornam-se seguras quando argumentos específicos são valores constantes (literais de valor, não referências de coluna). Quando os argumentos relevantes são constantes, o Amazon Redshift pode enviar esses predicados para as varreduras da tabela base, melhorando o desempenho da consulta.

Por exemplo, `DATE_TRUNC('day', timestamp_col)` é condicionalmente segura porque `'day'` é um literal constante. No entanto, `DATE_TRUNC(datepart_col, timestamp_col)` não é condicionalmente segura porque `datepart_col` é uma referência de coluna.

A tabela a seguir lista as categorias de funções condicionalmente seguras e quais argumentos devem ser constantes para que a função seja considerada segura.

Categoria da função	Funções	Argumento que deve ser constante
DATE_TRUNC	DATE_TRUNC(datepart, timestamp), DATE_TRUNC(datepart, timestamp tz)	datepart (primeiro argumento)
EXTRACT / DATE_PART	EXTRACT(field FROM source), DATE_PART(field, source) para tipos timestamp, timestamptz, date, time, timetz, interval e datetime	field (primeiro argumento)
DATEDIFF	DATEDIFF(datepart, start, end) para tipos timestamp e time	datepart (primeiro argumento)
TO_CHAR	TO_CHAR(timestamp, format), TO_CHAR(timestamptz, format)	format (segundo argumento)

Categoria da função	Funções	Argumento que deve ser constante
CONVERT_TIMEZONE	CONVERT_TIMEZONE(source_tz, target_tz, timestamp), CONVERT_TIMEZONE(target_tz, timestamp)	argumentos de fuso horário
LEFT / RIGHT	LEFT(string, length), RIGHT(string, length)	length (segundo argumento)
SUBSTRING	SUBSTRING(string, start, length) para tipos text, bytea e varbyte	length (terceiro argumento)
SPLIT_PART	SPLIT_PART(string, delimiter, part)	part (terceiro argumento)

Use a instrução SELECT a seguir para verificar quais funções são condicionalmente seguras e quais posições de argumentos devem ser constantes.

```
SELECT proname, proc_is_rls_conditionally_safe(oid) FROM pg_proc
WHERE proc_is_rls_conditionally_safe(oid) IS NOT NULL;
```

A função retorna um array de posições de argumento indexadas a partir de 0 que devem ser constantes, ou NULL se a função não for condicionalmente segura.

Armazenamento em cache dos resultados

Para reduzir o tempo de execução da consulta e melhorar a performance do sistema, o Amazon Redshift armazena em cache os resultados de certos tipos de consultas na memória no nó líder.

O Amazon Redshift usa resultados armazenados em cache para uma nova consulta que verifica tabelas protegidas por RLS quando todas as condições para tabelas desprotegidas são verdadeiras e quando todas as seguintes opções são verdadeiras:

- As tabelas ou as exibições na política não foram modificadas.
- A política não usa uma função que precisa ser avaliada cada vez que ela é executada, por exemplo GETDATE ou CURRENT_USER.

Para melhorar a performance, evite usar predicados de política que não satisfaçam as condições anteriores.

Para obter mais informações sobre o armazenamento em cache de resultados no Amazon Redshift, consulte Armazenamento em cache dos resultados.

Políticas complexas

Para obter um melhor desempenho, evite usar políticas complexas com subconsultas que unem várias tabelas.

Uso de argumentos constantes para melhor pushdown

Ao usar funções como DATE_TRUNC, EXTRACT, DATEDIFF, TO_CHAR, CONVERT_TIMEZONE, LEFT, RIGHT, SUBSTRING ou SPLIT_PART em consultas sobre tabelas protegidas por RLS, utilize argumentos literais constantes em vez de referências a colunas para os argumentos que controlam o comportamento de erro. Isso permite que o Amazon Redshift classifique essas funções como seguras e aplique os predicados diretamente nas leituras da tabela base, o que pode melhorar significativamente o desempenho.

Por exemplo, a consulta a seguir permite o pushdown de predicado porque o argumento datepart é um valor constante:

```
SELECT * FROM rls_protected_table
WHERE DATE_TRUNC('month', event_date) = '2024-01-01';
```

A consulta a seguir impede o pushdown de predicado porque o argumento datepart é uma referência a coluna:

```
SELECT * FROM rls_protected_table
WHERE DATE_TRUNC(datepart_col, event_date) = '2024-01-01';
```

Consulte a lista completa de funções condicionalmente seguras e quais argumentos devem ser constantes em Funções condicionalmente seguras.

Exemplo completo de segurança em nível de linha

Este é um exemplo de ponta a ponta para ilustrar como um superusuário cria alguns usuários e funções. Em seguida, um usuário com a função secadmin cria, anexa, desanexa e descarta políticas

de RLS. Este exemplo usa o exemplo do banco de dados tickit. Para obter mais informações, consulte Carregar dados do Amazon S3 para o Amazon Redshift no Guia de conceitos básicos do Amazon Redshift.

```
-- Create users and roles referenced in the policy statements.
CREATE ROLE analyst;
CREATE ROLE consumer;
CREATE ROLE dbadmin;
CREATE ROLE auditor;
CREATE USER bob WITH PASSWORD 'Name_is_bob_1';
CREATE USER alice WITH PASSWORD 'Name_is_alice_1';
CREATE USER joe WITH PASSWORD 'Name_is_joe_1';
CREATE USER molly WITH PASSWORD 'Name_is_molly_1';
CREATE USER bruce WITH PASSWORD 'Name_is_bruce_1';
GRANT ROLE sys:secadmin TO bob;
GRANT ROLE analyst TO alice;
GRANT ROLE consumer TO joe;
GRANT ROLE dbadmin TO molly;
GRANT ROLE auditor TO bruce;
GRANT ALL ON TABLE tickit_category_redshift TO PUBLIC;
GRANT ALL ON TABLE tickit_sales_redshift TO PUBLIC;
GRANT ALL ON TABLE tickit_event_redshift TO PUBLIC;

-- Create table and schema referenced in the policy statements.
CREATE SCHEMA target_schema;
GRANT ALL ON SCHEMA target_schema TO PUBLIC;
CREATE TABLE target_schema.target_event_table (LIKE tickit_event_redshift);
GRANT ALL ON TABLE target_schema.target_event_table TO PUBLIC;

-- Change session to analyst alice.
SET SESSION AUTHORIZATION alice;

-- Check the tuples visible to analyst alice.
-- Should contain all 3 categories.
SELECT catgroup, count(*)
FROM tickit_category_redshift
GROUP BY catgroup ORDER BY catgroup;

-- Change session to security administrator bob.
SET SESSION AUTHORIZATION bob;

CREATE RLS POLICY policy_concerts
WITH (catgroup VARCHAR(10))
```

```
USING (catgroup = 'Concerts');

SELECT poldb, polname, polalias, polatts, polqual, polenabled, polmodifiedby FROM
 svv_rls_policy WHERE poldb = CURRENT_DATABASE();

ATTACH RLS POLICY policy_concerts ON tickit_category_redshift TO ROLE analyst, ROLE
 dbadmin;

ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY ON;

SELECT * FROM svv_rls_attached_policy;

-- Change session to analyst alice.
SET SESSION AUTHORIZATION alice;

-- Check that tuples with only `Concert` category will be visible to analyst alice.
SELECT catgroup, count(*)
FROM tickit_category_redshift
GROUP BY catgroup ORDER BY catgroup;

-- Change session to consumer joe.
SET SESSION AUTHORIZATION joe;

-- Although the policy is attached to a different role, no tuples will be
-- visible to consumer joe because the default deny all policy is applied.
SELECT catgroup, count(*)
FROM tickit_category_redshift
GROUP BY catgroup ORDER BY catgroup;

-- Change session to dbadmin molly.
SET SESSION AUTHORIZATION molly;

-- Check that tuples with only `Concert` category will be visible to dbadmin molly.
SELECT catgroup, count(*)
FROM tickit_category_redshift
GROUP BY catgroup ORDER BY catgroup;

-- Check that EXPLAIN output contains RLS SecureScan to prevent disclosure of
-- sensitive information such as RLS filters.
EXPLAIN SELECT catgroup, count(*) FROM tickit_category_redshift GROUP BY catgroup ORDER
 BY catgroup;

-- Change session to security administrator bob.
SET SESSION AUTHORIZATION bob;
```

```
-- Grant IGNORE RLS permission so that RLS policies do not get applicable to role
 dbadmin.
GRANT IGNORE RLS TO ROLE dbadmin;

-- Grant EXPLAIN RLS permission so that anyone in role auditor can view complete
 EXPLAIN output.
GRANT EXPLAIN RLS TO ROLE auditor;

-- Change session to dbadmin molly.
SET SESSION AUTHORIZATION molly;

-- Check that all tuples are visible to dbadmin molly because `IGNORE RLS` is granted
 to role dbadmin.
SELECT catgroup, count(*)
FROM tickit_category_redshift
GROUP BY catgroup ORDER BY catgroup;

-- Change session to auditor bruce.
SET SESSION AUTHORIZATION bruce;

-- Check explain plan is visible to auditor bruce because `EXPLAIN RLS` is granted to
 role auditor.
EXPLAIN SELECT catgroup, count(*) FROM tickit_category_redshift GROUP BY catgroup ORDER
 BY catgroup;

-- Change session to security administrator bob.
SET SESSION AUTHORIZATION bob;

DETACH RLS POLICY policy_concerts ON tickit_category_redshift FROM ROLE analyst, ROLE
 dbadmin;

-- Change session to analyst alice.
SET SESSION AUTHORIZATION alice;

-- Check that no tuples are visible to analyst alice.
-- Although the policy is detached, no tuples will be visible to analyst alice
-- because of default deny all policy is applied if the table has RLS on.
SELECT catgroup, count(*)
FROM tickit_category_redshift
GROUP BY catgroup ORDER BY catgroup;

-- Change session to security administrator bob.
SET SESSION AUTHORIZATION bob;
```

```
CREATE RLS POLICY policy_events
WITH (eventid INTEGER) AS ev
USING (
    ev.eventid IN (SELECT eventid FROM tickit_sales_redshift WHERE qtysold <3)
);

ATTACH RLS POLICY policy_events ON tickit_event_redshift TO ROLE analyst;
ATTACH RLS POLICY policy_events ON target_schema.target_event_table TO ROLE consumer;

RESET SESSION AUTHORIZATION;

-- Can not cannot alter type of dependent column.
ALTER TABLE target_schema.target_event_table ALTER COLUMN eventid TYPE float;
ALTER TABLE tickit_event_redshift ALTER COLUMN eventid TYPE float;
ALTER TABLE tickit_sales_redshift ALTER COLUMN eventid TYPE float;
ALTER TABLE tickit_sales_redshift ALTER COLUMN qtysold TYPE float;

-- Can not cannot rename dependent column.
ALTER TABLE target_schema.target_event_table RENAME COLUMN eventid TO renamed_eventid;
ALTER TABLE tickit_event_redshift RENAME COLUMN eventid TO renamed_eventid;
ALTER TABLE tickit_sales_redshift RENAME COLUMN eventid TO renamed_eventid;
ALTER TABLE tickit_sales_redshift RENAME COLUMN qtysold TO renamed_qtysold;

-- Can not drop dependent column.
ALTER TABLE target_schema.target_event_table DROP COLUMN eventid CASCADE;
ALTER TABLE tickit_event_redshift DROP COLUMN eventid CASCADE;
ALTER TABLE tickit_sales_redshift DROP COLUMN eventid CASCADE;
ALTER TABLE tickit_sales_redshift DROP COLUMN qtysold CASCADE;

-- Can not drop lookup table.
DROP TABLE tickit_sales_redshift CASCADE;

-- Change session to security administrator bob.
SET SESSION AUTHORIZATION bob;

DROP RLS POLICY policy_concerts;
DROP RLS POLICY IF EXISTS policy_events;

ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY OFF;

RESET SESSION AUTHORIZATION;

-- Drop users and roles.
```

```
DROP USER bob;
DROP USER alice;
DROP USER joe;
DROP USER molly;
DROP USER bruce;
DROP ROLE analyst;
DROP ROLE consumer;
DROP ROLE auditor FORCE;
DROP ROLE dbadmin FORCE;
```

Segurança de metadados

Assim como a segurança no nível da linha do Amazon Redshift, a segurança de metadados oferece um controle mais granular sobre os metadados. Se a segurança de metadados estiver habilitada para o cluster provisionado ou o grupo de trabalho de tecnologia sem servidor, os usuários poderão ver metadados dos objetos aos quais têm acesso de exibição. Com a segurança de metadados, é possível separar a visibilidade com base nas suas necessidades. Por exemplo, você pode usar um único data warehouse para centralizar todo o armazenamento de dados. No entanto, se você armazenar dados de vários setores, o gerenciamento da segurança poderá se tornar problemático. Com a segurança de metadados habilitada, você pode configurar a visibilidade. Os usuários de um setor podem ter mais visibilidade sobre os objetos, ao mesmo tempo em que você restringe o acesso de exibição a usuários de outro setor. A segurança de metadados dá suporte a todos os tipos de objeto, como esquemas, tabelas, exibições, visões materializadas, procedimentos armazenados, funções definidas pelo usuário e modelos de machine learning.

Os usuários podem ver metadados de objetos sob as seguintes circunstâncias:

- Se o acesso ao objeto for concedido ao usuário.
- Se o acesso ao objeto for concedido a um grupo ou a uma função da qual o usuário faz parte.
- O objeto é público.
- O usuário é o proprietário do objeto do banco de dados.

Para habilitar a segurança de metadados, use o comando ALTER SYSTEM. Esta é a sintaxe de como usar o comando ALTER SYSTEM com segurança de metadados.

```
ALTER SYSTEM SET metadata_security=[true|t|on|false|f|off];
```

Quando você habilita a segurança de metadados, todos os usuários que tenham as permissões necessárias conseguem ver os metadados relevantes de objetos aos quais têm acesso. Se você quiser que apenas determinados usuários consigam ver a segurança dos metadados, conceda a permissão `ACCESS CATALOG` a uma função e, em seguida, atribua a função ao usuário. Para obter mais informações sobre como usar funções para controlar melhor a segurança, consulte Role-based access control.

O exemplo a seguir demonstra como conceder a permissão `ACCESS CATALOG` a uma função e, em seguida, atribuir a função a um usuário. Para obter mais informações sobre como conceder permissões, consulte o comando GRANT.

```
CREATE ROLE sample_metadata_viewer;

GRANT ACCESS CATALOG TO ROLE sample_metadata_viewer;

GRANT ROLE sample_metadata_viewer to salesadmin;
```

Se você preferir usar funções já definidas, as funções definidas pelo sistema `operator`, `secadmin`, `dba` e `superuser` terão todas as permissões necessárias para exibir metadados de objeto. Por padrão, os superusuários conseguem ver o catálogo completo.

```
GRANT ROLE operator to sample_user;
```

Se estiver usando funções para controlar a segurança de metadados, você terá acesso a todas as exibições e funções do sistema que acompanham o controle de acesso baseado em funções. Por exemplo, é possível consultar a visualização SVV_ROLES para ver todos os perfis. Para saber se um usuário é membro de uma função ou de um grupo, use a função USER_IS_MEMBER_OF. Para obter uma lista completa de exibições SVV, consulte você SVV metadata views. Para obter uma lista das funções de informações do sistema, consulte System information functions.

Mascaramento dinâmico de dados

> **Note**
>
> O Amazon Redshift mascara automaticamente determinadas colunas das tabelas do sistema ao registrar informações sobre consultas feitas às visualizações do catálogo de dados para evitar a exposição de metadados confidenciais. Para obter mais informações, consulte Registros em log seguros no Guia de gerenciamento do Amazon Redshift.

Usando o mascaramento dinâmico de dados (DDM) no Amazon Redshift, é possível proteger dados confidenciais no seu data warehouse. Você pode manipular a forma como o Amazon Redshift mostra dados confidenciais para o usuário no momento da consulta, sem transformá-los no banco de dados. Você controla o acesso aos dados por meio de políticas de mascaramento que aplicam regras de ofuscação personalizadas para determinado usuário ou perfil. Dessa forma, você pode responder às mudanças nos requisitos de privacidade sem alterar os dados subjacentes ou editar consultas SQL.

As políticas de mascaramento dinâmico de dados ocultam, ofuscam ou pseudonimizam dados que correspondem a determinado formato. Quando anexada a uma tabela, a expressão de mascaramento é aplicada a uma ou mais de suas colunas. Você pode modificar ainda mais as políticas de mascaramento para aplicá-las somente a determinados usuários, ou a perfis definidos por usuários que podem ser criados com Regras de controle de acesso com base em função (RBAC). Além disso, você pode aplicar o DDM em células usando colunas condicionais ao criar sua política de mascaramento. Para obter mais informações sobre o mascaramento condicional, consulte Mascaramento dinâmico de dados condicional.

Você pode aplicar várias políticas de mascaramento com níveis variados de ofuscação à mesma coluna em uma tabela e atribuí-las a diferentes perfis. Para evitar conflitos quando você tem perfis diferentes com políticas diferentes aplicadas a uma coluna, você pode definir prioridades para cada aplicação. Dessa forma, você pode controlar quais dados determinado usuário ou perfil pode acessar. As políticas de DDM podem redigir dados parcial ou completamente, ou aplicar hash neles usando funções definidas por usuários escritas em SQL, em Python ou com o AWS Lambda. Ao mascarar dados usando hashes, você pode aplicar uniões nesses dados sem acessar informações potencialmente confidenciais.

Comandos SQL para gerenciar políticas de mascaramento dinâmico de dados

Você pode realizar as seguintes ações para criar, anexar, desanexar e excluir políticas de mascaramento dinâmico de dados (DDM):

- Para criar uma política de DDM, use o comando CREATE MASKING POLICY.

  Veja a seguir um exemplo de criação de uma política de mascaramento usando uma função de hash SHA-2.

```
CREATE MASKING POLICY hash_credit
WITH (credit_card varchar(256))
```

```
USING (sha2(credit_card + 'testSalt', 256));
```

- Para alterar uma política de DDM existente, use o comando ALTER MASKING POLICY.

  Veja a seguir um exemplo de como alterar uma política de mascaramento existente.

  ```
  ALTER MASKING POLICY hash_credit
  USING (sha2(credit_card + 'otherTestSalt', 256));
  ```

- Para anexar uma política de DDM em uma tabela a um ou mais usuários ou perfis, use o comando ATTACH MASKING POLICY.

  Veja a seguir um exemplo de como anexar uma política de mascaramento a um par de coluna/perfil.

  ```
   ATTACH MASKING POLICY hash_credit
  ON credit_cards (credit_card)
  TO ROLE science_role
  PRIORITY 30;
  ```

  A cláusula PRIORITY determina qual política de mascaramento se aplica a uma sessão de usuário quando várias políticas estão anexadas à mesma coluna. Por exemplo, se o usuário no exemplo anterior tiver outra política de mascaramento anexada à mesma coluna de cartões de crédito com uma prioridade de 20, a política de science_role se aplicará, pois tem prioridade mais alta (30).

- Para desanexar uma política de DDM em uma tabela de um ou mais usuários ou perfis, use o comando DETACH MASKING POLICY.

  Veja a seguir um exemplo de como desanexar uma política de mascaramento de um par de coluna/perfil.

  ```
  DETACH MASKING POLICY hash_credit
  ON credit_cards(credit_card)
  FROM ROLE science_role;
  ```

- Para descartar uma política de DDM de todos os bancos de dados, use o comando DROP MASKING POLICY.

  Veja a seguir um exemplo de como descartar uma política de mascaramento de todos os bancos de dados.

```
DROP MASKING POLICY hash_credit;
```

Hierarquia de políticas de mascaramento dinâmico de dados

Ao anexar várias políticas de mascaramento, considere o seguinte:

- É possível anexar várias políticas de mascaramento a uma única coluna.
- Quando várias políticas de mascaramento são aplicáveis a uma consulta, a política de maior prioridade anexada a cada coluna respectiva se aplica. Considere o exemplo a seguir.

```
ATTACH MASKING POLICY partial_hash
ON credit_cards(address, credit_card)
TO ROLE analytics_role
PRIORITY 20;

ATTACH MASKING POLICY full_hash
ON credit_cards(credit_card, ssn)
TO ROLE auditor_role
PRIORITY 30;

SELECT address, credit_card, ssn
FROM credit_cards;
```

  Ao executar a instrução SELECT, um usuário com os perfis de analista e auditor vê a coluna de endereços com a política de mascaramento `partial_hash` aplicada. Ele vê as colunas de cartão de crédito e SSN com a política de mascaramento `full_hash` aplicada, porque a política `full_hash` tem a prioridade mais alta na coluna do cartão de crédito.
- Se você não especificar uma prioridade ao anexar uma política de mascaramento, a prioridade padrão será 0.
- Você não pode anexar duas políticas com a mesma prioridade à mesma coluna.
- Você não pode anexar duas políticas à mesma combinação de usuário e coluna ou função e coluna.
- Quando várias políticas de mascaramento são aplicáveis ao longo do mesmo caminho SUPER enquanto anexadas ao mesmo usuário ou função, somente o anexo de prioridade mais alta entra em vigor. Considere os seguintes exemplos:

O primeiro exemplo mostra duas políticas de mascaramento anexadas no mesmo caminho, com a política de prioridade mais alta entrando em vigor.

```
ATTACH MASKING POLICY hide_name
ON employees(col_person.name)
TO PUBLIC
PRIORITY 20;

ATTACH MASKING POLICY hide_last_name
ON employees(col_person.name.last)
TO PUBLIC
PRIORITY 30;

--Only the hide_last_name policy takes effect.
SELECT employees.col_person.name FROM employees;
```

O segundo exemplo mostra duas políticas de mascaramento anexadas a caminhos diferentes no mesmo objeto SUPER, sem conflito entre as políticas. Ambos os anexos serão aplicados simultaneamente.

```
ATTACH MASKING POLICY hide_first_name
ON employees(col_person.name.first)
TO PUBLIC
PRIORITY 20;

ATTACH MASKING POLICY hide_last_name
ON employees(col_person.name.last)
TO PUBLIC
PRIORITY 20;

--Both col_person.name.first and col_person.name.last are masked.
SELECT employees.col_person.name FROM employees;
```

Para confirmar qual política de mascaramento se aplica a uma determinada combinação de usuário e coluna ou função, os usuários com a função `sys:secadmin` podem consultar o par de coluna/perfil ou coluna/usuário na exibição do sistema SVV_ATTACHED_MASKING_POLICY. Para obter mais informações, consulte Visualizações do sistema de mascaramento dinâmico de dados.

Uso do mascaramento de dados dinâmico com caminhos do tipo de dados SUPER

O Amazon Redshift dá suporte à anexação de políticas de mascaramento de dados dinâmicas a caminhos das colunas do tipo SUPER. Para obter mais informações sobre tipos de dados SUPER, consulte Dados semiestruturados no Amazon Redshift.

Ao anexar políticas de mascaramento a caminhos de colunas do tipo SUPER, considere o seguinte.

- Ao anexar uma política de mascaramento a um caminho em uma coluna, essa coluna deve ser definida como o tipo de dados SUPER. Você só pode aplicar políticas de mascaramento a valores escalares no caminho SUPER. Você não pode aplicar políticas de mascaramento a estruturas ou matrizes complexas.
- Você pode aplicar políticas de mascaramento diferentes a vários valores escalares em uma única coluna SUPER, desde que os caminhos SUPER não entrem em conflito. Por exemplo, os caminhos SUPER `a.b` e `a.b.c` estão em conflito porque estão no mesmo caminho, com `a.b` sendo o pai de `a.b.c`. Os caminhos SUPER `a.b.c` e `a.b.d` não entram em conflito.
- O Amazon Redshift não consegue verificar se os caminhos anexados por uma política de mascaramento existem nos dados e são do tipo esperado até que a política seja aplicada no runtime da consulta do usuário. Por exemplo, quando você anexar uma política de mascaramento que mascara valores TEXT a um caminho SUPER contendo um valor INT, o Amazon Redshift tentará converter o tipo do valor no caminho.

  Nessas situações, o comportamento do Amazon Redshift no runtime depende das definições de configuração para consulta de objetos SUPER. Por padrão, o Amazon Redshift está em modo relaxado e vai resolver caminhos não encontrados e conversões inválidas como `NULL` para o caminho SUPER indicado. Para obter mais informações sobre definições de configuração relacionadas a SUPER, consulte Configurações SUPER.
- SUPER é um tipo sem esquema, o que significa que o Amazon Redshift não consegue confirmar a existência do valor em um determinado caminho SUPER. Se você anexar uma política de mascaramento a um caminho SUPER não existente e o Amazon Redshift estiver em modo flexível, o Amazon Redshift vai resolver o caminho para um valor `NULL`. É recomendável considerar o formato esperado de objetos SUPER e a probabilidade de terem atributos inesperados durante a anexação das políticas de mascaramento aos caminhos de colunas SUPER. Se você acha que possa haver um esquema inesperado na coluna SUPER, considere anexar diretamente as políticas de mascaramento à coluna SUPER. Você pode usar as funções de informações do tipo SUPER para verificar atributos e tipos e usar `OBJECT_TRANSFORM` para mascarar os valores.

Para obter mais informações sobre funções de informações do tipo SUPER, consulte Funções de informação de tipo SUPER.

Exemplos

Anexação das políticas de mascaramento a caminhos SUPER

O exemplo a seguir anexa várias políticas de mascaramento a vários caminhos do tipo SUPER em uma coluna.

```
CREATE TABLE employees (
    col_person SUPER
);

INSERT INTO employees
VALUES
    (
        json_parse('
            {
                "name": {
                    "first": "John",
                    "last": "Doe"
                },
                "age": 25,
                "ssn": "111-22-3333",
                "company": "Company Inc."
            }
        ')
    ),
    (
        json_parse('
            {
                "name": {
                    "first": "Jane",
                    "last": "Appleseed"
                },
                "age": 34,
                "ssn": "444-55-7777",
                "company": "Organization Org."
            }
        ')
    )
```

```
;
GRANT ALL ON ALL TABLES IN SCHEMA "public" TO PUBLIC;

-- Create the masking policies.

-- This policy converts the given name to all uppercase letters.
CREATE MASKING POLICY mask_first_name
WITH(first_name TEXT)
USING ( UPPER(first_name) );

-- This policy replaces the given name with the fixed string 'XXXX'.
CREATE MASKING POLICY mask_last_name
WITH(last_name TEXT)
USING ( 'XXXX'::TEXT );

-- This policy rounds down the given age to the nearest 10.
CREATE MASKING POLICY mask_age
WITH(age INT)
USING ( (FLOOR(age::FLOAT / 10) * 10)::INT );

-- This policy converts the first five digits of the given SSN to 'XXX-XX'.
CREATE MASKING POLICY mask_ssn
WITH(ssn TEXT)
USING ( 'XXX-XX-'::TEXT || SUBSTRING(ssn::TEXT FROM 8 FOR 4) );

-- Attach the masking policies to the employees table.
ATTACH MASKING POLICY mask_first_name
ON employees(col_person.name.first)
TO PUBLIC;

ATTACH MASKING POLICY mask_last_name
ON employees(col_person.name.last)
TO PUBLIC;

ATTACH MASKING POLICY mask_age
ON employees(col_person.age)
TO PUBLIC;

ATTACH MASKING POLICY mask_ssn
ON employees(col_person.ssn)
TO PUBLIC;

-- Verify that your masking policies are attached.
SELECT
```

```
    policy_name,
    TABLE_NAME,
    priority,
    input_columns,
    output_columns
FROM
    svv_attached_masking_policy;

   policy_name   | table_name | priority |           input_columns           |
 output_columns
-----------------+------------+----------+-----------------------------------
+-----------------------------------
 mask_age        | employees  |        0 | ["col_person.\"age\""]            |
 ["col_person.\"age\""]
 mask_first_name | employees  |        0 | ["col_person.\"name\".\"first\""] |
 ["col_person.\"name\".\"first\""]
 mask_last_name  | employees  |        0 | ["col_person.\"name\".\"last\""]  |
 ["col_person.\"name\".\"last\""]
 mask_ssn        | employees  |        0 | ["col_person.\"ssn\""]            |
 ["col_person.\"ssn\""]
(4 rows)

-- Observe the masking policies taking effect.
SELECT col_person FROM employees ORDER BY col_person.age;

-- This result is formatted for ease of reading.
         col_person
--------------------------------
{
    "name": {
        "first": "JOHN",
        "last": "XXXX"
    },
    "age": 20,
    "ssn": "XXX-XX-3333",
    "company": "Company Inc."
}
{
    "name": {
        "first": "JANE",
        "last": "XXXX"
    },
    "age": 30,
    "ssn": "XXX-XX-7777",
```

```
    "company": "Organization Org."
}
```

Estes são alguns exemplos de anexos da política de mascaramento inválidos aos caminhos SUPER.

```
-- This attachment fails because there is already a policy
-- with equal priority attached to employees.name.last, which is
-- on the same SUPER path as employees.name.
ATTACH MASKING POLICY mask_ssn
ON employees(col_person.name)
TO PUBLIC;
ERROR:  DDM policy "mask_last_name" is already attached on relation "employees" column
 "col_person."name"."last"" with same priority

-- Create a masking policy that masks DATETIME objects.
CREATE MASKING POLICY mask_date
WITH(INPUT DATETIME)
USING ( INPUT );

-- This attachment fails because SUPER type columns can't contain DATETIME objects.
ATTACH MASKING POLICY mask_date
ON employees(col_person.company)
TO PUBLIC;
ERROR:  cannot attach masking policy for output of type "timestamp without time zone"
 to column "col_person."company"" of type "super
```

Este é um exemplo de anexação de uma política de mascaramento a um caminho SUPER não existente. Por padrão, o Amazon Redshift vai resolver o caminho para `NULL`.

```
ATTACH MASKING POLICY mask_first_name
ON employees(col_person.not_exists)
TO PUBLIC;

SELECT col_person FROM employees LIMIT 1;

-- This result is formatted for ease of reading.
         col_person
---------------------------------
{
    "name": {
        "first": "JOHN",
        "last": "XXXX"
    },
```

```
    "age": 20,
    "ssn": "XXX-XX-3333",
    "company": "Company Inc.",
    "not_exists": null
}
```

Mascaramento dinâmico de dados condicional

Você pode mascarar dados no nível de célula criando políticas de mascaramento com expressões condicionais na expressão de mascaramento. Por exemplo, você pode criar uma política de mascaramento que aplica máscaras diferentes a um valor, dependendo do valor de outra coluna nessa linha.

Veja a seguir um exemplo de como usar o mascaramento condicional de dados para criar e anexar uma política de mascaramento que edita parcialmente os números de cartão de crédito envolvidos em fraudes e oculta completamente todos os outros números de cartão de crédito. Você deve ser um superusuário ou ter o perfil `sys:secadmin` para executar este exemplo.

```
--Create an analyst role.
CREATE ROLE analyst;

--Create a credit card table. The table contains an is_fraud boolean column,
--which is TRUE if the credit card number in that row was involved in a fraudulent
 transaction.
CREATE TABLE credit_cards (id INT, is_fraud BOOLEAN, credit_card_number VARCHAR(16));

--Create a function that partially redacts credit card numbers.
CREATE FUNCTION REDACT_CREDIT_CARD (credit_card VARCHAR(16))
RETURNS VARCHAR(16) IMMUTABLE
AS $$
    import re
    regexp = re.compile("^([0-9]{6})[0-9]{5,6}([0-9]{4})")

    match = regexp.search(credit_card)
    if match != None:
        first = match.group(1)
        last = match.group(2)
    else:
        first = "000000"
        last = "0000"

    return "{}XXXXX{}".format(first, last)
```

```
$$ LANGUAGE plpythonu;

--Create a masking policy that partially redacts credit card numbers if the is_fraud
 value for that row is TRUE,
--and otherwise blanks out the credit card number completely.
CREATE MASKING POLICY card_number_conditional_mask
    WITH (fraudulent BOOLEAN, pan varchar(16))
    USING (CASE WHEN fraudulent THEN REDACT_CREDIT_CARD(pan)
                ELSE Null
           END);

--Attach the masking policy to the credit_cards/analyst table/role pair.
ATTACH MASKING POLICY card_number_conditional_mask ON credit_cards (credit_card_number)
 USING (is_fraud, credit_card_number)
 TO ROLE analyst PRIORITY 100;
```

Visualizações do sistema de mascaramento dinâmico de dados

Superusuários, usuários com a função `sys:operator` e usuários com a permissão ACCESS SYSTEM TABLE podem acessar as visualizações de sistema relacionadas a DDM a seguir.

- SVV_MASKING_POLICY

  Use SVV_MASKING_POLICY para visualizar todas as políticas de mascaramento criadas no cluster ou grupo de trabalho.
- SVV_ATTACHED_MASKING_POLICY

  Use SVV_ATTACHED_MASKING_POLICY para visualizar todas as relações e usuários ou funções com políticas anexadas no banco de dados atualmente conectado.
- SYS_APPLIED_MASKING_POLICY_LOG

  Use SYS_APPLIED_MASKING_POLICY_LOG para monitorar a aplicação de políticas de mascaramento em consultas que fazem referência a relações protegidas por DDM.

Veja a seguir alguns exemplos das informações que você pode encontrar usando as visualizações do sistema.

```
--Select all policies associated with specific users, as opposed to roles
SELECT policy_name,
       schema_name,
```

```
       table_name,
       grantee
FROM svv_attached_masking_policy
WHERE grantee_type = 'user';

--Select all policies attached to a specific user
SELECT policy_name,
       schema_name,
       table_name,
       grantee
FROM svv_attached_masking_policy
WHERE grantee = 'target_grantee_name'

--Select all policies attached to a given table
SELECT policy_name,
       schema_name,
       table_name,
       grantee
FROM svv_attached_masking_policy
WHERE table_name = 'target_table_name'
      AND schema_name = 'target_schema_name';

--Select the highest priority policy attachment for a given role
SELECT samp.policy_name,
       samp.priority,
       samp.grantee,
       smp.policy_expression
FROM svv_masking_policy AS smp
JOIN svv_attached_masking_policy AS samp
    ON samp.policy_name = smp.policy_name
WHERE
    samp.grantee_type = 'role' AND
    samp.policy_name = mask_get_policy_for_role_on_column(
        'target_schema_name',
        'target_table_name',
        'target_column_name',
        'target_role_name')
ORDER BY samp.priority desc
LIMIT 1;

--See which policy a specific user will see on a specific column in a given relation
SELECT samp.policy_name,
       samp.priority,
       samp.grantee,
```

```
       smp.policy_expression
FROM svv_masking_policy AS smp
JOIN svv_attached_masking_policy AS samp
    ON samp.policy_name = smp.policy_name
WHERE
    samp.grantee_type = 'role' AND
    samp.policy_name = mask_get_policy_for_user_on_column(
        'target_schema_name',
        'target_table_name',
        'target_column_name',
        'target_user_name')
ORDER BY samp.priority desc;

 --Select all policies attached to a given relation.
SELECT policy_name,
schema_name,
relation_name,
database_name
FROM sys_applied_masking_policy_log
WHERE relation_name = 'relation_name'
AND schema_name = 'schema_name';
```

Considerações ao usar o mascaramento dinâmico de dados

Ao usar o mascaramento dinâmico de dados, considere o seguinte:

- Ao consultar objetos criados com base em tabelas, como visualizações, os usuários verão os resultados com base em suas próprias políticas de mascaramento, não nas políticas do usuário que criou os objetos. Por exemplo, um usuário com o perfil de analista consultando uma visualização criada por um secadmin verá resultados com as políticas de mascaramento associadas ao perfil de analista.
- Para evitar que o comando EXPLAIN exponha filtros confidenciais de políticas de mascaramento, somente usuários com a permissão SYS_EXPLAIN_DDM podem ver as políticas de mascaramento aplicadas nas saídas do EXPLAIN. Os usuários não têm a permissão SYS_EXPLAIN_DDM por padrão.

  A sintaxe a seguir concede a permissão a um perfil.

  ```
  GRANT EXPLAIN MASKING TO ROLE rolename
  ```

  Para obter mais informações sobre o comando EXPLAIN, consulte EXPLAIN.

- Usuários com perfis diferentes podem ver resultados diferentes com base nas condições do filtro ou nas condições de união usadas. Por exemplo, a execução de um comando SELECT em uma tabela usando um valor de coluna específico falhará se o usuário que executa o comando tiver uma política de mascaramento aplicada que ofusque essa coluna.
- As políticas de DDM devem ser aplicadas antes de qualquer operação ou projeção de predicados. As políticas de mascaramento podem incluir o seguinte:
  - Operações constantes de baixo custo, como converter um valor em nulo
  - Operações de custo moderado, como hashing HMAC
  - Operações de alto custo, como chamadas para funções externas do Lambda definidas por usuários

  Assim, recomendamos o uso de expressões de mascaramento simples sempre que possível.
- Você pode usar políticas de DDM para perfis com políticas de segurança por linha, mas observe que as políticas de RLS são aplicadas antes do DDM. Uma expressão de mascaramento de dados dinâmica não conseguirá ler uma linha protegida por RLS. Para obter mais informações sobre RLS, consulte Segurança por linha.
- Ao usar o comando COPY para copiar de parquet para tabelas de destino protegidas, você deve especificar explicitamente as colunas na instrução COPY. Para obter mais informações sobre como mapear colunas com COPY, consulte Opções de mapeamento da coluna.
- As políticas DDM não podem ser anexadas às seguintes relações:
  - Tabelas e catálogos do sistema
  - Tabelas externas
  - Tabelas de compartilhamento de dados
  - Relações entre bancos de dados
  - Tabelas temporárias
  - Consultas correlacionadas
- As políticas DDM podem conter tabelas de pesquisa. As tabelas de pesquisa podem estar presentes na cláusula USING. Os seguintes tipos de relação não podem ser usados como tabelas de pesquisa:
  - Tabelas e catálogos do sistema
  - Tabelas externas
  - Tabelas de compartilhamento de dados
  - Exibições, visões materializadas e exibições de vinculação tardia

- Relações entre bancos de dados
- Tabelas temporárias
- Consultas correlacionadas

Veja a seguir um exemplo de como anexar uma política de mascaramento a uma tabela de pesquisa.

```
--Create a masking policy referencing a lookup table
CREATE MASKING POLICY lookup_mask_credit_card WITH (credit_card TEXT) USING (
  CASE
    WHEN
      credit_card IN (SELECT credit_card_lookup FROM credit_cards_lookup)
    THEN '000000XXXX0000'
    ELSE REDACT_CREDIT_CARD(credit_card)
    END
  );

--Provides access to the lookup table via a policy attached to a role
GRANT SELECT ON TABLE credit_cards_lookup TO MASKING POLICY lookup_mask_credit_card;
```

- Você não pode anexar uma política de mascaramento que produza uma saída incompatível com o tipo e o tamanho da coluna de destino. Por exemplo, você não pode anexar uma política de mascaramento que gera uma string de 12 caracteres em uma coluna VARCHAR(10). O Amazon Redshift é compatível com as exceções a seguir:
  - Uma política de mascaramento com o tipo de entrada INTN pode ser anexada a uma política com tamanho INTM, desde que M < N. Por exemplo, uma política de entrada BIGINT (INT8) pode ser anexada a uma coluna smallint (INT4).
  - Uma política de mascaramento com o tipo de entrada NUMERIC ou DECIMAL sempre pode ser anexada a uma coluna FLOAT.
- Não é possível usar políticas de DDM com o compartilhamento de dados. Se o produtor de dados da unidade de compartilhamento de dados anexar uma política de DDM a uma tabela na unidade de compartilhamento de dados, a tabela se tornará inacessível aos usuários do consumidor de dados que estão tentando consultar a tabela. A tentativa de adicionar a relação com uma unidade de compartilhamento de dados falhará no cluster ou namespace do lado do produtor com o seguinte erro:

```
<ddm_protected_relation> or a relation dependent on it is protected by a masking
 policy and cannot be added to a datashare
```

Se você anexar uma política de mascaramento a uma relação no lado do produtor e ela já estiver incluída em uma unidade de compartilhamento de dados, a tentativa de consultar a relação no lado do consumidor falhará com o seguinte erro:

```
cross-cluster query of the masked relation <ddm_protected_relation> is not supported.
```

É possível desativar o DDM para unidades de compartilhamento de dados usando o comando ALTER TABLE com o parâmetro MASKING OFF FOR DATASHARES. Para obter mais informações, consulte ALTER TABLE.

- Você não poderá consultar relações que tenham políticas de DDM anexadas se os valores para qualquer uma das opções de configuração a seguir não corresponderem ao valor padrão da sessão:
  - `enable_case_sensitive_super_attribute`
  - `enable_case_sensitive_identifier`
  - `downcase_delimited_identifier`

  Considere redefinir as opções de configuração da sessão se você tentar consultar uma relação com uma política DDM anexada e vir a mensagem “DDM protected relation does not support session level config on case sensitivity being different from its default value”.
- Quando o cluster provisionado ou namespace sem servidor tem alguma política de mascaramento dinâmico de dados, os seguintes comandos são bloqueados para usuários comuns:

  ```
  ALTER <current_user> SET enable_case_sensitive_super_attribute/
  enable_case_sensitive_identifier/downcase_delimited_identifier
  ```

  Ao criar políticas de DDM, recomendamos que você altere as definições de opções da configuração padrão para usuários comuns a fim de que correspondam às definições de opções da configuração da sessão no momento em que a política foi criada. Superusuários e usuários com o privilégio ALTER USER podem fazer isso usando as configurações do grupo de parâmetros ou o comando ALTER USER. Para obter informações sobre grupos de parâmetros, consulte Amazon Redshift parameter groups no Guia de gerenciamento do Amazon Redshift. Para obter mais informações sobre o comando ALTER USER, consulte ALTER USER.
- As exibições e as exibições de vinculação tardia com políticas DDM não podem ser substituídas por usuários regulares usando o comando CREATE VIEW. Para substituir exibições ou LBVs por políticas DDM, primeiro desanexe todas as políticas DDM anexadas, substitua as exibições

ou LBVs e reanexe as políticas. Os superusuários e usuários com a permissão `sys:secadmin` podem usar CREATE VIEW em exibições ou LBVs com políticas DDM sem desanexar as políticas.

- As exibições com políticas DDM anexadas não podem referenciar tabelas e exibições de sistema. Exibições de vinculação tardia podem referenciar tabelas e exibições de sistema.
- As exibições de vinculação tardia com políticas DDM anexadas não podem referenciar dados aninhados em data lakes, como documentos JSON.
- As exibições de vinculação tardia não poderão ter políticas DDM anexadas se essa exibição de vinculação tardia for referenciada por qualquer exibição.
- As políticas DDM anexadas às exibições de vinculação tardia são anexadas pelo nome da coluna. No momento da consulta, o Amazon Redshift valida se todas as políticas de mascaramento anexadas à exibição de vinculação tardia foram aplicadas com êxito e se o tipo de coluna de saída da exibição de vinculação tardia corresponde aos tipos nas políticas de mascaramento anexadas. Se a validação falhar, o Amazon Redshift retornará um erro para a consulta.
- Você pode usar variáveis de contexto de sessão personalizadas ao criar políticas de DDM. O exemplo a seguir define variáveis de contexto de sessão para uma política de DDM.

```
-- Set a customized context variable.
SELECT set_config('app.city', 'XXXX', FALSE);

-- Create a MASKING policy using current_setting() to get the value of a customized
 context variable.
CREATE MASKING POLICY city_mask
WITH (city VARCHAR(30))
USING (current_setting('app.city')::VARCHAR(30));

-- Attach the policy on the target table to one or more roles.
ATTACH MASKING POLICY city_mask
ON tickit_users_redshift(city)
TO ROLE analyst, ROLE dbadmin;
```

Para obter detalhes sobre como definir e recuperar variáveis de contexto de sessão personalizadas, acesse SET, SET_CONFIG, SHOW, CURRENT_SETTING e RESET. Para obter mais informações sobre como modificar a configuração do servidor em geral, acesse Modificar a configuração do servidor.

> **Important**
>
> Quando são usadas variáveis de contexto de sessão em políticas de DDM, a política de segurança depende do usuário ou da função que invoca a política. Tenha cuidado para evitar vulnerabilidades de segurança ao usar variáveis de contexto de sessão em políticas de DDM.

Exemplo completo do mascaramento dinâmico de dados

Veja a seguir um exemplo completo que mostra como criar e anexar políticas de mascaramento a uma coluna. Essas políticas permitem que os usuários acessem uma coluna e vejam valores diferentes, dependendo do grau de ofuscação nas políticas associadas aos seus perfis. Você deve ser um superusuário ou ter o perfil `sys:secadmin` para executar este exemplo.

Criar uma política de mascaramento

Primeiro, crie uma tabela e preencha-a com valores de cartões de crédito.

```
--create the table
CREATE TABLE credit_cards (
  customer_id INT,
  credit_card TEXT
);

--populate the table with sample values
INSERT INTO credit_cards
VALUES
  (100, '4532993817514842'),
  (100, '4716002041425888'),
  (102, '5243112427642649'),
  (102, '6011720771834675'),
  (102, '6011378662059710'),
  (103, '373611968625635')
;

--run GRANT to grant permission to use the SELECT statement on the table
GRANT SELECT ON credit_cards TO PUBLIC;

--create two users
```

```
CREATE USER regular_user WITH PASSWORD '1234Test!';

CREATE USER analytics_user WITH PASSWORD '1234Test!';

--create the analytics_role role and grant it to analytics_user
--regular_user does not have a role
CREATE ROLE analytics_role;

GRANT ROLE analytics_role TO analytics_user;
```

Depois, crie uma política de mascaramento para aplicar ao perfil de análise.

```
--create a masking policy that fully masks the credit card number
CREATE MASKING POLICY mask_credit_card_full
WITH (credit_card VARCHAR(256))
USING ('000000XXXX0000'::TEXT);

--create a user-defined function that partially obfuscates credit card data
CREATE FUNCTION REDACT_CREDIT_CARD (credit_card TEXT)
RETURNS TEXT IMMUTABLE
AS $$
    import re
    regexp = re.compile("^([0-9]{6})[0-9]{5,6}([0-9]{4})")

    match = regexp.search(credit_card)
    if match != None:
        first = match.group(1)
        last = match.group(2)
    else:
        first = "000000"
        last = "0000"

    return "{}XXXXX{}".format(first, last)
$$ LANGUAGE plpythonu;

--create a masking policy that applies the REDACT_CREDIT_CARD function
CREATE MASKING POLICY mask_credit_card_partial
WITH (credit_card VARCHAR(256))
USING (REDACT_CREDIT_CARD(credit_card));

--confirm the masking policies using the associated system views
SELECT * FROM svv_masking_policy;
```

```
SELECT * FROM svv_attached_masking_policy;
```

Anexar uma política de mascaramento

Anexe as políticas de mascaramento à tabela de cartões de crédito.

```
--attach mask_credit_card_full to the credit card table as the default policy
--all users will see this masking policy unless a higher priority masking policy is
 attached to them or their role
ATTACH MASKING POLICY mask_credit_card_full
ON credit_cards(credit_card)
TO PUBLIC;

--attach mask_credit_card_partial to the analytics role
--users with the analytics role can see partial credit card information
ATTACH MASKING POLICY mask_credit_card_partial
ON credit_cards(credit_card)
TO ROLE analytics_role
PRIORITY 10;

--confirm the masking policies are applied to the table and role in the associated
 system view
SELECT * FROM svv_attached_masking_policy;

--confirm the full masking policy is in place for normal users by selecting from the
 credit card table as regular_user
SET SESSION AUTHORIZATION regular_user;

SELECT * FROM credit_cards;

--confirm the partial masking policy is in place for users with the analytics role by
 selecting from the credit card table as analytics_user
SET SESSION AUTHORIZATION analytics_user;

SELECT * FROM credit_cards;
```

Alterar uma política de mascaramento

A seção a seguir mostra como alterar uma política de mascaramento dinâmico de dados.

```
--reset session authorization to the default
RESET SESSION AUTHORIZATION;
```

```
--alter the mask_credit_card_full policy
ALTER MASKING POLICY mask_credit_card_full
USING ('00000000000000'::TEXT);

--confirm the full masking policy is in place after altering the policy, and that
 results are altered from '000000XXXX0000' to '00000000000000'
SELECT * FROM credit_cards;
```

Desanexar e descartar uma política de mascaramento

A seção a seguir mostra como desanexar e descartar políticas de mascaramento removendo todas as políticas de mascaramento dinâmico de dados da tabela.

```
--reset session authorization to the default
RESET SESSION AUTHORIZATION;

--detach both masking policies from the credit_cards table
DETACH MASKING POLICY mask_credit_card_full
ON credit_cards(credit_card)
FROM PUBLIC;

DETACH MASKING POLICY mask_credit_card_partial
ON credit_cards(credit_card)
FROM ROLE analytics_role;

--drop both masking policies
DROP MASKING POLICY mask_credit_card_full;

DROP MASKING POLICY mask_credit_card_partial;
```

Permissões em escopo

Com as permissões em escopo, é possível conceder permissões a um usuário ou função em todos os objetos de um tipo em um banco de dados ou esquema. Usuários e funções com permissões em escopo têm as permissões especificadas em todos os objetos atuais e futuros no banco de dados ou esquema.

Você pode visualizar o escopo das permissões com escopo em nível de banco de dados em SVV_DATABASE_PRIVILEGES. Você pode visualizar o escopo das permissões com escopo em nível de esquema em SVV_SCHEMA_PRIVILEGES.

Para obter mais informações sobre a aplicação de permissões em escopo definido, consulte GRANT e REVOKE.

Considerações sobre como usar permissões em escopo

Ao usar permissões em escopo, considere o seguinte:

- Você pode usar permissões em escopo a fim de conceder ou revogar permissões em um escopo de banco de dados ou esquema de uma função ou de um usuário especificado.
- Não é possível conceder permissões em escopo a grupos de usuários.
- A concessão ou a revogação de permissões em escopo altera permissões de todos os objetos atuais e futuros no escopo.
- As permissões em escopo e em nível de objeto operam independentemente uma da outra. Por exemplo, um usuário manterá as permissões em uma tabela nos dois casos a seguir.
  - O usuário recebe a permissão SELECT na tabela schema1.table1 e a permissão SELECT em escopo na tabela schema1. O usuário revoga SELECT para todas as tabelas no esquema schema1. O usuário retém SELECT em schema1.table1.
  - O usuário recebe a permissão SELECT na tabela schema1.table1 e a permissão SELECT em escopo na tabela schema1. O usuário revoga SELECT para schema1.table1. O usuário retém SELECT em schema1.table1.
- Para conceder ou revogar permissões em escopo, você deve atender a um dos seguintes critérios:
  - Superusuários.
  - Usuários com a opção de concessão dessa permissão. Para obter mais informações sobre opções de concessão, consulte o parâmetro WITH GRANT OPTION em GRANT.
- As permissões em escopo só podem ser concedidas a ou revogadas de objetos do banco de dados conectado ou de bancos de dados importados de uma unidade de compartilhamento de dados.
- Você pode usar permissões em escopo para definir as permissões padrão em um banco de dados criado em uma unidade de compartilhamento de dados. Um usuário da unidade de compartilhamento de dados no lado do consumidor que recebe permissões em escopo em um banco de dados compartilhado vai receber automaticamente essas permissões para qualquer novo objeto adicionado à unidade de compartilhamento de dados no lado do produtor.
- Os produtores podem conceder permissões em escopo em objetos dentro de um esquema para uma unidade de compartilhamento de dados. (pré-visualização)

Referência SQL

Com o Amazon Redshift, é possível utilizar SQL para consultar e analisar com eficiência grandes quantidades de dados armazenados no data warehouse. A referência SQL abrange a sintaxe e o uso de comandos SQL, tipos de dados, funções, operadores e muito mais, permitindo que você extraia insights e tome decisões orientadas por dados. Você pode consultar essa referência para escrever consultas otimizadas, criar e gerenciar objetos de banco de dados e realizar transformações de dados complexas. A referência a seguir fornece detalhes abrangentes sobre a utilização de SQL no ambiente do Amazon Redshift para atender aos seus requisitos de análise de dados.

Tópicos

- SQL do Amazon Redshift
- Uso de SQL
- Comandos SQL
- Referência de funções SQL
- Palavras reservadas

SQL do Amazon Redshift

Tópicos

- Funções SQL compatíveis no nó de liderança
- Amazon Redshift e PostgreSQL

O Amazon Redshift é construído em torno do SQL padrão da indústria, com funcionalidade adicional para gerenciar conjuntos de dados muito grandes e oferecer suporte a análises e relatórios de alta performance desses dados.

> **Note**
>
> O tamanho máximo para uma única instrução SQL do Amazon Redshift é 16 MB.

Funções SQL compatíveis no nó de liderança

Algumas consultas do Amazon Redshift são distribuídas e executadas em nós de computação, e outras consultas são executadas exclusivamente no nó líder.

O nó de liderança distribui a SQL aos nós de computação sempre que uma consulta se refere a tabelas criadas pelo usuário ou tabelas de sistema (tabelas com um prefixo STL ou STV e exibições de sistema com um prefixo SVL ou SVV). Uma consulta que se refere apenas a tabelas do catálogo (tabelas com um prefixo PG, tal como PG_TABLE_DEF, que reside no nó de liderança) ou que não se refere a qualquer tabela é executada exclusivamente no nó de liderança.

Algumas funções do Amazon Redshift SQL são suportadas apenas no nó líder e não são suportadas nos nós de computação. Uma consulta que usa uma função de nó líder deve ser executada exclusivamente no nó líder, não nos nós de computação, ou retornará um erro.

A documentação de cada função que deve ser executada exclusivamente no nó líder inclui uma nota informando que a função retornará um erro se fizer referência a tabelas definidas pelo usuário ou tabelas do sistema Amazon Redshift. Consulte Função de apenas nó líder para obter uma lista de funções executadas exclusivamente no nó de liderança.

Exemplos

Os exemplos a seguir usam o banco de dados TICKIT para elucidação. Para ter mais informações sobre o banco de dados de exemplo, acesse Banco de dados de exemplo.

CURRENT_SCHEMA

A função CURRENT_SCHEMA é uma função apenas do nó de liderança. Neste exemplo, a consulta não faz referência a uma tabela, portanto ela é executada exclusivamente no nó de liderança.

```
select current_schema();

current_schema
---------------
public
```

No próximo exemplo, a consulta faz referência a uma tabela de catálogo do sistema, portanto ela é executada exclusivamente no nó de liderança.

```
select * from pg_table_def
```

```
where schemaname = current_schema() limit 1;

 schemaname | tablename | column |   type   | encoding | distkey | sortkey | notnull
------------+-----------+--------+----------+----------+---------+---------+---------
 public     | category  | catid  | smallint | none     | t       |       1 | t
```

No próximo exemplo, a consulta faz referência a uma tabela do sistema Amazon Redshift que reside nos nós de computação, portanto, ela retorna um erro.

```
select current_schema(), userid from users;

INFO:  Function "current_schema()" not supported.
ERROR:  Specified types or functions (one per INFO message) not supported on Amazon
 Redshift tables.
```

SUBSTR

SUBSTR também é uma função exclusiva do nó líder. No exemplo a seguir, a consulta é exclusiva no nó líder porque não referencia uma tabela.

```
SELECT SUBSTR('amazon', 5);

+--------+
| substr |
+--------+
| on     |
+--------+
```

No exemplo a seguir, a consulta referencia uma tabela residente nos nós de computação. Isso resulta em um erro.

```
SELECT SUBSTR(catdesc, 1) FROM category LIMIT 1;

ERROR: SUBSTR() function is not supported (Hint: use SUBSTRING instead)
```

Para executar com êxito a consulta anterior, use SUBSTRING.

```
SELECT SUBSTRING(catdesc, 1) FROM category LIMIT 1;

+--------------------------------+
|           substring            |
```

```
+---------------------------------+
| National Basketball Association |
+---------------------------------+
```

FACTORIAL()

FACTORIAL() é uma função exclusiva do nó líder. No exemplo a seguir, a consulta é exclusiva no nó líder porque não referencia uma tabela.

```
SELECT FACTORIAL(5);

 factorial
-------------
 120
```

No exemplo a seguir, a consulta referencia uma tabela residente nos nós de computação. Isso resulta em um erro quando se utiliza o editor de consultas v2.

```
create table t(a int);
insert into t values (5);
select factorial(a) from t;

ERROR: Specified types or functions (one per INFO message) not supported on Redshift
 tables.
Info: Function "factorial(bigint)" not supported.
```

Amazon Redshift e PostgreSQL

Tópicos

- JDBC e ODBC do Amazon Redshift e PostgreSQL
- Recursos que são implementados de forma diferente
- Recursos incompatíveis do PostgreSQL
- Tipos de dados do PostgreSQL não compatíveis
- Funções incompatíveis do PostgreSQL

O Amazon Redshift é baseado no PostgreSQL. O Amazon Redshift e o PostgreSQL têm uma série de diferenças muito importantes que você deve conhecer ao projetar e desenvolver suas aplicações de data warehouse.

O Amazon Redshift foi projetado especificamente para aplicações de processamento analítico online (OLAP) e business intelligence (BI), que exigem consultas complexas em grandes conjuntos de dados. Por atender a requisitos muito diferentes, o esquema de armazenamento de dados especializado e o mecanismo de execução de consultas que o Amazon Redshift usa são completamente diferentes da implementação do PostgreSQL. Por exemplo, onde aplicações de processamento de transações online (OLTP) normalmente armazenam dados em linhas, o Amazon Redshift armazena dados em colunas, usando codificações especializadas de compactação de dados para uso de memória e E/S de disco otimizadas. Alguns recursos do PostgreSQL que são adequados para processamento OLTP em menor escala, tais como Índices secundários e operações eficientes de manipulação de dados de linha única, foram omitidos para melhorar a performance.

Consulte Arquitetura do Amazon Redshift para uma explicação detalhada da arquitetura do sistema de data warehouse do Amazon Redshift.

PostgreSQL 9.x inclui alguns recursos que não são suportados no Amazon Redshift. Além disso, existem diferenças importantes entre o Amazon Redshift SQL e o PostgreSQL que você deve conhecer. Esta seção destaca as diferenças entre o Amazon Redshift e o PostgreSQL e fornece orientação para o desenvolvimento de um data warehouse que aproveita ao máximo a implementação do Amazon Redshift SQL.

JDBC e ODBC do Amazon Redshift e PostgreSQL

Como o Amazon Redshift é baseado no PostgreSQL, recomendamos anteriormente o uso da versão 8.4.703 do driver JDBC4 do Postgresql e da versão 9.x dos drivers psqlODBC. Se você estiver usando esses drivers atualmente, recomendamos mudar para os novos drivers específicos do Amazon Redshift do Amazon Redshift. Para obter mais informações sobre drivers e configuração de conexões, consulte “Drivers JDBC, Pyhton e ODBC para o Amazon Redshift” no Guia de gerenciamento de clusters do Amazon Redshift.

Para evitar erros de falta de memória por parte do cliente ao recuperar grandes conjuntos de dados usando JDBC, você pode habilitar seu cliente para obter dados em lotes definindo o parâmetro JDBC para o tamanho da busca. Para obter mais informações, consulte Como configurar o parâmetro JDBC para o tamanho da busca.

O Amazon Redshift não reconhece o parâmetro JDBC maxRows. Em vez disso, especifique uma cláusula LIMIT para restringir o conjunto de resultados. Você também pode usar uma cláusula OFFSET para ir direto para um ponto inicial específico no conjunto de resultados.

Recursos que são implementados de forma diferente

Muitos elementos da linguagem Amazon Redshift SQL têm características de performance diferentes e usam sintaxe e semântica e que são bastante diferentes da implementação PostgreSQL equivalente.

> **Important**
>
> Não presuma que a semântica dos elementos que o Amazon Redshift e o PostgreSQL têm em comum são idênticos. Certifique-se de consultar o Guia do desenvolvedor do Amazon Redshift Comandos SQL para compreender as diferenças frequentemente sutis.

Um exemplo em particular é o comando VACUUM, que é usado para limpar e reorganizar tabelas. VACUUM funciona de forma diferente e usa um conjunto diferente de parâmetros da versão do PostgreSQL. Consulte Vacuum de tabelas para obter mais informações sobre o uso de VACUUM no Amazon Redshift.

Com frequência, as ferramentas e recursos de administração e gerenciamento de banco de dados também são diferentes. Por exemplo, o Amazon Redshift mantém um conjunto de tabelas e visualizações do sistema que fornecem informações sobre como o sistema está funcionando. Consulte Visualizações de monitoramento de SYS para obter mais informações.

A lista a seguir inclui alguns exemplos de recursos SQL que são implementados de forma diferente no Amazon Redshift.

- CRIAR TABELA

  O Amazon Redshift não oferece suporte a espaços de tabela, particionamento de tabela, herança e certas restrições. Com a implementação de CREATE TABLE do Amazon Redshift, é possível definir os algoritmos de classificação e distribuição das tabelas para otimizar o processamento paralelo.

  O Amazon Redshift Spectrum oferece suporte ao particionamento de tabela usando o comando CREATE EXTERNAL TABLE.

- ALTER TABLE

  Somente um subconjunto de ações ALTER COLUMN é suportado.

ADD COLUMN oferece suporte à inclusão de somente uma coluna em cada instrução ALTER TABLE.

- COPY

  O comando COPY do Amazon Redshift é altamente especializado para permitir o carregamento de dados de buckets do Amazon S3 e tabelas do Amazon DynamoDB e para facilitar a compactação automática. Consulte a seção Carregar dados no Amazon Redshift e a referência do comando COPY para obter detalhes.

- VACUUM

  Os parâmetros para VACUUM são totalmente diferentes. Por exemplo, a operação VACUUM padrão no PostgreSQL simplesmente recupera espaço e o torna disponível para reutilização; no entanto, a operação VACUUM padrão no Amazon Redshift é VACUUM FULL, que recupera espaço em disco e recorre a todas as linhas.

- Os espaços de rastreamento em valores VARCHAR são ignorados quando os valores de strings são comparados. Para obter mais informações, consulte Significância de espaços em branco.

Recursos incompatíveis do PostgreSQL

Esses recursos do PostgreSQL são incompatívesi com o Amazon Redshift.

> **Important**
>
> Não presuma que a semântica dos elementos que o Amazon Redshift e o PostgreSQL têm em comum são idênticos. Certifique-se de consultar o Guia do desenvolvedor do Amazon Redshift Comandos SQL para compreender as diferenças frequentemente sutis.

- A ferramenta de consulta psql não é compatível. O cliente Amazon Redshift RSQL é compatível.
- Particionamento de tabela (intervalo e particionamento da lista)
- Tablespaces
- Restrições
  - Exclusivo
  - Chave externa
  - Chave primária

  - Restrições de verificação
  - Restrições de exclusão

  Restrições exclusivas de chave primária e chave estrangeira são permitidas, mas elas são apenas informativas. Elas não são impostas pelo sistema, mas são utilizadas pelo planejador de consulta.
- Herança
- Colunas de sistema PostgreSQL

  O SQL do Amazon Redshift não define colunas do sistema implicitamente. Contudo, os seguintes nomes de coluna de sistema do PostgreSQL não podem ser usados como nomes de colunas definidas pelo usuário: `oid`, `tableoid`, `xmin`, `cmin`, `xmax`, `cmax` e `ctid` Para obter mais informações, consulte https://www.postgresql.org/docs/8.0/static/ddl-system-columns.html.
- Índices
- Cláusula NULLS em funções de janela
- Agrupamentos

  O Amazon Redshift não oferece suporte a sequências de intercalação específicas de localidade ou definidas pelo usuário. Consulte Sequências de colação.
- Expressões de valor
  - Expressões subscritas
  - Construtores de matriz
  - Construtores de linha
- Acionadores
- Gerenciamento de dados externos (SQL/MED)
- Funções da tabela
- Lista de VALUES usada como tabelas constantes
- Sequências
- Pesquisa de texto completo
- As permissões RULE e TRIGGER.

  O Amazon Redshift concede ou revoga essas permissões quando você executa GRANT ALL ou REVOKE ALL, mas a presença ou ausência das permissões RULE e TRIGGER não afeta de forma alguma as permissões de acesso do beneficiário.

Tipos de dados do PostgreSQL não compatíveis

Geralmente, se uma consulta tenta usar um tipo de dado incompatível, incluindo conversões explícitas ou implícitas, ela retornará um erro. Contudo, algumas consultas usando tipos de dados incompatíveis serão executadas no nó de liderança, mas não nos nós de computação. Consulte Funções SQL compatíveis no nó de liderança.

Para obter uma lista dos tipos de dados compatíveis, consulte Tipos de dados.

Esses tipos de dados PostgreSQL são incompatíveis com o Amazon Redshift.

- Matrizes
- BIT, BIT VARYING
- BYTEA
- Tipos de compostos
- Tipos enumerados
- Tipos geométricos (a implementação de tipos geométricos do Amazon Redshift e do PostgreSQL é diferente)
- HSTORE
- JSON
- Tipos de endereço de rede
- Tipos numéricos
  - SERIAL, BIGSERIAL, SMALLSERIAL
  - MONEY
- Tipos de identificador de objeto
- Pseudotipos
- Tipos de intervalo
- Tipos de caracteres especiais
  - "char" – Um tipo interno de byte único (onde o tipo de dados denominado char é colocado entre aspas).
  - name – Um tipo interno para nomes de objetos.

  Para obter mais informações sobre esses tipos, consulte Tipos de caracteres especiais na documentação do PostgreSQL.

- Tipos de pesquisa de texto
- TXID_SNAPSHOT
- UUID
- XML

Funções incompatíveis do PostgreSQL

Muitas funções que não são excluídas têm semântica ou uso diferentes. Por exemplo, algumas funções compatíveis serão executadas somente no nó de liderança. Além disso, algumas funções incompatíveis não retornarão um erro quando executadas no nó de liderança. O fato de essas funções não retornarem um erro em alguns casos não deve ser considerado como uma indicação de que a função é compatível com o Amazon Redshift.

> **Important**
>
> Não presuma que a semântica dos elementos que o Amazon Redshift e o PostgreSQL têm em comum são idênticos. Certifique-se de consultar o Comandos SQL do Guia do desenvolvedor de banco de dados do Amazon Redshift para compreender as diferenças frequentemente sutis.

Para obter mais informações, consulte Funções SQL compatíveis no nó de liderança.

Essas funções PostgreSQL são incompatíveis com o Amazon Redshift.

- Acesso a funções de consulta de privilégio
- Funções de advisory lock
- Funções agregadas
  - STRING_AGG()
  - ARRAY_AGG()
  - EVERY()
  - XML_AGG()
  - CORR()
  - COVAR_POP()
  - COVAR_SAMP()

  - REGR_AVGX(), REGR_AVGY()
  - REGR_COUNT()
  - REGR_INTERCEPT()
  - REGR_R2()
  - REGR_SLOPE()
  - REGR_SXX(), REGR_SXY(), REGR_SYY()
- Funções e operadores da matriz
- Funções de controle de backup
- Funções de informações de comentário
- Funções de localização de objetos do banco de dados
- Funções do tamanho de objetos do banco de dados
- Funções e operadores de data e hora
  - CLOCK_TIMESTAMP()
  - JUSTIFY_DAYS(), JUSTIFY_HOURS(), JUSTIFY_INTERVAL()
  - PG_SLEEP()
  - TRANSACTION_TIMESTAMP()
- Funções de suporte ENUM
- Funções e operadores geométricos
- Funções de acesso de arquivos genéricas
- IS DISTINCT FROM
- Funções e operadores de endereço de rede
- Funções matemáticas
  - DIV()
  - SETSEED()
  - WIDTH_BUCKET()
- Funções de retorno de conjuntos
  - GENERATE_SERIES()
  - GENERATE_SUBSCRIPTS()
- Funções e operadores de intervalo
- Funções de controle de recuperação

- Funções de informações de recuperação
- Função ROLLBACK TO SAVEPOINT
- Funções de consulta de visibilidade do schema
- Funções de sinalização de servidor
- Funções de sincronização de snapshot
- Funções de manipulação de sequências
- Funções de string
  - BIT_LENGTH()
  - OVERLAY()
  - CONVERT(), CONVERT_FROM(), CONVERT_TO()
  - ENCODE()
  - FORMAT()
  - QUOTE_NULLABLE()
  - REGEXP_MATCHES()
  - REGEXP_SPLIT_TO_ARRAY()
  - REGEXP_SPLIT_TO_TABLE()
- Funções de informações de catálogo do sistema
- Funções de informação do sistema
  - CURRENT_CATALOG CURRENT_QUERY()
  - INET_CLIENT_ADDR()
  - INET_CLIENT_PORT()
  - INET_SERVER_ADDR() INET_SERVER_PORT()
  - PG_CONF_LOAD_TIME()
  - PG_IS_OTHER_TEMP_SCHEMA()
  - PG_LISTENING_CHANNELS()
  - PG_MY_TEMP_SCHEMA()
  - PG_POSTMASTER_START_TIME()
  - PG_TRIGGER_DEPTH()
  - SHOW VERSION()
- Funções e operadores de pesquisa de texto

- IDs de transação e funções de snapshots
- Funções de trigger
- Funções XML

Uso de SQL

Tópicos

- Convenções de referência do SQL
- Elementos básicos
- Expressões
- Condições

A linguagem SQL consiste em comandos e funções que você usa para trabalhar com bancos de dados e objetos de banco de dados. A linguagem também impõe regras sobre o uso de tipos de dados, expressões e literais.

Convenções de referência do SQL

Esta seção explica as convenções que são usadas para gravar a sintaxe para as expressões, comandos e funções SQL descritas na seção de referência de SQL.

Caractere	Descrição
CAPS	Palavras em letras maiúsculas são palavras chave.
[ ]	Parênteses denotam argumentos opcionais. Vários argumentos em parênteses indicam que você pode escolher qualquer número de argumento s. Além disso, os argumentos entre colchetes em linhas separadas indicam que o analisador Amazon Redshift espera que os argumentos estejam na ordem em que estão listados na sintaxe. Para ver um exemplo, consulte SELECT.
{ }	As chaves indicam que será necessário escolher um dos argumentos nelas.
|	Barras verticais indicam que você pode escolher entre os argumentos.

Caractere	Descrição
itálico	As palavras em itálico indicam os espaços reservados. Você deve inserir o valor apropriado no lugar da palavra em itálico.
. . .	Uma elipse indica que você pode repetir o elemento anterior.
'	Palavras entre aspas simples indicam que você deve digitar as aspas.

Elementos básicos

Tópicos

- Nomes e identificadores
- Literais
- Nulos
- Tipos de dados
- Sequências de colação

Esta seção aborda as regras para trabalhar com nomes de objetos do banco de dados, literais, nulos e tipos de dados.

Nomes e identificadores

Os nomes identificam objetos do banco de dados, incluindo tabelas e colunas, assim como usuários e senhas. Os termos nome e identificador podem ser usados de forma intercambiável. Há dois tipos de identificadores, identificadores padrão e identificadores citados ou delimitados. Os identificadores devem consistir somente em caracteres UTF-8 imprimíveis. As letras ASCII em identificadores padrão e delimitados não fazem distinção entre maiúsculas e minúsculas e são convertidas em minúsculas no banco de dados. Nos resultados da consulta, os nomes de coluna são retornados, por padrão, em minúsculas. Para retornar os nomes de coluna em maiúsculas, defina o parâmetro de configuração describe_field_name_in_uppercase como **`true`**.

Identificadores padrão

Os identificadores SQL padrão aderem a um conjunto de regras e devem:

- Começar com um caractere ASCII alfabético de único byte ou caractere sublinhado ou com um caractere UTF-8 multibyte com dois a quatro bytes de extensão.
- Os caracteres subsequentes podem ser caracteres ASCII alfanuméricos, sublinhados ou com cifrão ou caracteres UTF-8 multibyte com dois a quatro bytes de extensão.
- Ter entre 1 e 127 bytes de comprimento, sem incluir aspas para identificadores delimitados.
- Não conter qualquer aspa ou espaço.
- Não ser uma palavra chave SQL reservada.

Identificadores delimitados

Identificadores delimitados (também conhecidos como identificadores citados) começam e terminam com aspas ("). Se você usar um identificador delimitado, deverá usar aspas duplas para todas as referências para aquele objeto. O identificador pode conter qualquer caractere padrão UTF-8 imprimível, exceto as próprias aspas duplas. Portanto, você pode criar os nomes de coluna ou tabela incluindo caracteres geralmente ilegais, tais como espaços ou o símbolo de por cento.

As letras ASCII em identificadores delimitados não fazem distinção entre maiúsculas e minúsculas e são convertidas em minúsculas. Para usar aspas duplas em uma string, você deve precedê-la com outro caractere de aspas duplas.

Identificadores que diferenciam maiúsculas e minúsculas

Identificadores que diferenciam maiúsculas e minúsculas (também conhecidos como identificadores de maiúsculas e minúsculas) podem conter letras maiúsculas e minúsculas. Para usar identificadores que diferenciam maiúsculas e minúsculas, você pode definir a configuração `enable_case_sensitive_identifier` como `true`. Você pode definir essa configuração para o cluster ou para uma sessão. Para obter mais informações, consulte “Default parameter values” (Valores de parâmetro comuns) no Guia de gerenciamento de clusters do Amazon Redshift e enable_case_sensitive_identifier.

Nomes de coluna de sistema

Os nomes de coluna de sistema do PostgreSQL a seguir não podem ser usados como nomes de colunas definidas pelo usuário. Para obter mais informações, consulte https://www.postgresql.org/docs/8.0/static/ddl-system-columns.html.

- `oid`
- `tableoid`

- xmin
- cmin
- xmax
- cmax
- ctid

Exemplos

Esta tabela mostra exemplos de identificadores delimitados, a saída resultante e uma discussão:

Sintaxe	Resultado	Discussão
"grupo"	grupo	GRUPO é uma palavra reservada, portanto, seu uso em um identificador requer aspas duplas.
"""WHERE"""	"where"	WHERE também é uma palavra reservada. Para incluir aspas na string, faça o escape de cada caractere de aspas duplas com caracteres de aspas duplas adicionais.
“Este nome”	este nome	As aspas duplas são necessárias para preservar o espaço.
"É ""ISSO"""	é "isso"	As aspas em torno de IS IT devem ser precedidas por uma aspa extra para fazer parte do nome.

Para criar uma tabela chamada grupo com uma coluna chamada “isso”:

```
create table "group" (
"This ""IS IT""" char(10));
```

As seguintes consultas retornam o mesmo resultado:

```
select "This ""IS IT"""
from "group";

this "is it"
--------------
```

```
(0 rows)
```

```
select "this ""is it"""
from "group";

this "is it"
--------------
(0 rows)
```

A seguinte sintaxe `table.column` totalmente qualificada também retorna o mesmo resultado:

```
select "group"."this ""is it"""
from "group";

this "is it"
--------------
(0 rows)
```

O comando CREATE TABLE a seguir cria uma tabela com uma barra em um nome de coluna:

```
create table if not exists city_slash_id(
                  "city/id" integer not null,
                  state char(2) not null);
```

Literais

Um literal ou constante é um valor fixo de dados, composto de uma sequência de caracteres ou uma constante numérica. O Amazon Redshift oferece suporte a vários tipos de literais, incluindo:

- Literais numéricos para números inteiros, decimais e números de ponto flutuante. Para obter mais informações, consulte Literais de número inteiro e de ponto flutuante.
- Literais do caractere, também chamados de strings, strings de caracteres ou constantes de caracteres
- Literais de data e hora e de intervalo, usados com tipos de dados de data e hora. Para obter mais informações, consulte Literais de data, hora e timestamp e Tipos de dados e literais de intervalo.

Nulos

Se uma coluna em uma linha estiver ausente, for desconhecida ou não aplicável, ela será um valor nulo ou considerada como contendo nulo. Nulos podem aparecer em campos de qualquer tipo de dados que não estejam restritos por uma chave primária ou por restrições NOT NULL. Um nulo não é equivalente ao valor zero ou a uma string vazia.

Qualquer expressão aritmética que contenha um null é sempre avaliada como um null. Todos os operadores exibem um nulo quando recebem um argumento ou operando nulo.

Para testar nulos, use as condições de comparação IS NULL e IS NOT NULL. Como nulo representa uma ausência de dados, um nulo não é igual ou desigual a qualquer valor ou a outro nulo.

Tipos de dados

Tópicos

- Caracteres multibyte
- Tipos numéricos
- Tipos de caracteres
- Tipos de datetime
- Tipo booliano
- Tipo HLLSKETCH
- Tipo SUPER
- Tipo VARBYTE
- Compatibilidade e conversão dos tipos

Cada valor que o Amazon Redshift armazena ou recupera tem um tipo de dados com um conjunto fixo de propriedades associadas. Os tipos de dados são declarados quando as tabelas são criadas. Um tipo de dado restringe um conjunto de valores que uma coluna ou argumento pode conter.

A tabela a seguir lista os tipos de dados que você pode usar nas tabelas do Amazon Redshift.

Tipo de dados	Aliases	Descrição
SMALLINT	INT2	Número inteiro de dois bytes assinado

Tipo de dados	Aliases	Descrição
INTEGER	INT, INT4	Número inteiro de quatro bytes assinado
BIGINT	INT8	Número inteiro de oito bytes assinado
DECIMAL	NUMERIC	Numérico exato com precisão selecionável
REAL	FLOAT4	Número de ponto flutuante de precisão simples
DOUBLE PRECISION	FLOAT8, FLOAT	Número de ponto flutuante de precisão dupla
CHAR	CHARACTER, NCHAR, BPCHAR	String de caracteres com comprimento fixo
VARCHAR	CHARACTER VARYING, NVARCHAR, TEXT	String de caracteres de comprimento variável com limite definido pelo usuário
DATE		Data de calendário (ano, mês, dia)
TIME	TIME WITHOUT TIME ZONE	Hora do dia
TIMETZ	TIME WITH TIME ZONE	Hora do dia com fuso horário
TIMESTAMP	TIMESTAMP WITHOUT TIME ZONE	Data e hora (sem fuso horário)
TIMESTAMPTZ	TIMESTAMP WITH TIME ZONE	Data e hora (com fuso horário)
INTERVALO ENTRE UM ANO E UM MÊS		Tempo decorrido na ordem de ano para mês

Tipo de dados	Aliases	Descrição
INTERVALO ENTRE UM DIA E UM SEGUNDO		Tempo decorrido na ordem de dia para segundo
BOOLEAN	BOOL	Booleanos lógicos (verdadeiro/falso)
HLLSKETCH		Tipo usado com esboços do HyperLogLog.
SUPER		Um tipo de superconjunto de dados que abrange todos os tipos escalares do Amazon Redshift, inclusive tipos complexos, como ARRAY e STRUCTS.
VARBYTE	VARBINARY, BINARY VARYING	Valor binário de comprimento variável
GEOMETRY		Dados espaciais
GEOGRAPHY		Dados espaciais

> **Note**
>
> Para obter informações sobre tipos de dados incompatíveis, como "char" (observe que char está entre aspas), consulte Tipos de dados do PostgreSQL não compatíveis.

Caracteres multibyte

O tipo de dados VARCHAR é compatível com caracteres UTF-8 multibyte até um máximo de quatro bytes. Caracteres de cinco ou mais bytes são incompatíveis. Para calcular o tamanho de uma coluna VARCHAR que contenha caracteres multibyte, multiplique o número de caracteres pelo número de bytes por caractere. Por exemplo, se uma string contém quatro caracteres chineses e cada caractere

possui três bytes de comprimento, você precisará de uma coluna VARCHAR(12) para armazenar a string.

O VARCHAR não é compatível com os seguintes pontos de código de caracteres UTF-8 inválidos:

`0xD800 – 0xDFFF` (Sequências de byte: `ED A0 80` – `ED BF BF`)

O tipo de dados CHAR não é compatível com caracteres multibyte.

Tipos numéricos

Tópicos

- Tipos de inteiros
- Tipo DECIMAL ou NUMERIC
- Observações sobre o uso de colunas do tipo DECIMAL ou NUMERIC de 128 bits
- Tipos de ponto flutuante
- Computações com valores numéricos
- Literais de número inteiro e de ponto flutuante
- Exemplos com tipos numéricos

Os tipos de dados numéricos incluem números inteiros, decimais e números de ponto flutuante.

Tipos de inteiros

Use tipos de dados SMALLINT, INTEGER e BIGINT para armazenar números inteiros de vários intervalos. Você não pode armazenar valores fora do intervalo permitido para cada tipo.

Nome	Armazenamento	Intervalo
SMALLINT ou INT2	2 bytes	-32768 a +32767
INTEGER, INT, ou INT4	4 bytes	-2147483648 a +2147483647
BIGINT ou INT8	8 bytes	-9223372036854775808 a 9223372036854775807

Tipo DECIMAL ou NUMERIC

Use o tipo de dados DECIMAL ou NUMERIC para armazenar valores com uma precisão definida pelo usuário. As palavras-chave DECIMAL e NUMERIC são intercambiáveis. Neste documento, decimal é o termo preferido para esse tipo de dados. O termo numeric é usado genericamente para se referir aos tipos de dados de número inteiro, decimal e de ponto flutuante.

Armazenamento	Intervalo
Variável, até 128 bits para tipos DECIMAL não compactados.	Números inteiros assinados de 128 bits com até 38 dígitos de precisão.

Define uma coluna DECIMAL em uma tabela especificando uma precisão e escala:

```
decimal(precision, scale)
```

precisão

O número total de dígitos significativos no valor inteiro: o número de dígitos em ambos os lados do ponto decimal. Por exemplo, o número `48.2891` tem precisão de 6 e uma escala de 4. A precisão padrão, quando não especificada, é 18. A precisão máxima é 38.

Se o número de dígitos à esquerda do ponto decimal em um valor de entrada excede a precisão de coluna menos a sua escala, o valor não pode ser copiado para a coluna (ou inserido ou atualizado). Esta regra aplica-se a qualquer valor que caia fora do intervalo de definição da coluna. Por exemplo, o intervalo permitido de valores para uma coluna `numeric(5,2)` é `-999.99` a `999.99`.

scale

O número de dígitos decimais na parte fracionada do valor, à direita do ponto decimal. Números inteiros têm uma escala de zero. Em uma especificação de coluna, o valor de escala deve ser menor ou igual ao valor de precisão. A escala padrão, quando não especificada, é 0. A escala máxima é 37.

Se a escala de um valor de entrada carregado em uma tabela for maior do que a escala da coluna, o valor será arredondado para a escala especificada. Por exemplo, a coluna PRICEPAID na tabela SALES é uma coluna DECIMAL(8,2). Se um valor DECIMAL(8,4) é inserido na coluna PRICEPAID, o valor é arredondado para uma escala de 2.

```
insert into sales
values (0, 8, 1, 1, 2000, 14, 5, 4323.8951, 11.00, null);

select pricepaid, salesid from sales where salesid=0;

pricepaid | salesid
-----------+---------
4323.90 |       0
(1 row)
```

Contudo, os resultados de conversões explícitas dos valores selecionados a partir de tabelas não são arredondados.

> **Note**
>
> O valor positivo máximo que você pode inserir na coluna DECIMAL(19,0) é `9223372036854775807` (2^{63} -1). O valor negativo máximo é `-9223372036854775808`. Por exemplo, uma tentativa de inserir o valor `9999999999999999999` (19 noves) causará um erro de transbordamento. Independentemente do posicionamento do ponto decimal, a maior string que o Amazon Redshift pode representar como um número DECIMAL é `9223372036854775807`. Por exemplo, o maior valor que você pode carregar em uma coluna DECIMAL(19,18) é `9.223372036854775807`.
> Essas regras ocorrem porque valores DECIMAL com 19 ou menos dígitos significativos de precisão são armazenados internamente como valores inteiros de 8 bytes, enquanto valores DECIMAL com 20 a 38 dígitos significativos de precisão são armazenados como valores inteiros de 16 bytes.

Observações sobre o uso de colunas do tipo DECIMAL ou NUMERIC de 128 bits

Não designe arbitrariamente a precisão máxima às colunas do tipo DECIMAL, a não ser que você esteja certo de que sua aplicação requer essa precisão. Valores de 128 bits usam duas vezes mais espaço em disco do que valores de 64 bits e podem retardar o tempo de execução da consulta.

Tipos de ponto flutuante

Use os tipos de dados REAL e DOUBLE PRECISION para armazenar valores numéricos com precisão variável. Esses tipos são inexatos, o que significa que alguns valores são armazenados

como aproximações, de tal forma que o armazenamento e retorno de um valor específico pode resultar em ligeiras discrepâncias. Se você precisar de armazenamento e cálculos precisos (como para quantidades monetárias), use o tipo de dados DECIMAL.

REAL representa o formato de ponto flutuante de precisão única, de acordo com o padrão 754 do IEEE para aritmética de ponto flutuante binário. Tem uma precisão de cerca de 6 dígitos e um intervalo de cerca de 1E-37 a 1E+37. Você também pode especificar esse tipo de dado como FLOAT4.

DOUBLE PRECISION representa o formato de ponto flutuante de precisão dupla, de acordo com o padrão 754 do IEEE para aritmética de ponto flutuante binário. Tem uma precisão de cerca de 15 dígitos e um intervalo de cerca de 1E-307 a 1E+308. Você também pode especificar esse tipo de dado como FLOAT ou FLOAT8.

Além dos valores numéricos comuns, os tipos de ponto flutuante têm diversos valores especiais. Use aspas simples em torno desses valores ao usá-los em SQL:

- `NaN`: not-a-number
- `Infinity`: infinity
- `-Infinity`: negative infinity

Por exemplo, para inserir not-a-number na coluna `day_charge` da tabela `customer_activity`, execute o seguinte SQL:

```
insert into customer_activity(day_charge) values('NaN');
```

Computações com valores numéricos

Neste contexto, computação refere-se à operações matemáticas binárias: adição, subtração, multiplicação e divisão. Esta seção descreve os tipos de retorno previstos para essas operações, assim como a fórmula específica que é aplicada para determinar a precisão e escala quando dados do tipo DECIMAL estão envolvidos.

Quando os valores numéricos são computados durante o processamento da consulta, você pode encontrar casos onde o cálculo é impossível e a consulta retorna um erro de transbordamento numérico. Você também pode encontrar casos em que a escala de valores computados varia ou é inesperada. Para algumas operações, você pode usar conversão explícita (promoção de tipo) ou parâmetros de configuração do Amazon Redshift para solucionar esses problemas.

Para obter informações sobre os resultados de computações semelhantes com funções SQL, consulte Funções agregadas.

Tipos de retorno para computações

Dado o conjunto de tipos de dados numéricos suportados no Amazon Redshift, a tabela a seguir mostra os tipos de retorno esperados para operações de adição, subtração, multiplicação e divisão. A primeira coluna do lado esquerdo da tabela representa o primeiro operando no cálculo e a linha superior representa o segundo operando.

Operando 1	Operando 2	Tipo de retorno
INT2	INT2	INT2
INT2	INT4	INT4
INT2	INT8	INT8
INT2	DECIMAL	DECIMAL
INT2	FLOAT4	FLOAT8
INT2	FLOAT8	FLOAT8
INT4	INT4	INT4
INT4	INT8	INT8
INT4	DECIMAL	DECIMAL
INT4	FLOAT4	FLOAT8
INT4	FLOAT8	FLOAT8
INT8	INT8	INT8
INT8	DECIMAL	DECIMAL
INT8	FLOAT4	FLOAT8
INT8	FLOAT8	FLOAT8

Operando 1	Operando 2	Tipo de retorno
DECIMAL	DECIMAL	DECIMAL
DECIMAL	FLOAT4	FLOAT8
DECIMAL	FLOAT8	FLOAT8
FLOAT4	FLOAT8	FLOAT8
FLOAT8	FLOAT8	FLOAT8

Precisão e escala de resultados de DECIMAL computados

A tabela a seguir resume as regras para precisão e escala resultantes de computação quando operações matemáticas retornam resultados DECIMAIS. Nessa tabela, `p1` e `s1` representam a precisão e escala do primeiro operando no cálculo e `p2` e `s2` representam a precisão e escala de segundo operando. (Independentemente desses cálculos, a precisão máxima de resultado é 38 e a escala máxima de resultado é 38.)

Operation	Precisão e escala de resultados
+ ou -	Escalar = `max(s1,s2)` Precisão = `max(p1-s1,p2-s2)+1+scale`
*	Escalar = `s1+s2` Precisão = `p1+p2+1`
/	Escalar = `max(4,s1+p2-s2+1)` Precisão = `p1-s1+ s2+scale`

Por exemplo, as colunas PRICEPAID e COMMISSION na tabela SALES são ambas colunas do tipo DECIMAL(8,2). Se você dividir PRICEPAID pela COMMISSION (ou vice-versa), a fórmula será aplicada da seguinte forma:

```
Precision = 8-2 + 2 + max(4,2+8-2+1)
```

```
= 6 + 2 + 9 = 17

Scale = max(4,2+8-2+1) = 9

Result = DECIMAL(17,9)
```

O seguinte cálculo é a regra geral para computação da precisão e escala resultantes para operações executadas em valores DECIMAIS com operadores de conjunto tais como UNION, INTERSECT e EXCEPT ou funções como COALESCE e DECODE:

```
Scale = max(s1,s2)
Precision = min(max(p1-s1,p2-s2)+scale,19)
```

Por exemplo, uma tabela DEC1 com uma colina DECIMAL(7,2) é unida a uma tabela DEC2 com uma coluna DECIMAL(15,3) para criar uma tabela DEC3. O esquema de DEC3 mostra que a coluna se torna uma coluna NUMERIC(15,3).

```
create table dec3 as select * from dec1 union select * from dec2;
```

Resultado

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'dec3';

column |     type      | encoding | distkey | sortkey
-------+---------------+----------+---------+---------
c1     | numeric(15,3) | none     | f       | 0
```

No exemplo acima, a fórmula é aplicada da seguinte forma:

```
Precision = min(max(7-2,15-3) + max(2,3), 19)
= 12 + 3 = 15

Scale = max(2,3) = 3

Result = DECIMAL(15,3)
```

Observações sobre operações de divisão

Para operações de divisão, condições de dividir por zero retornam erros.

O limite de escala de 100 é aplicado após o cálculo da precisão e escala. Se a escala de resultados calculada for maior que 100, os resultados da divisão serão escalados da seguinte forma:

- Precisão = `precision - (scale - max_scale)`
- Escalar = `max_scale`

Se a precisão calculada for maior do que a precisão máxima (38), a precisão será reduzida para 38 e a escala será o resultado de: `max((38 + scale - precision), min(4, 100))`

Condições de transbordamento

O transbordamento é verificado para todas as computações numéricas. Dados DECIMAIS com uma precisão de 19 ou o menos são armazenados como números inteiros de 64 bits. Dados DECIMAIS com uma precisão maior que 19 são armazenados como números inteiros de 128 bits. A precisão máxima para todos os valores DECIMAIS é 38 e a escala máxima é 37. Erros de transbordamento ocorrem quando um valor excede esses limites, que se aplicam a conjuntos de resultados intermediário e final:

- A conversão explícita resulta em erros de transbordamento de tempo de execução quando valores de dados específicos não se enquadram na precisão ou escala necessárias especificadas pela função de conversão. Por exemplo, você não pode converter todos os valores da coluna PRICEPAID na tabela SALES (uma coluna DECIMAL(8,2)) e retornar um resultado DECIMAL(7,3):

  ```
  select pricepaid::decimal(7,3) from sales;
  ERROR:  Numeric data overflow (result precision)
  ```

  Este erro ocorre porque alguns dos valores maiores na coluna PRICEPAID não podem ser convertidos.
- As operações de multiplicação produzem resultados em que a escala de resultados é a soma de escala de cada operando. Se ambos os operandos têm uma escala de 4, por exemplo, a escala de resultados é 8, deixando apenas 10 dígitos para o lado esquerdo do ponto decimal. Portanto, é relativamente fácil se deparar com condições de transbordamento ao multiplicar dois números grandes que possuem uma escala significativa.

  O exemplo a seguir retorna um erro de transbordamento.

  ```
  SELECT CAST(1 AS DECIMAL(38, 20)) * CAST(10 AS DECIMAL(38, 20));
  ERROR: 128 bit numeric data overflow (multiplication)
  ```

Você pode contornar o erro de transbordamento usando divisão em vez de multiplicação. Use o exemplo a seguir para dividir por 1 dividido pelo divisor original.

```
SELECT CAST(1 AS DECIMAL(38, 20)) / (1 / CAST(10 AS DECIMAL(38, 20)));
+----------+
| ?column? |
+----------+
| 10       |
+----------+
```

Cálculos numéricos com os tipos INTEGER e DECIMAL

Quando um dos operandos em um cálculo tem um tipo de dado INTEGER e o outro operando é DECIMAL, o operando INTEGER é implicitamente convertido como um DECIMAL:

- INT2 (SMALLINT) é convertido como DECIMAL(5,0)
- INT4 (INTEGER) é convertido como DECIMAL(10,0)
- INT8 (BIGINT) é convertido como DECIMAL(19,0)

Por exemplo, se você multiplicar SALES.COMMISSION, uma coluna DECIMAL(8,2), e SALES.QTYSOLD, uma coluna SMALLINT, este cálculo será convertido como:

```
DECIMAL(8,2) * DECIMAL(5,0)
```

Literais de número inteiro e de ponto flutuante

Literais ou constantes que representam números podem ser números inteiros ou de ponto flutuante.

Literais de número inteiro

Uma constante de número inteiro é uma sequência de dígitos de 0-9, com um sinal positivo (+) ou negativo (-) opcional precedendo os dígitos.

Sintaxe

```
[ + | - ] digit ...
```

Exemplos

Números inteiros válidos incluem os seguintes:

```
23
-555
+17
```

Literais de ponto flutuante

Literais de ponto flutuante (também referidos como literais decimais, numéricos, ou fracionários) são sequências de dígitos que podem incluir um ponto decimal e o marcador de expoente opcional (e).

Sintaxe

```
[ + | - ] digit ... [ . ] [ digit ...]
[ e | E [ + | - ] digit ... ]
```

Argumentos

e | E

e ou E indica que o número é especificado em notação científica.

Exemplos

Literais de ponto flutuante válidos incluem o seguinte:

```
3.14159
-37.
2.0e19
-2E-19
```

Exemplos com tipos numéricos

Instrução CREATE TABLE

A seguinte instrução CREATE TABLE demonstra a declaração de diferentes tipos de dados numéricos:

```
create table film (
```

```
film_id integer,
language_id smallint,
original_language_id smallint,
rental_duration smallint default 3,
rental_rate numeric(4,2) default 4.99,
length smallint,
replacement_cost real default 25.00);
```

Tentativa de inserir um número inteiro que está fora do intervalo

O seguinte exemplo tenta inserir o valor 33000 em uma coluna SMALLINT.

```
insert into film(language_id) values(33000);
```

O intervalo para SMALLINT é de -32768 a +32767, portanto, o Amazon Redshift retorna um erro.

```
An error occurred when executing the SQL command:
insert into film(language_id) values(33000)

ERROR: smallint out of range [SQL State=22003]
```

Inserção de um valor decimal em uma coluna de número inteiro

O seguinte exemplo insere o valor decimal em uma coluna INT.

```
insert into film(language_id) values(1.5);
```

Este valor é inserido, mas arredondado para o valor inteiro de 2.

Inserção de um decimal que suceda porque sua escala é arredondada

O seguinte exemplo insere um valor decimal que possui precisão maior que a coluna.

```
insert into film(rental_rate) values(35.512);
```

Nesse caso, o valor `35.51` é inserido na coluna.

Tentativa de inserir um valor decimal que esteja fora do intervalo

Nesse caso, o valor `350.10` está fora de intervalo. O número de dígitos para valores em colunas DECIMAIS é igual à precisão da coluna menos sua escala (4 menos 2 na coluna RENTAL_RATE). Em outras palavras o intervalo permitido para uma coluna `DECIMAL(4,2)` é `-99.99` a `99.99`.

```
insert into film(rental_rate) values (350.10);
ERROR:  numeric field overflow
DETAIL:  The absolute value is greater than or equal to 10^2 for field with precision
 4, scale 2.
```

Inserção de valores de precisão variável em uma coluna REAL

O seguinte exemplo insere valores de precisão variável em uma coluna REAL.

```
insert into film(replacement_cost) values(1999999.99);

insert into film(replacement_cost) values(1999.99);

select replacement_cost from film;

+------------------+
| replacement_cost |
+------------------+
| 2000000          |
| 1999.99          |
+------------------+
```

O valor `1999999.99` é convertido em `2000000` para atender ao requisito de precisão da coluna `REAL`. O valor `1999.99` é carregado como está.

Tipos de caracteres

Tópicos

- Armazenamento e intervalos
- CHAR ou CHARACTER
- VARCHAR ou CHARACTER VARYING
- Tipos NCHAR e NVARCHAR
- Tipos TEXT e BPCHAR
- Significância de espaços em branco
- Exemplos com tipos de caracteres

Os tipos de dados de caracteres incluem CHAR (caractere) e VARCHAR (caractere variável).

Armazenamento e intervalos

Os tipos de dados CHAR e VARCHAR são definidos em termos de bytes, não caracteres. Uma coluna CHAR pode ter somente caracteres de byte único, portanto a coluna CHAR(10) pode conter uma string com um comprimento máximo de 10 bytes. Uma coluna VARCHAR pode conter caracteres de multibyte, até o máximo de quatro bytes por caractere. Por exemplo, uma coluna VARCHAR(12) pode conter 12 caracteres de único byte, 6 caracteres de dois bytes, 4 caracteres de três bytes ou 3 caracteres de quatro bytes.

Nome	Armazenamento	Intervalo (largura da coluna)
CHAR, CHARACTER ou NCHAR	Comprimento da string, incluindo espaços em branco (se houver)	4.096 bytes
VARCHAR, CHARACTER VARYING ou NVARCHAR	4 bytes + total de bytes dos caracteres, onde cada caractere pode ser de 1 a 4 bytes.	65.535 bytes (64K -1)
BPCHAR	Convertido para CHAR(256) de comprimento fixo	256 bytes
TEXT	Convertido para VARCHAR(256).	260 bytes

Note

A sintaxe de CREATE TABLE é compatível com a palavra chave MAX para os tipos de dados de caracteres. Por exemplo:

```
create table test(col1 varchar(max));
```

Para CHAR, MAX define a largura da coluna como 4.096 bytes.
Para VARCHAR, MAX define a largura da coluna como 65.535 bytes nas instruções CREATE TABLE. Para operações na memória, VARCHAR (MAX) permite até 16.000.000 bytes.

CHAR ou CHARACTER

Use uma coluna CHAR OU CHARACTER para armazenar strings de comprimento fixo. Essas strings são protegidas com espaços, portanto uma coluna CHAR(10) sempre ocupa 10 bytes de armazenamento.

```
char(10)
```

Uma coluna CHAR sem a especificação de comprimento resulta em uma coluna CHAR(1).

VARCHAR ou CHARACTER VARYING

Use uma coluna VARCHAR ou CHARACTER VARYING para armazenar strings de tamanho variável com um limite fixo. Essas strings não são protegidas com espaços, portanto uma coluna VARCHAR(120) consistem em um máximo de 120 caracteres de único byte, 60 caracteres de dois bytes, 40 caracteres de três bytes ou 30 caracteres de quatro bytes.

```
varchar(120)
```

Se você usar o tipo de dados VARCHAR sem um especificador de comprimento em uma instrução CREATE TABLE, o comprimento padrão será 256.

Funções de string agora comportam até 16 milhões de bytes. Por exemplo, a saída da função CONCAT era anteriormente limitada a 65.535 bytes, mas agora comporta até 16 milhões de bytes.

```
SELECT LEN(CONCAT(REPEAT('A', 5000000), REPEAT('B', 5000000))) AS total_length;

 total_length
--------------
     10000000
```

Tipos NCHAR e NVARCHAR

Você pode criar colunas com os tipos NCHAR e NVARCHAR (também conhecidos como tipos NATIONAL CHARACTER e NATIONAL CHARACTER VARYING). Esses tipos são convertidos para os tipos CHAR e VARCHAR, respectivamente, e armazenados com o número de bytes especificado.

Uma coluna NCHAR sem a especificação de comprimento é convertida em uma coluna CHAR(1).

Uma coluna NVARCHAR sem uma especificação de comprimento é convertida em uma coluna VARCHAR(256).

Tipos TEXT e BPCHAR

Você pode criar uma tabela Amazon Redshift com uma coluna TEXT, mas ela é convertida em uma coluna VARCHAR(256) que aceita valores de comprimento variável com no máximo 256 caracteres.

Você pode criar uma coluna Amazon Redshift com um tipo BPCHAR (caractere preenchido em branco), que o Amazon Redshift converte em uma coluna CHAR(256) de comprimento fixo.

Significância de espaços em branco

Tanto os tipos de dados CHAR quanto VARCHAR armazenam strings de até n bytes de comprimento. Uma tentativa de armazenar uma string mais longa em um destes tipos de coluna resulta em um erro, a menos que os caracteres adicionais sejam todos espaços (em branco); nesse caso, a string é truncada para o comprimento máximo. Se a string é mais curta do que o comprimento máximo, os valores CHAR são protegidos por espaços, mas valores CHAR armazenam a string sem os espaços.

Espaços em branco em valores CHAR são sempre semanticamente insignificantes. Eles são ignorados quando você compara dois valores CHAR, não são incluídos em cálculos de COMPRIMENTO e são removidos quando você converte um valor CHAR para outro tipo de string.

Os espaços em branco em valores VARCHAR e de CHAR são tratados como semanticamente insignificantes quando os valores são comparados.

Os cálculos de comprimento retornam o comprimento de strings de caracteres VARCHAR com espaços em branco incluídos no comprimento. Os espaços em branco não são contados no comprimento para strings de caracteres de comprimento fixo.

Exemplos com tipos de caracteres

Instrução CREATE TABLE

A seguinte instrução CREATE TABLE demonstra o uso dos tipos de dados VARCHAR e CHAR:

```
create table address(
address_id integer,
address1 varchar(100),
address2 varchar(50),
district varchar(20),
city_name char(20),
state char(2),
postal_code char(5)
);
```

Os exemplos a seguir usam esta tabela.

Espaços em branco em strings de caracteres de comprimento variável

Como ADDRESS1 é uma coluna VARCHAR, os espaços em branco no segundo endereço inserido são semanticamente insignificantes. Em outras palavras, estes dois endereços inseridos correspondem.

```
insert into address(address1) values('9516 Magnolia Boulevard');

insert into address(address1) values('9516 Magnolia Boulevard  ');
```

```
select count(*) from address
where address1='9516 Magnolia Boulevard';

count
-------
2
(1 row)
```

Se a coluna ADDRESS1 fosse uma coluna CHAR e os mesmos valores fossem inseridos, a consulta de CONTAGEM(*) reconheceria as strings de caracteres como as mesmas e retornaria 2.

Resultados da função LENGTH

A função de COMPRIMENTO reconhece espaços em branco em colunas VARCHAR:

```
select length(address1) from address;

length
--------
23
25
(2 rows)
```

Um valor de `Augusta` na coluna CITY_NAME, que é uma coluna CHAR, retornaria sempre um comprimento de 7 caracteres, independente de quaisquer espaços em branco na string de entrada.

Valores que excedem o comprimento da coluna

As strings de caracteres não são truncadas para se adequar à largura declarada da coluna:

```
insert into address(city_name) values('City of South San Francisco');
ERROR: value too long for type character(20)
```

Uma ação alternativa para esse problema é converter o valor para o tamanho da coluna:

```
insert into address(city_name)
values('City of South San Francisco'::char(20));
```

Neste caso, os primeiros 20 caracteres da string (`City of South San Fr`) seriam carregados na coluna.

Tipos de datetime

Tópicos

- Armazenamento e intervalos
- DATE
- TIME
- TIMETZ
- TIMESTAMP
- TIMESTAMPTZ
- Exemplos com tipos de datetime
- Literais de data, hora e timestamp
- Tipos de dados e literais de intervalo

Os tipos de dados de data e hora incluem DATE, TIME, TIMETZ, TIMESTAMP e TIMESTAMPTZ.

Armazenamento e intervalos

Nome	Armazenamento	Intervalo	Resolução
DATE	4 bytes	4713 AC a 294276 DC	1 dia
TIME	8 bytes	00:00:00 para 24:00:00	1 microssegundo
TIMETZ	8 bytes	00:00:00+1459 para 00:00:00+1459	1 microssegundo
TIMESTAMP	8 bytes	4713 AC a 294276 DC	1 microssegundo
TIMESTAMPTZ	8 bytes	4713 AC a 294276 DC	1 microssegundo

DATE

Use o tipo de dados DATE para armazenar datas de calendário simples sem timestamps.

TIME

TIME é um alias de TIME WITHOUT TIME ZONE.

Use o tipo de dados TIME para armazenar a hora do dia.

As colunas TIME armazenam valores com até um máximo de seis dígitos de precisão para segundos fracionários.

Por padrão, os valores de TIME são Tempo Universal Coordenado (UTC) nas tabelas do usuário e nas tabelas do sistema Amazon Redshift.

TIMETZ

TIMETZ é um alias de TIME WITH TIME ZONE.

Use o tipo de dados TIMETZ para armazenar a hora do dia com um fuso horário.

As colunas TIMETZ armazenam valores com até um máximo de seis dígitos de precisão para segundos fracionários.

Por padrão, os valores de TIMETZ são UTC nas tabelas de usuário e nas tabelas de sistema do Amazon Redshift.

TIMESTAMP

TIMESTAMP é um alias de TIMESTAMP SEM FUSO HORÁRIO.

Use o tipo de dados TIMESTAMP para armazenar valores completos de registro de data e hora que incluem a data e a hora do dia.

As colunas TIMESTAMP armazenam valores com até um máximo de seis dígitos de precisão para segundos fracionários.

Se você inserir uma data em uma coluna TIMESTAMP ou uma data com um valor de timestamp parcial, o valor será convertido implicitamente em um valor de timestamp completo. Esse valor de timestamp completo tem valores padrão (00) para horas, minutos e segundos ausentes. Os valores de fuso horário nas strings de entrada são ignorados.

Por padrão, os valores de TIMESTAMP são UTC nas tabelas de usuário e nas tabelas de sistema do Amazon Redshift.

TIMESTAMPTZ

TIMESTAMPTZ é um alias de TIMESTAMP COM FUSO HORÁRIO.

Use o tipo de dados TIMESTAMPTZ para inserir valores de registro de data e hora completos que incluem a data, a hora do dia e um fuso horário. Quando um valor de entrada inclui um fuso horário, o Amazon Redshift usa o fuso horário para converter o valor em UTC e armazena o valor UTC.

Para visualizar uma lista de nomes de fusos horários compatíveis, execute o comando a seguir.

```
select pg_timezone_names();
```

Para visualizar uma lista de abreviações de fusos horários compatíveis, execute o comando a seguir.

```
select pg_timezone_abbrevs();
```

Você também pode encontrar informações atuais sobre fusos horários no Banco de dados de fuso horário de IANA.

A tabela a seguir tem exemplos de formatos de fuso horário.

Formato	Exemplo
dd mon hh:mi:ss yyyy tz	17 Dez 07:37:16 1997 PST
mm/dd/yyyy hh:mi:ss.ss tz	12/17/1997 07:37:16.00 PST
mm/dd/yyyy hh:mi:ss.ss tz	12/17/1997 07:37:16.00 EUA/Pacífico
yyyy-mm-dd hh:mi:ss+/-tz	1997-12-17 07:37:16-08
dd.mm.yyyy hh:mi:ss tz	17.12.1997 07:37:16.00 PST

As colunas TIMESTAMPTZ armazenam valores com até um máximo de seis dígitos de precisão para segundos fracionários.

Se você inserir uma data em uma coluna TIMESTAMPTZ, ou uma data com um timestamp parcial, o valor será convertido implicitamente em um valor de timestamp completo. Esse valor de timestamp completo tem valores padrão (00) para horas, minutos e segundos ausentes.

Os valores TIMESTAMPTZ são em UTC em tabelas de usuário.

Exemplos com tipos de datetime

A seguir, você encontrará exemplos para trabalhar com tipos de data e hora compatíveis com o Amazon Redshift.

Exemplos de data

Os exemplos a seguir inserem datas que possuem formatos diferentes e exibem a saída.

```
create table datetable (start_date date, end_date date);
```

```
insert into datetable values ('2008-06-01','2008-12-31');

insert into datetable values ('Jun 1,2008','20081231');
```

```
select * from datetable order by 1;

start_date |  end_date
```

```
-----------------------
2008-06-01 | 2008-12-31
2008-06-01 | 2008-12-31
```

Se você inserir um valor de timestamp em uma coluna DATE, a parte da hora será ignorada e apenas a data será carregada.

Exemplos de tempo

Os exemplos a seguir inserem os valores TIME e TIMETZ com formatos diferentes e exibem a saída.

```
create table timetable (start_time time, end_time timetz);
```

```
insert into timetable values ('19:11:19','20:41:19 UTC');
insert into timetable values ('191119', '204119 UTC');
```

```
select * from timetable order by 1;
start_time |  end_time
-----------------------
 19:11:19  | 20:41:19+00
 19:11:19  | 20:41:19+00
```

Exemplos de time stamp

Se você inserir uma data em uma coluna TIMESTAMP ou TIMESTAMPTZ, a hora é revertida para meia-noite. Por exemplo, se você inserir o literal `20081231`, o valor armazenado será `2008-12-31 00:00:00`.

Para alterar o fuso horário da sessão atual, use o comando SET para definir o parâmetro de configuração timezone.

O exemplo a seguir insere carimbos de data e hora com formatos diferentes e exibem a tabela resultante.

```
create table tstamp(timeofday timestamp, timeofdaytz timestamptz);

insert into tstamp values('Jun 1,2008  09:59:59', 'Jun 1,2008 09:59:59 EST' );
insert into tstamp values('Dec 31,2008 18:20','Dec 31,2008 18:20');
insert into tstamp values('Jun 1,2008  09:59:59 EST', 'Jun 1,2008 09:59:59');
```

```
SELECT * FROM tstamp;

+---------------------+------------------------+
|      timeofday      |      timeofdaytz       |
+---------------------+------------------------+
| 2008-06-01 09:59:59 | 2008-06-01 14:59:59+00 |
| 2008-12-31 18:20:00 | 2008-12-31 18:20:00+00 |
| 2008-06-01 09:59:59 | 2008-06-01 09:59:59+00 |
+---------------------+------------------------+
```

Literais de data, hora e timestamp

A seguir, estão as regras para trabalhar com literais de data, hora e timestamp compatíveis com o Amazon Redshift.

Datas

As datas de entrada a seguir são exemplos válidos de valores de data literais que você pode carregar nas tabelas do Amazon Redshift. O modo `MDY DateStyle` é considerado em vigor. Este modo significa que o valor do mês precede o valor do dia em strings tais como `1999-01-08` e `01/02/00`.

> **Note**
>
> Um literal de data ou timestamp deve ser colocado entre aspas ao carregá-lo em uma tabela.

Data de entrada	Data completa
8 de janeiro de 1999	8 de janeiro de 1999
1999-01-08	8 de janeiro de 1999
1/8/1999	8 de janeiro de 1999
01/02/00	2 de janeiro de 2000
2000-Jan-31	31 de janeiro de 2000
Jan-31-2000	31 de janeiro de 2000

Data de entrada	Data completa
31-Jan-2000	31 de janeiro de 2000
20080215	15 de fevereiro de 2008
080215	15 de fevereiro de 2008
2008.366	31 de dezembro de 2008 (a parte de três dígitos da data deve estar entre 001 e 366)

Times

Os tempos de entrada a seguir são exemplos válidos de valores de tempo literais para os tipos de dados TIME e TIMETZ que você pode carregar nas tabelas do Amazon Redshift.

Tempos de entrada	Descrição (da parte da hora)
04:05:06.789	4:05 e 6.789 segundos
04:05:06	4:05 e 6 segundos
04:05	Exatamente 4:05
040506	4:05 e 6 segundos
04:05	Exatamente 4:05; AM é opcional
04:05	Exatamente 4:05; o valor de hora deve ser menor do que 12.
16:05	Exatamente 16:05

Carimbos de data/hora

Os carimbos de data e hora de entrada a seguir são exemplos válidos de valores de tempo literais para os tipos de dados TIMESTAMP e TIMESTAMPTZ que você pode carregar nas tabelas do Amazon Redshift. Todos os literais de data válidos podem ser combinados com os seguintes literais de hora.

Time stamps de entrada (datas e horas concatenadas)	Descrição (da parte da hora)
20080215 04:05:06.789	4:05 e 6,789 segundos
20080215 04:05:06	4:05 e 6 segundos
20080215 04:05	Exatamente 4:05
20080215 040506	4:05 e 6 segundos
20080215 04:05 AM	Exatamente 4:05; AM é opcional
20080215 04:05 PM	Exatamente 4:05; o valor de hora deve ser menor do que 12.
20080215 16:05	Exatamente 16:05
20080215	Meia noite (por padrão)

Valores especiais de datetime

Os seguintes valores especiais podem ser usados como literais de data e hora e como argumentos para funções de data. Eles exigem aspas simples e são convertidos em valores de timestamp regulares durante o processamento da consulta.

Valor especial	Descrição
`now`	Avalia para a hora de início da transação e retorna um timestamp com precisão de microssegundo.
`today`	Avalia para a data apropriada e retorna um timestamp com zeros para as partes do tempo.
`tomorrow`	Avalia para a data apropriada e retorna um timestamp com zeros para as partes do tempo.

Valor especial	Descrição
`yesterday`	Avalia para a data apropriada e retorna um timestamp com zeros para as partes do tempo.

Os exemplos a seguir mostram como `now` e `today` trabalham com a função DATEADD.

```
select dateadd(day,1,'today');

date_add
---------------------
2009-11-17 00:00:00
(1 row)

select dateadd(day,1,'now');

date_add
----------------------------
2009-11-17 10:45:32.021394
(1 row)
```

Tipos de dados e literais de intervalo

Você pode usar um tipo de dados de intervalo para armazenar espaços de tempo em unidades como `seconds`, `minutes`, `hours`, `days`, `months` e `years`. Tipos de dados e literais de intervalo podem ser usados em cálculos de data e hora, como adicionar intervalos a datas e carimbos de data e hora, somar intervalos e subtrair um intervalo de uma data ou carimbo de data e hora. Os literais de intervalo podem ser usados como valores de entrada para colunas de tipo de dados de intervalo em uma tabela.

Sintaxe do tipo de dados de intervalo

Para especificar um tipo de dados de intervalo para armazenar um espaço de tempo em anos e meses:

```
INTERVAL year_to_month_qualifier
```

Para especificar um tipo de dados de intervalo para armazenar uma duração em dias, horas, minutos e segundos:

```
INTERVAL day_to_second_qualifier [ (fractional_precision) ]
```

Sintaxe de literal de intervalo

Para especificar um literal de intervalo para definir um espaço de tempo em anos e meses:

```
INTERVAL quoted-string year_to_month_qualifier
```

Para especificar um literal de intervalo para definir uma duração em dias, horas, minutos e segundos:

```
INTERVAL quoted-string day_to_second_qualifier [ (fractional_precision) ]
```

Argumentos

quoted-string

Determina um valor numérico positivo ou negativo especificando uma quantidade e a unidade de data e hora como uma string de entrada. Se quoted-string contiver apenas um valor numérico, o Amazon Redshift determinará as unidades de year_to_month_qualifier ou day_to_second_qualifier. Por exemplo, `'23' MONTH` representa `1 year 11 months`, `'-2' DAY` representa `-2 days 0 hours 0 minutes 0.0 seconds`, `'1-2' MONTH` representa `1 year 2 months` e `'13 day 1 hour 1 minute 1.123 seconds' SECOND` representa `13 days 1 hour 1 minute 1.123 seconds`. Para obter mais informações sobre formatos de saída de um intervalo, consulte Estilos de intervalo.

year_to_month_qualifier

Especifica a faixa do intervalo. Se você usar um qualificador e criar um intervalo com unidades de tempo menores do que o qualificador, o Amazon Redshift truncará e descartará as partes menores do intervalo. Os valores válidos para year_to_month_qualifier são:

- `YEAR`
- `MONTH`
- `YEAR TO MONTH`

day_to_second_qualifier

Especifica a faixa do intervalo. Se você usar um qualificador e criar um intervalo com unidades de tempo menores do que o qualificador, o Amazon Redshift truncará e descartará as partes menores do intervalo. Os valores válidos para year_to_month_qualifier são:

- DAY
- HOUR
- MINUTE
- SECOND
- DAY TO HOUR
- DAY TO MINUTE
- DAY TO SECOND
- HOUR TO MINUTE
- HOUR TO SECOND
- MINUTE TO SECOND

A saída do literal INTERVAL é truncada no menor componente INTERVAL especificado. Por exemplo, ao usar um qualificador MINUTE, o Amazon Redshift descarta as unidades de tempo menores do que MINUTE.

```
select INTERVAL '1 day 1 hour 1 minute 1.123 seconds' MINUTE
```

O valor resultante é truncado para `'1 day 01:01:00'`.

fractional_precision

Parâmetro opcional que especifica o número de dígitos fracionários permitidos no intervalo. O argumento fractional_precision só deverá ser especificado se seu intervalo contiver SECOND. Por exemplo, `SECOND(3)` cria um intervalo que permite somente três dígitos fracionários, como 1.234 segundos. O número máximo de dígitos fracionários é seis.

A configuração da sessão `interval_forbid_composite_literals` determina se um erro é retornado quando um intervalo é especificado com as partes YEAR TO MONTH e DAY TO SECOND. Para obter mais informações, consulte interval_forbid_composite_literals.

Operações aritméticas de intervalo

É possível usar valores de intervalo com outros valores de data e hora para realizar operações aritméticas. As tabelas a seguir descrevem as operações disponíveis e qual tipo de dados resulta de cada operação.

> **Note**
>
> As operações que podem produzir resultados `date` e `timestamp` o fazem com base na menor unidade de tempo incluída na equação. Por exemplo, quando você adiciona `interval` a `date`, o resultado é `date`, se for um intervalo de YEAR TO MONTH, e um carimbo de data e hora, se for um intervalo de DAY TO SECOND.

As operações em que o primeiro operando é um `interval` produzem os seguintes resultados para o segundo operando fornecido:

Operador	Data	Marca de data e hora	Intervalo	Numérico
-	N/D	N/D	Intervalo	N/D
+	Data	Data/carimbo de data e hora	Intervalo	N/D
*	N/D	N/D	N/D	Intervalo
/	N/D	N/D	N/D	Intervalo

As operações em que o primeiro operando é um `date` produzem os seguintes resultados para o segundo operando fornecido:

Operador	Data	Marca de data e hora	Intervalo	Numérico
-	Numérico	Intervalo	Data/carimbo de data e hora	Data
+	N/D	N/D	N/D	N/D

As operações em que o primeiro operando é um `timestamp` produzem os seguintes resultados para o segundo operando fornecido:

Operador	Data	Marca de data e hora	Intervalo	Numérico
-	Numérico	Intervalo	Timestamp	Timestamp
+	N/D	N/D	N/D	N/D

Estilos de intervalo

É possível usar o comando SQL the section called “SET” para alterar o formato de exibição da saída dos valores do intervalo. Ao usar o tipo de dados de intervalo no SQL, converta-o em texto para ver o estilo de intervalo esperado; por exemplo, `YEAR TO MONTH::text`. Os valores disponíveis para DEFINIR o valor `IntervalStyle` são:

- `postgres`: segue o estilo do PostgreSQL. Esse é o padrão.
- `postgres_verbose`: segue o estilo detalhado do PostgreSQL.
- `sql_standard`: segue o estilo de literais de intervalo padrão do SQL.

O comando a seguir define o estilo de intervalo como `sql_standard`.

```
SET IntervalStyle to 'sql_standard';
```

formato de saída postgres

Veja abaixo o formato de saída para o estilo de intervalo `postgres`. Cada valor numérico pode ser negativo.

```
'<numeric> <unit> [, <numeric> <unit> ...]'
```

```
select INTERVAL '1-2' YEAR TO MONTH::text

varchar
---------------
1 year 2 mons
```

```
select INTERVAL '1 2:3:4.5678' DAY TO SECOND::text
```

```
varchar
------------------
1 day 02:03:04.5678
```

formato de saída postgres_verbose

A sintaxe postgres_verbose é semelhante à do postgres, mas as saídas do postgres_verbose também contêm a unidade de tempo.

```
'[@] <numeric> <unit> [, <numeric> <unit> ...] [direction]'
```

```
select INTERVAL '1-2' YEAR TO MONTH::text

varchar
-----------------
@ 1 year 2 mons
```

```
select INTERVAL '1 2:3:4.5678' DAY TO SECOND::text

varchar
--------------------------
@ 1 day 2 hours 3 mins 4.56 secs
```

formato de saída sql_standard

Os valores do intervalo de ano para mês são formatados da maneira a seguir. Especificar um sinal negativo antes do intervalo indica que o intervalo é um valor negativo e se aplica a todo o intervalo.

```
'[-]yy-mm'
```

Os valores do intervalo de dia para segundo são formatados da maneira a seguir.

```
'[-]dd hh:mm:ss.ffffff'
```

```
SELECT INTERVAL '1-2' YEAR TO MONTH::text

varchar
-------
```

```
1-2
```

```
select INTERVAL '1 2:3:4.5678' DAY TO SECOND::text

varchar
---------------
1 2:03:04.5678
```

Exemplos do tipo de dados de intervalo

Os exemplos a seguir demonstram como usar tipos de dados INTERVAL com tabelas.

```
create table sample_intervals (y2m interval month, h2m interval hour to minute);
insert into sample_intervals values (interval '20' month, interval '2 days
 1:1:1.123456' day to second);
select y2m::text, h2m::text from sample_intervals;


      y2m      |       h2m
---------------+-----------------
 1 year 8 mons | 2 days 01:01:00
```

```
update sample_intervals set y2m = interval '2' year where y2m = interval '1-8' year to
 month;
select * from sample_intervals;

   y2m   |       h2m
---------+-----------------
 2 years | 2 days 01:01:00
```

```
delete from sample_intervals where h2m = interval '2 1:1:0' day to second;
select * from sample_intervals;

 y2m | h2m
-----+-----
```

Exemplos de literais de intervalo

Os exemplos a seguir são executados com o estilo de intervalo definido como `postgres`.

O exemplo a seguir demonstra como criar um literal INTERVAL de um ano.

```
select INTERVAL '1' YEAR

intervaly2m
---------------
1 years 0 mons
```

Se você especificar um quoted-string que exceda o qualificador, as unidades de tempo restantes serão truncadas do intervalo. No exemplo a seguir, um intervalo de 13 meses se torna um ano e um mês, mas o mês restante é omitido devido ao qualificador YEAR.

```
select INTERVAL '13 months' YEAR

intervaly2m
---------------
1 years 0 mons
```

Se você usar um qualificador inferior à string de intervalo, as unidades restantes serão incluídas.

```
select INTERVAL '13 months' MONTH

intervaly2m
---------------
1 years 1 mons
```

Especificar uma precisão no intervalo trunca o número de dígitos fracionários até a precisão especificada.

```
select INTERVAL '1.234567' SECOND (3)

intervald2s
--------------------------------
0 days 0 hours 0 mins 1.235 secs
```

Se você não especificar uma precisão, o Amazon Redshift usará a precisão máxima de seis.

```
select INTERVAL '1.23456789' SECOND

intervald2s
```

```
---------------------------------
0 days 0 hours 0 mins 1.234567 secs
```

O exemplo a seguir demonstra como criar um intervalo em faixas.

```
select INTERVAL '2:2' MINUTE TO SECOND

intervald2s
-----------------------------
0 days 0 hours 2 mins 2.0 secs
```

Os qualificadores ditam as unidades que você está especificando. Por exemplo, embora o exemplo a seguir use o mesmo quoted-string de '2:2' do exemplo anterior, o Amazon Redshift reconhece que ele usa unidades de tempo diferentes por causa do qualificador.

```
select INTERVAL '2:2' HOUR TO MINUTE

intervald2s
-----------------------------
0 days 2 hours 2 mins 0.0 secs
```

Abreviações e plurais de cada unidade também são aceitos. Por exemplo, `5s`, `5 second` e `5 seconds` são intervalos equivalentes. As unidades aceitas são anos, meses, horas, minutos e segundos.

```
select INTERVAL '5s' SECOND

intervald2s
-----------------------------
0 days 0 hours 0 mins 5.0 secs
```

```
select INTERVAL '5 HOURS' HOUR

intervald2s
-----------------------------
0 days 5 hours 0 mins 0.0 secs
```

```
select INTERVAL '5 h' HOUR
```

```
intervald2s
-----------------------------
0 days 5 hours 0 mins 0.0 secs
```

Exemplos de literais de intervalo sem sintaxe de qualificador

> **Note**
>
> Os exemplos a seguir demonstram o uso de um literal de intervalo sem um qualificador `YEAR TO MONTH` ou `DAY TO SECOND`. Para obter informações sobre como usar o literal de intervalo recomendado com um qualificador, consulte Tipos de dados e literais de intervalo.

Use um literal de intervalo para identificar períodos de tempo específicos, tais como `12 hours` ou `6 months`. Você pode usar esses literais de intervalo em condições e cálculos que envolvem expressões de data e hora.

Um literal de intervalo é expresso como uma combinação da palavra-chave de INTERVAL com uma quantidade numérica e uma parte da data compatível; por exemplo `INTERVAL '7 days'` ou `INTERVAL '59 minutes'`. Você pode conectar várias quantidades e unidades para formar um intervalo mais preciso, por exemplo: `INTERVAL '7 days, 3 hours, 59 minutes'`. As abreviaturas e os plurais de cada unidade também são compatíveis; por exemplo: `5 s`, `5 second` e `5 seconds` são intervalos equivalentes.

Se você não especificar uma parte da data, o valor do intervalo representará os segundos. Você pode especificar o valor de quantidade como uma fração (por exemplo: `0.5 days`).

Os exemplos a seguir mostram uma série de cálculos com diferentes valores de intervalo.

O exemplo a seguir adiciona 1 segundo à data especificada.

```
select caldate + interval '1 second' as dateplus from date
where caldate='12-31-2008';
dateplus
---------------------
2008-12-31 00:00:01
(1 row)
```

O exemplo a seguir adiciona 1 minuto à data especificada.

```
select caldate + interval '1 minute' as dateplus from date
where caldate='12-31-2008';
dateplus
---------------------
2008-12-31 00:01:00
(1 row)
```

O exemplo a seguir adiciona 3 horas e 35 minutos à data especificada.

```
select caldate + interval '3 hours, 35 minutes' as dateplus from date
where caldate='12-31-2008';
dateplus
---------------------
2008-12-31 03:35:00
(1 row)
```

O exemplo a seguir adiciona 52 semanas à data especificada.

```
select caldate + interval '52 weeks' as dateplus from date
where caldate='12-31-2008';
dateplus
---------------------
2009-12-30 00:00:00
(1 row)
```

O seguinte adiciona 1 semana, 1 hora, 1 minuto e 1 segundo à data especificada.

```
select caldate + interval '1w, 1h, 1m, 1s' as dateplus from date
where caldate='12-31-2008';
dateplus
---------------------
2009-01-07 01:01:01
(1 row)
```

O exemplo a seguir adiciona 12 horas (meio dia) à data especificada.

```
select caldate + interval '0.5 days' as dateplus from date
where caldate='12-31-2008';
dateplus
---------------------
2008-12-31 12:00:00
```

```
(1 row)
```

O exemplo a seguir subtrai quatro meses de 15 de fevereiro de 2023 e o resultado é 15 de outubro de 2022.

```
select date '2023-02-15' - interval '4 months';

?column?
---------------------
2022-10-15 00:00:00
```

O exemplo a seguir subtrai quatro meses de 31 de março de 2023 e o resultado é 30 de novembro de 2022. O cálculo considera o número de dias em um mês.

```
select date '2023-03-31' - interval '4 months';

?column?
---------------------
2022-11-30 00:00:00
```

Tipo booliano

Use o tipo de dados BOOLEAN para armazenar valores verdadeiros e falsos em uma coluna de único byte. A tabela a seguir descreve os três estados possíveis para um valor booleano e os valores de literal que resultam naquele estado. Independente da string de entrada, a coluna booleana armazena e fornece "t" para verdadeiro e "f" para falso.

Estado	Valores válidos de literal	Armazenamento
Verdadeiro	`TRUE 't' 'true' 'y' 'yes' '1'`	1 byte
Falso	`FALSE 'f' 'false' 'n' 'no' '0'`	1 byte
Desconhecido	`NULL`	1 byte

É possível usar uma comparação IS para verificar um valor booleano somente como um predicado na cláusula WHERE. Não é possível usar a comparação IS com um valor booleano na lista SELECT.

Exemplos

Você pode usar uma coluna BOOLEAN para armazenar um estado "Ativo/inativo" para cada cliente em uma tabela CUSTOMER.

```
create table customer(
custid int,
active_flag boolean default true);
```

```
insert into customer values(100, default);
```

```
select * from customer;
custid | active_flag
-------+-------------
   100 | t
```

Se nenhum valor padrão (`true` ou `false`) é especificado na instrução CREATE TABLE, inserir um nome padrão significa inserir um null.

Neste exemplo, a consulta seleciona usuários da tabela USERS que gostam de esportes, mas não gostam de teatro:

```
select firstname, lastname, likesports, liketheatre
from users
where likesports is true and liketheatre is false
order by userid limit 10;

firstname |  lastname  | likesports | liketheatre
----------+------------+------------+-------------
Lars      | Ratliff    | t          | f
Mufutau   | Watkins    | t          | f
Scarlett  | Mayer      | t          | f
Shafira   | Glenn      | t          | f
Winifred  | Cherry     | t          | f
Chase     | Lamb       | t          | f
Liberty   | Ellison    | t          | f
Aladdin   | Haney      | t          | f
Tashya    | Michael    | t          | f
```

```
Lucian    | Montgomery | t          | f
(10 rows)
```

O exemplo a seguir seleciona usuários da tabela USERS para os quais não se sabe se eles gostam de rock:

```
select firstname, lastname, likerock
from users
where likerock is unknown
order by userid limit 10;

firstname | lastname | likerock
----------+----------+----------
Rafael    | Taylor   |
Vladimir  | Humphrey |
Barry     | Roy      |
Tamekah   | Juarez   |
Mufutau   | Watkins  |
Naida     | Calderon |
Anika     | Huff     |
Bruce     | Beck     |
Mallory   | Farrell  |
Scarlett  | Mayer    |
(10 rows)
```

O exemplo a seguir retorna um erro porque ele usa uma comparação IS na lista SELECT.

```
select firstname, lastname, likerock is true as "check"
from users
order by userid limit 10;

[Amazon](500310) Invalid operation: Not implemented
```

O exemplo a seguir é bem-sucedido porque ele usa uma comparação igual (=) na lista SELECT em vez da comparação IS.

```
select firstname, lastname, likerock = true as "check"
from users
order by userid limit 10;

firstname | lastname  | check
----------+-----------+------
```

```
Rafael    | Taylor    |
Vladimir  | Humphrey  |
Lars      | Ratliff   | true
Barry     | Roy       |
Reagan    | Hodge     | true
Victor    | Hernandez | true
Tamekah   | Juarez    |
Colton    | Roy       | false
Mufutau   | Watkins   |
Naida     | Calderon  |
```

Tipo HLLSKETCH

Use o tipo de dados HLLSKETCH para esboços do HyperLogLog. O Amazon Redshift oferece suporte a representações de esboço do HyperLogLog que são esparsas ou densas. Os esboços começam como esparsos e mudam para denso quando o formato denso é mais eficiente para minimizar o espaço de memória usado.

O Amazon Redshift faz a transição automática de um esboço do HyperLogLog esparso ao importar, exportar ou imprimir esboços no seguinte formato JSON.

```
{"logm":15,"sparse":{"indices":[4878,9559,14523],"values":[1,2,1]}}
```

O Amazon Redshift usa uma representação de string em um formato Base64 para representar um esboço denso do HyperLogLog.

O Amazon Redshift usa a seguinte representação de string em um formato Base64 para representar um esboço denso do HyperLogLog.

```
"ABAABA..."
```

O tamanho máximo de um objeto HLLSKETCH é 24.580 bytes quando usado na compactação bruta.

Tipo SUPER

Use o tipo de dados SUPER para armazenar dados semiestruturados ou documentos como valores. Embora o Amazon Redshift seja capaz de armazenar esses valores usando VARCHAR, recomendamos que você use o tipo de dados SUPER em vez disso.

Os dados semiestruturados não estão em conformidade com a estrutura rígida e tabular do modelo de dados relacional usado em bancos de dados SQL. Ele contém etiquetas que fazem referência

a entidades distintas dentro dos dados. Eles podem conter valores complexos, como matrizes, estruturas aninhadas e outras estruturas complexas associadas a formatos de serialização, como JSON. O tipo de dados SUPER é um conjunto de valores de array e estrutura sem esquema que englobam todos os outros tipos escalares do Amazon Redshift.

O tipo de dado SUPER comporta até 16 MB de dados para um objeto SUPER individual. Para obter mais informações sobre o tipo de dados SUPER, incluindo exemplos de implementação em uma tabela, consulte Dados semiestruturados no Amazon Redshift.

O Amazon Redshift fornece suporte integrado para ingerir os seguintes formatos de dados semiestruturados usando o comando COPY:

- JSON
- ARRAY
- TEXT
- CSV

Objetos SUPER maiores que 1 MB só podem ser ingeridos nos seguintes formatos de arquivo:

- Parquet
- JSON
- TEXT
- CSV

O tipo de dados SUPER tem as seguintes propriedades:

- Um valor escalar do Amazon Redshift:
  - Um nulo
  - Um booleano
  - Um número, como smallint, inteiro, bigint, decimal ou ponto flutuante (como float4 ou float8)
  - Um valor de string, como varchar ou char
- Um valor complexo:
  - Um array de valores, incluindo escalar ou complexo
  - Uma estrutura, também conhecida como tupla ou objeto, que é um mapa de nomes e valores de atributos (escalar ou complexo)

Qualquer um dos dois tipos de valores complexos contém seus próprios escalares ou valores complexos sem ter quaisquer restrições de regularidade.

A codificação de compressão padrão para o tipo de dados SUPER é ZSTD. Para obter mais informações sobre a codificação de compactação, consulte Codificações de compactação.

O tipo de dados SUPER suporta a persistência de dados semiestruturados em uma forma sem esquema. Embora modelo de dados hierárquico pode mudar, as versões antigas de dados podem coexistir na mesma coluna SUPER.

O Amazon Redshift usa o PartiQL para habilitar a navegação em matrizes e estruturas. O Amazon Redshift também usa a sintaxe PartiQL para iterar matrizes SUPER. Para obter mais informações, consulte PartiQL: uma linguagem de consultas compatível com SQL para o Amazon Redshift.

O Amazon Redshift usa digitação dinâmica para processar dados SUPER sem esquema sem a necessidade de declarar os tipos de dados antes de usá-los na consulta. Para obter mais informações, consulte Digitação dinâmica.

Você pode aplicar políticas de mascaramento de dados dinâmicas aos valores escalares nos caminhos das colunas do tipo SUPER. Para obter mais informações sobre mascaramento de dados dinâmico, consulte Mascaramento dinâmico de dados. Para obter mais informações sobre como usar o mascaramento de dados dinâmico com o tipo de dados SUPER, consulte Uso do mascaramento de dados dinâmico com caminhos do tipo de dados SUPER.

Recomendamos que você defina a opção de configuração `r_enable_case_sensitive_super_attribute` como verdadeira ao trabalhar com dados SUPER. Para obter mais informações, consulte enable_case_sensitive_super_attribute.

Tipo VARBYTE

Use uma coluna VARBYTE, VARBINARY ou BINARY VARYING para armazenar valores binários de tamanho variável com limite fixo.

```
varbyte [ (n) ]
```

O número máximo de bytes (n) pode variar de 1 a 16.777.216. O padrão é 64.000.

Estes são alguns exemplos em que convém usar um tipo de dados VARBYTE:

- Unir tabelas em colunas VARBYTE.

- Criar visões materializadas que contenham colunas VARBYTE. Há suporte para a atualização incremental de visões materializadas que contêm colunas VARBYTE. Porém, funções agregadas que não sejam COUNT, MIN e MAX e GROUP BY nas colunas VARBYTE não são compatíveis com atualização incremental.

Para garantir que todos os bytes sejam caracteres imprimíveis, o Amazon Redshift usa o formato hexadecimal para imprimir valores VARBYTE. Por exemplo, o seguinte SQL converte a cadeia de caracteres hexadecimal 6162 em um valor binário. Mesmo que o valor retornado seja um valor binário, os resultados são impressos como hexadecimais 6162.

```
select from_hex('6162');

 from_hex
----------
 6162
```

O Amazon Redshift é compatível com a conversão entre VARBYTE e os seguintes tipos de dados:

- CHAR
- VARCHAR
- SMALLINT
- INTEGER
- BIGINT

Ao converter CHAR e VARCHAR, utiliza-se o formato UTF-8. Para obter mais informações sobre o formato do UTF-8, consulte TO_VARBYTE. Ao converter SMALLINT, INTEGER e BIGINT, mantém-se o número de bytes do tipo de dados original. São dois bytes para SMALLINT, quatro bytes para INTEGER e oito bytes para BIGINT.

A instrução SQL a seguir converte uma cadeia de caracteres VARCHAR para VARBYTE. Mesmo que o valor retornado seja um valor binário, os resultados são impressos como hexadecimais 616263.

```
select 'abc'::varbyte;

 varbyte
---------
```

```
 616263
```

A instrução SQL a seguir converte um valor CHAR em uma coluna para VARBYTE. Este exemplo cria uma tabela com uma coluna (c) CHAR(10), insere valores de caracteres menores que o comprimento de 10. A conversão resultante insere o resultado com caracteres de espaço (hex'20') para o tamanho de coluna definido. Mesmo que o valor retornado seja um valor binário, os resultados são impressos como hexadecimais.

```
create table t (c char(10));
insert into t values ('aa'), ('abc');
select c::varbyte from t;
          c
----------------------
 61612020202020202020
 61626320202020202020
```

A instrução SQL a seguir converte uma cadeia de caracteres SMALLINT para VARBYTE. Mesmo que o valor retornado seja um valor binário, os resultados são impressos como hexadecimais `0005`, que são caracteres hexadecimais de dois ou quatro bytes.

```
select 5::smallint::varbyte;

 varbyte
---------
 0005
```

A instrução SQL a seguir converte um INTEGER para VARBYTE. Mesmo que o valor retornado seja um valor binário, os resultados são impressos como hexadecimais `00000005`, que são caracteres hexadecimais de quatro ou oito bytes.

```
select 5::int::varbyte;

 varbyte
----------
 00000005
```

A instrução SQL a seguir converte um BIGINT para VARBYTE. Mesmo que o valor retornado seja um valor binário, os resultados são impressos como hexadecimais `0000000000000005`, que são caracteres hexadecimais de oito ou 16 bytes.

```
select 5::bigint::varbyte;

     varbyte
------------------
 0000000000000005
```

Os recursos do Amazon Redshift compatíveis com o tipo de dados VARBYTE incluem:

- Operadores VARBYTE
- CONCAT
- LEN
- Função LENGTH
- OCTET_LENGTH
- Função SUBSTRING
- FROM_HEX
- TO_HEX
- FROM_VARBYTE
- TO_VARBYTE
- GETBIT
- Carregar uma coluna do tipo de dados VARBYTE
- Descarregar uma coluna do tipo de dados VARBYTE

Limitações ao usar dados do tipo VARBYTE com o Amazon Redshift

Estas são as limitações ao usar dados do tipo VARBYTE com o Amazon Redshift:

- O Amazon Redshift Spectrum é compatível com o tipo de dado VARBYTE somente para arquivos Parquet e ORC.
- O editor de consultas do Amazon Redshift e o editor de consultas do Amazon Redshift v2 ainda não são totalmente compatíveis com o tipo de dados VARBYTE. Portanto, use um cliente SQL diferente ao trabalhar com expressões VARBYTE.

  Como uma solução alternativa para usar o editor de consultas, se o comprimento dos dados for 16 milhões de bytes ou menos e o conteúdo for UTF-8 válido, você poderá converter os valores VARBYTE em VARCHAR, por exemplo:

```
select to_varbyte('6162', 'hex')::varchar;
```

- Não é possível usar tipos de dados VARBYTE com funções definidas pelo usuário (UDFs) em Python ou Lambda.
- Não é possível criar uma coluna HLLSKETCH a partir de uma coluna VARBYTE ou usar APPROXIMATE COUNT DISTINCT em uma coluna VARBYTE.
- Valores de VARBYTE maiores que 1 MB só podem ser ingeridos nos seguintes formatos de arquivo:
  - Parquet
  - Texto
  - Valores separados por vírgula (CSV)

Compatibilidade e conversão dos tipos

A seguir, você pode encontrar uma discussão sobre como as regras de conversão de tipo e a compatibilidade de tipo de dados funcionam no Amazon Redshift.

Compatibilidade

A correspondência de tipo de dados e de valores e constantes literais com tipos de dados ocorre durante diversas operações do banco de dados, incluindo o seguinte:

- Operações de linguagem de manipulação de dados (DML) em tabelas
- Consultas de UNION, INTERSECT e EXCEPT
- Expressões de CASOS
- Avaliação de predicados, tais como LIKE e IN
- Avaliação de funções SQL que fazem comparações ou extrações de dados
- Comparações com operadores matemáticos

Os resultados dessas operações dependem das regras de conversão de tipo e da compatibilidade dos tipos de dados. Compatibilidade indica que uma correspondência linear de determinado valor e determinado tipo de dado nem sempre é necessária. Como alguns tipos de dados são compatíveis, uma conversão implícita, ou coerção, é possível (para mais informações, consulte Tipos de conversão implícita). Quando os tipos de dados são incompatíveis, você pode às vezes converter um valor de um tipo de dados para outro usando uma função de conversão explícita.

Regras gerais de compatibilidade e conversão

Observe as seguintes regras de compatibilidade e conversão:

- Em geral, tipos de dados que se enquadram no mesmo tipo de categoria (como diferentes tipos de dados numéricos) são compatíveis e podem ser implicitamente convertidos.

  Por exemplo, com a conversão implícita você pode inserir um valor decimal em uma coluna de número inteiro. O decimal é arredondado para produzir um número inteiro. Ou você pode extrair um valor numérico, tal como `2008`, a partir de uma data e inserir este valor uma coluna de inteiro.

- Tipos de dados numéricos aplicam as condições de transbordamento que ocorrem quando você tenta inserir valores fora do intervalo. Por exemplo, um valor decimal com uma precisão de 5 não se enquadra em uma coluna decimal que foi definida com uma precisão de 4. Um número inteiro ou parte inteira de um decimal nunca é truncada; entretanto, a parte fracionária de um decimal pode ser arredondada para cima ou para baixo, conforme apropriado. Contudo, os resultados de conversões explícitas dos valores selecionados a partir de tabelas não são arredondados.

- Diferentes tipos de strings de caracteres são compatíveis; strings da coluna VARCHAR contendo dados único byte e strings de coluna CHAR são comparáveis e implicitamente conversíveis. Strings VARCHAR que contêm dados de multibyte não são comparáveis. Além disso, você pode converter uma string de caracteres em uma data, hora, timestamp ou valor numérico se a string for um valor literal apropriado; quaisquer espaços à esquerda ou à direita são ignorados. Por outro lado, você pode converter uma data, hora, timestamp ou valor numérico em uma sequência de caracteres de comprimento fixo ou variável.

  > **Note**
  >
  > Uma string de caracteres que você queira converter em um tipo numérico deve conter uma representação de caractere de um número. Por exemplo, você pode converter as strings `'1.0'` ou `'5.9'` em valores decimais, mas não pode converter a string `'ABC'` em qualquer tipo numérico.

- Se você comparar valores do tipo DECIMAL com strings de caracteres, o Amazon Redshift tentará converter a string de caracteres em um valor DECIMAL. Ao comparar todos os outros valores numéricos com strings de caracteres, os valores numéricos são convertidos em strings de caracteres. Para impor a conversão oposta (por exemplo, converter strings de caracteres em números inteiros ou converter valores do tipo DECIMAL em strings de caracteres), use uma função explícita, como CAST.

- Para converter valores do tipo DECIMAL ou NUMERIC de 64 bits em uma precisão mais alta, você deve usar uma função de conversão explícita tal como CAST ou CONVERT.
- Ao converter DATE ou TIMESTAMP em TIMESTAMP ou converter TIME em TIMETZ, o fuso horário é definido para o fuso horário da sessão atual. O fuso horário da sessão é UTC por padrão. Para obter mais informações sobre como configurar o fuso horário da sessão, consulte timezone.
- Da mesma forma, TIMESTAMPTZ é convertido em DATE, TIME ou TIMESTAMP com base no fuso horário da sessão atual. O fuso horário da sessão é UTC por padrão. Após a conversão, as informações de fuso horário são removidas.
- As strings de caracteres que representam um timestamp com fuso horário especificado são convertidas em TIMESTAMPTZ usando o fuso horário da sessão atual, que é UTC por padrão. Da mesma forma, strings de caracteres que representam uma hora com fuso horário especificado são convertidas em TIMETZ usando o fuso horário da sessão atual, que é UTC por padrão.

Tipos de conversão implícita

Há dois tipos de conversão implícita:

- Conversões implícitas em atribuições, tais como na definição de valores nos comandos INSERT ou UPDATE.
- Conversões implícitas em expressões, tal como a execução de comparações na cláusula WHERE.

A seguinte tabela lista os tipos de dados que podem ser convertidos implicitamente nas atribuições ou expressões. Você também pode usar uma função de conversão explícita para realizar essas conversões.

Do tipo	Para o tipo
BIGINT (INT8)	BOOLEAN
	CHAR
	DECIMAL (NUMERIC)
	DOUBLE PRECISION (FLOAT8)
	INTEGER (INT, INT4)

Do tipo	Para o tipo
	REAL (FLOAT4)
	SMALLINT (INT2)
	VARCHAR
CHAR	VARCHAR
DATE	CHAR
	VARCHAR
	TIMESTAMP
	TIMESTAMPTZ
DECIMAL (NUMERIC)	BIGINT (INT8)
	CHAR
	DOUBLE PRECISION (FLOAT8)
	INTEGER (INT, INT4)
	REAL (FLOAT4)
	SMALLINT (INT2)
	VARCHAR
DOUBLE PRECISION (FLOAT8)	BIGINT (INT8)
	CHAR
	DECIMAL (NUMERIC)
	INTEGER (INT, INT4)
	REAL (FLOAT4)

Do tipo	Para o tipo
	SMALLINT (INT2)
	VARCHAR
INTEGER (INT, INT4)	BIGINT (INT8)
	BOOLEAN
	CHAR
	DECIMAL (NUMERIC)
	DOUBLE PRECISION (FLOAT8)
	REAL (FLOAT4)
	SMALLINT (INT2)
	VARCHAR
REAL (FLOAT4)	BIGINT (INT8)
	CHAR
	DECIMAL (NUMERIC)
	INTEGER (INT, INT4)
	SMALLINT (INT2)
	VARCHAR
SMALLINT (INT2)	BIGINT (INT8)
	BOOLEAN
	CHAR
	DECIMAL (NUMERIC)

Do tipo	Para o tipo
	DOUBLE PRECISION (FLOAT8)
	INTEGER (INT, INT4)
	REAL (FLOAT4)
	VARCHAR
TIMESTAMP	CHAR
	DATE
	VARCHAR
	TIMESTAMPTZ
	TIME
TIMESTAMPTZ	CHAR
	DATE
	VARCHAR
	TIMESTAMP
	TIMETZ
TIME	VARCHAR
	TIMETZ
	INTERVALO ENTRE UM DIA E UM SEGUNDO
TIMETZ	VARCHAR
	TIME
GEOMETRY	GEOGRAPHY

Do tipo	Para o tipo
GEOGRAPHY	GEOMETRY

> **Note**
>
> As conversões implícitas entre TIMESTAMPTZ, TIMESTAMP, DATE, TIME, TIMETZ ou strings de caracteres usam o fuso horário da sessão atual. Para obter mais informações sobre como definir o fuso horário da sessão atual, consulte timezone.
> Os tipos de dados GEOMETRY e GEOGRAPHY não podem ser convertidos explicitamente em outro tipo de dados, a não ser entre si. Para obter mais informações, consulte Função CAST.
> O tipo de dados VARBYTE não pode ser convertido explicitamente em outro tipo de dados. Para obter mais informações, consulte Função CAST.

Usar a digitação dinâmica para o tipo de dados SUPER

O Amazon Redshift usa a digitação dinâmica para processar dados SUPER sem esquema sem a necessidade de declarar os tipos de dados antes de usá-los em sua consulta. A digitação dinâmica usa os resultados da navegação em colunas de dados SUPER sem ter que convertê-las explicitamente nos tipos do Amazon Redshift. Para obter mais informações sobre como usar a digitação dinâmica para tipo de dados SUPER, consulte Digitação dinâmica.

Você pode converter valores SUPER de e para outros tipos de dados com algumas exceções. Para obter mais informações, consulte Limitações.

Sequências de colação

O Amazon Redshift não oferece suporte a sequências de agrupamento específicas de localidade ou definidas pelo usuário. Geralmente, os resultados de qualquer predicado em qualquer contexto podem ser afetados pela ausência de regras específicas do local para classificação e comparação de valores de dados. Por exemplo, expressões ORDER BY e funções como MIN, MAX e RANK retornam resultados baseados em uma classificação UTF8 binária dos dados que não considera caracteres específicos do local.

Expressões

Tópicos

- Expressões simples
- Expressões compostas
- Listas de expressões
- Subconsultas escalares
- Expressões de função

Uma expressão é uma combinação de um ou mais valores, operadores ou funções que avaliam a um valor. O tipo de dados de uma expressão é geralmente o de seus componentes.

Expressões simples

Uma expressão simples é uma das seguintes opções:

- Um valor constante ou literal
- Um nome de coluna ou uma referência de coluna
- Uma função escalar
- Uma função agregada (definida)
- Uma função de janela
- Uma subconsulta escalar

Exemplos de expressões simples incluem:

```
5+12
dateid
sales.qtysold * 100
sqrt (4)
max (qtysold)
(select max (qtysold) from sales)
```

Expressões compostas

Uma expressão composta é uma série de expressões simples juntadas por operadores aritméticos. Uma expressão simples usada em uma expressão composta deve retornar um valor numérico.

Sintaxe

```
expression
operator
expression | (compound_expression)
```

Argumentos

expressão

Uma expressão simples que avalia para um valor.

Operador

Uma expressão aritmética composta pode ser construída usando os seguintes operadores, nesta ordem de precedência:

- ( ) : parênteses para controlar a ordem de avaliação
- + , - : sinal/operador positivo e negativo
- ^ , |/ , ||/ : potenciação, raiz quadrada, raiz cúbica
- * , / , % : multiplicação, divisão e operadores de módulo
- @ : valor absoluto
- + , - : adição e subtração
- & , |, #, ~, <<, >> : AND, OR, NOT, shift esquerda, shift direita
- ||: concatenação

(compound_expression)

Expressões compostas podem ser aninhadas usando parênteses.

Exemplos

Exemplos de expressões compostas incluem o seguinte:

```
('SMITH' || 'JONES')
sum(x) / y
sqrt(256) * avg(column)
rank() over (order by qtysold) / 100
(select (pricepaid - commission) from sales where dateid = 1882) * (qtysold)
```

Algumas funções também podem ser aninhadas em outras funções. Por exemplo, qualquer função escalar pode aninhar-se em outra função escalar. O seguinte exemplo retorna a soma dos valores absolutos de um conjunto de números:

```
sum(abs(qtysold))
```

Funções da janela não podem ser usadas como argumentos para funções agregadas ou outras funções da janela. A seguinte expressão retornaria um erro:

```
avg(rank() over (order by qtysold))
```

Funções da janela podem ter uma função agregada aninhada. A seguinte expressão soma conjuntos de valores e depois os classifica:

```
rank() over (order by sum(qtysold))
```

Listas de expressões

Uma lista de expressão é uma combinação de expressões e pode aparecer em condições de associação e de comparação (cláusulas WHERE) e em cláusulas GROUP BY.

Sintaxe

```
expression , expression , ... | (expression, expression, ...)
```

Argumentos

expressão

Uma expressão simples que avalia para um valor. Uma lista de expressões pode conter uma ou mais expressões separadas por vírgula ou um ou mais conjuntos de expressões separados por vírgula. Quando há vários conjuntos de expressões, cada conjunto deve conter o mesmo número de expressões e estar separado por parênteses. O número de expressões em cada conjunto deve corresponder ao número de expressões antes do operador na condição.

Exemplos

A seguir, exemplos de listas de expressões nas condições:

```
(1, 5, 10)
('THESE', 'ARE', 'STRINGS')
(('one', 'two', 'three'), ('blue', 'yellow', 'green'))
```

O número de expressões em cada conjunto deve corresponder ao número na primeira parte da instrução:

```
select * from venue
where (venuecity, venuestate) in (('Miami', 'FL'), ('Tampa', 'FL'))
order by venueid;

venueid |        venuename        | venuecity | venuestate | venueseats
---------+-------------------------+-----------+------------+------------
28 | American Airlines Arena | Miami     | FL         |          0
54 | St. Pete Times Forum    | Tampa     | FL         |          0
91 | Raymond James Stadium   | Tampa     | FL         |      65647
(3 rows)
```

Subconsultas escalares

Uma subconsulta escalar é uma consulta SELECT regular entre parênteses que retorna exatamente um valor: uma linha com uma coluna. A consulta é executada, e o valor retornado é usado na consulta externa. Se a subconsulta retornar zero linhas, o valor da expressão da subconsulta será nulo. Se ele retornar mais de uma linha, o Amazon Redshift retornará um erro. A subconsulta pode consultar variáveis da consulta pai, que atuarão como constantes durante qualquer invocação da subconsulta.

Você pode usar subconsultas escalares na maioria das instruções que pedem uma expressão. As subconsultas escalares não são expressões válidas nos seguintes casos:

- Como valores padrão para expressões
- Em cláusulas GROUP BY e HAVING

Exemplo

As seguintes subconsultas computam o preço médio pago por venda ao longo de todo o ano de 2008; então, a consulta externa usa este valor na saída para comparar com o preço médio por venda por trimestre:

```
select qtr, avg(pricepaid) as avg_saleprice_per_qtr,
```

```
(select avg(pricepaid)
from sales join date on sales.dateid=date.dateid
where year = 2008) as avg_saleprice_yearly
from sales join date on sales.dateid=date.dateid
where year = 2008
group by qtr
order by qtr;
qtr  | avg_saleprice_per_qtr | avg_saleprice_yearly
-------+----------------------+---------------------
1     |               647.64 |              642.28
2     |               646.86 |              642.28
3     |               636.79 |              642.28
4     |               638.26 |              642.28
(4 rows)
```

Expressões de função

Sintaxe

Qualquer incorporação pode ser usada como uma expressão. A sintaxe para uma chamada de função é o nome de uma função acompanhado por sua lista de argumentos entre parênteses.

```
function ( [expression [, expression...]] )
```

Argumentos

Função

Qualquer função incorporada. Para obter algumas funções de exemplo, consulte Referência de funções SQL.

expressão

Qualquer expressão(ões) que corresponda(m) ao tipo de dado e contagem de parâmetros previstos pela função.

Exemplos

```
abs (variable)
select avg (qtysold + 3) from sales;
select dateadd (day,30,caldate) as plus30days from date;
```

Condições

Tópicos

- Sintaxe
- Condição de comparação
- Condições lógicas
- Condições de correspondência de padrões
- Condição de intervalo BETWEEN
- Condição null
- Condição EXISTS
- Condição IN

Uma condição é uma instrução de uma ou mais expressões e operadores lógicos que avalia para verdadeiro, falso ou desconhecido. As condições também são ocasionalmente chamadas de predicados.

> **Note**
>
> Todas as comparações de strings e correspondências de padrão LIKE diferenciam entre letras maiúsculas e minúsculas. Por exemplo, “A” e “a” não são correspondentes. No entanto, você pode fazer uma correspondência de padrão que não diferencia maiúsculas e minúsculas usando o predicado ILIKE.

Sintaxe

```
comparison_condition
| logical_condition
| range_condition
| pattern_matching_condition
| null_condition
| EXISTS_condition
| IN_condition
```

Condição de comparação

A comparação de condições indica as relações lógicas entre dois valores. Todas as condições de comparação são operadores binários com um tipo de retorno booleano. O Amazon Redshift oferece suporte para os operadores de comparação descritos na seguinte tabela:

Operador	Sintaxe	Descrição
<	`a < b`	O valor a é menor que o valor b.
>	`a > b`	O valor a é maior que o valor b.
<=	`a <= b`	O valor a é menor ou igual ao valor b.
>=	`a >= b`	O valor a é maior ou igual ao valor b.
=	`a = b`	O valor a é igual ao valor b.
<> ou !=	`a <> b or a != b`	O valor a não é igual ao valor b.
ANY | SOME	`a = ANY(subquery)`	O valor a é igual a qualquer valor retornado pela subconsulta.
ALL	`a <> ALL or != ALL (subquery))`	O valor a não é igual a qualquer valor retornado pela subconsulta.
IS TRUE | FALSE | UNKNOWN	`a IS TRUE`	O valor a é um booleano TRUE.

Observações de uso

= ANY | SOME

As palavras-chave ANY e SOME são sinônimos da condição IN e retornam true se a comparação for true para pelo menos um valor retornado por uma subconsulta que retorna um ou mais valores. O Amazon Redshift suporta apenas a condição = (igual) para ANY e SOME. As condições de desigualdade não são compatíveis.

> **Note**
>
> O predicado ALL não é compatível.

<> ALL

A palavra-chave ALL é sinônimo de NOT IN (consulte a condição Condição IN) e retorna true se a expressão não for incluída nos resultados da subconsulta. O Amazon Redshift oferece suporte apenas à condição <> ou != (não é igual) para ALL. Outras condições de comparação não são compatíveis.

IS TRUE/FALSE/UNKNOWN

Valores diferentes de zero equivalem a TRUE, 0 equivale a FALSE, e nulo equivale a UNKNOWN. Consulte o tipo de dado Tipo booliano.

Exemplos

Veja alguns exemplos simples de condições de comparação:

```
a = 5
a < b
min(x) >= 5
qtysold = any (select qtysold from sales where dateid = 1882
```

A seguinte consulta retorna locais com mais de 10.000 assentos da tabela VENUE:

```
select venueid, venuename, venueseats from venue
where venueseats > 10000
order by venueseats desc;

venueid |           venuename            | venueseats
---------+--------------------------------+------------
83 | FedExField                    |      91704
 6 | New York Giants Stadium       |      80242
79 | Arrowhead Stadium             |      79451
78 | INVESCO Field                 |      76125
69 | Dolphin Stadium               |      74916
67 | Ralph Wilson Stadium          |      73967
```

```
76 | Jacksonville Municipal Stadium |      73800
89 | Bank of America Stadium        |      73298
72 | Cleveland Browns Stadium       |      73200
86 | Lambeau Field                  |      72922
...
(57 rows)
```

Este exemplo seleciona os usuários (USERID) da tabela USERS que gostam de rock:

```
select userid from users where likerock = 't' order by 1 limit 5;

userid
--------
3
5
6
13
16
(5 rows)
```

Este exemplo seleciona os usuários (USERID) da tabela USERS onde não se sabe se eles gostam de rock:

```
select firstname, lastname, likerock
from users
where likerock is unknown
order by userid limit 10;

firstname | lastname | likerock
----------+----------+----------
Rafael    | Taylor   |
Vladimir  | Humphrey |
Barry     | Roy      |
Tamekah   | Juarez   |
Mufutau   | Watkins  |
Naida     | Calderon |
Anika     | Huff     |
Bruce     | Beck     |
Mallory   | Farrell  |
Scarlett  | Mayer    |
(10 rows
```

Exemplos com uma coluna TIME

A tabela de exemplo a seguir TIME_TEST tem uma coluna TIME_VAL (tipo TIME) com três valores inseridos.

```
select time_val from time_test;

time_val
---------------------
20:00:00
00:00:00.5550
00:58:00
```

O exemplo a seguir extrai as horas de cada timetz_val.

```
select time_val from time_test where time_val < '3:00';
   time_val
---------------
 00:00:00.5550
 00:58:00
```

O exemplo a seguir compara dois literais de tempo.

```
select time '18:25:33.123456' = time '18:25:33.123456';
 ?column?
----------
 t
```

Exemplos com uma coluna TIMETZ

A tabela de exemplo a seguir TIMETZ_TEST tem uma coluna TIMETZ_VAL (tipo TIMETZ) com três valores inseridos.

```
select timetz_val from timetz_test;

timetz_val
------------------
04:00:00+00
00:00:00.5550+00
05:58:00+00
```

O exemplo a seguir seleciona apenas os valores TIMETZ menores que `3:00:00 UTC`. A comparação é feita depois de converter o valor para UTC.

```
select timetz_val from timetz_test where timetz_val < '3:00:00 UTC';

   timetz_val
---------------
 00:00:00.5550+00
```

O seguinte exemplo compara dois literais TIMETZ. O fuso horário é ignorado para a comparação.

```
select time '18:25:33.123456 PST' < time '19:25:33.123456 EST';

 ?column?
----------
 t
```

Condições lógicas

As condições lógicas combinam o resultado de duas condições para produzir um único resultado. Todas as condições lógicas são operadores binários com um tipo de retorno booleano.

Sintaxe

```
expression
{ AND | OR }
expression
NOT expression
```

As condições lógicas usam uma lógica booleana de três valores onde o valor nulo representa uma relação desconhecida. A tabela a seguir descreve os resultados para condições lógicas, onde `E1` e `E2` representam expressões:

E1	E2	E1 E E2	E1 OU E2	NÃO E2
TRUE	TRUE	TRUE	TRUE	FALSE
TRUE	FALSE	FALSE	TRUE	TRUE
TRUE	UNKNOWN	UNKNOWN	TRUE	UNKNOWN

E1	E2	E1 E E2	E1 OU E2	NÃO E2
FALSE	TRUE	FALSE	TRUE	
FALSE	FALSE	FALSE	FALSE	
FALSE	UNKNOWN	FALSE	UNKNOWN	
UNKNOWN	TRUE	UNKNOWN	TRUE	
UNKNOWN	FALSE	FALSE	UNKNOWN	
UNKNOWN	UNKNOWN	UNKNOWN	UNKNOWN	

O operador NOT é avaliado antes de AND e o operador AND é avaliado antes do operador OR. O uso de qualquer parênteses pode cancelar esta ordem de avaliação padrão.

Exemplos

O seguinte exemplo retorna o USERID e USERNAME da tabela USERS onde o usuário gosta de Las Vegas e de esportes:

```
select userid, username from users
where likevegas = 1 and likesports = 1
order by userid;

userid | username
--------+----------
1 | JSG99FHE
67 | TWU10MZT
87 | DUF19VXU
92 | HYP36WEQ
109 | FPL38HZK
120 | DMJ24GUZ
123 | QZR22XGQ
130 | ZQC82ALK
133 | LBN45WCH
144 | UCX04JKN
165 | TEY68OEB
169 | AYQ83HGO
184 | TVX65AZX
...
```

```
(2128 rows)
```

O próximo exemplo retorna o USERID e USERNAME da tabela USERS onde o usuário gosta de Las Vegas, de esportes ou de ambos. Essa consulta retorna todas as saídas do exemplo anterior, mais os usuários que gostam somente de Las Vegas ou de esportes.

```
select userid, username from users
where likevegas = 1 or likesports = 1
order by userid;

userid | username
--------+----------
1 | JSG99FHE
2 | PGL08LJI
3 | IFT66TXU
5 | AEB55QTM
6 | NDQ15VBM
9 | MSD36KVR
10 | WKW41AIW
13 | QTF33MCG
15 | OWU78MTR
16 | ZMG93CDD
22 | RHT62AGI
27 | KOY02CVE
29 | HUH27PKK
...
(18968 rows)
```

A seguinte consulta usa parênteses em torno da condição 0R para encontrar locais em Nova Iorque ou na Califórnia onde Macbeth foi apresentada:

```
select distinct venuename, venuecity
from venue join event on venue.venueid=event.venueid
where (venuestate = 'NY' or venuestate = 'CA') and eventname='Macbeth'
order by 2,1;

venuename                |   venuecity
-----------------------------------------+---------------
Geffen Playhouse                         | Los Angeles
Greek Theatre                            | Los Angeles
Royce Hall                               | Los Angeles
American Airlines Theatre                | New York City
```

```
August Wilson Theatre                    | New York City
Belasco Theatre                          | New York City
Bernard B. Jacobs Theatre                | New York City
...
```

A remoção dos parênteses neste exemplo altera a lógica e os resultados da consulta.

O seguinte exemplo usa o operador NOT:

```
select * from category
where not catid=1
order by 1;

catid | catgroup |  catname  |                 catdesc
-------+----------+-----------+--------------------------------------------
2 | Sports   | NHL       | National Hockey League
3 | Sports   | NFL       | National Football League
4 | Sports   | NBA       | National Basketball Association
5 | Sports   | MLS       | Major League Soccer
...
```

O seguinte exemplo usa uma condição NOT seguida de uma condição AND:

```
select * from category
where (not catid=1) and catgroup='Sports'
order by catid;

catid | catgroup | catname |             catdesc
-------+----------+---------+----------------------------------
2 | Sports   | NHL     | National Hockey League
3 | Sports   | NFL     | National Football League
4 | Sports   | NBA     | National Basketball Association
5 | Sports   | MLS     | Major League Soccer
(4 rows)
```

Condições de correspondência de padrões

Tópicos

- LIKE
- SIMILAR TO
- Operadores POSIX

Um operador de correspondência de padrões busca uma string para um padrão especificado na expressão condicional e retorna verdadeiro ou falso, dependendo da localização ou não de uma correspondência. O Amazon Redshift usa três métodos para correspondência de padrão:

- Expressões LIKE

  O operador LIKE compara uma expressão de string, tal como um nome de coluna, a um padrão que usa os caracteres curinga `%` (por cento) e `_` (sublinhado). A correspondência de padrão LIKE sempre abrange toda a string. LIKE executa uma correspondência com diferenciação entre letras maiúsculas e minúsculas e ILIKE executa uma correspondência sem diferenciação entre maiúsculas e minúsculas.

- Expressões regulares SIMILAR TO

  O operador SIMILAR TO faz a correspondência de uma string com um padrão de expressão regular SQL, o que pode incluir um conjunto de metacaracteres de correspondência de padrão que inclui os dois compatíveis com o operador LIKE. SIMILAR TO corresponde toda a string e executa uma correspondência com diferenciação entre letras maiúsculas e minúsculas.

- Expressões regulares estilo POSIX

  As expressões regulares POSIX fornecem uma forma mais poderosa para a correspondência de padrão do que os operadores LIKE e SIMILAR TO. Os padrões de expressão regular POSIX podem fazer a correspondência de qualquer parte da string e executam uma correspondência com diferenciação entre maiúsculas e minúsculas.

A correspondência de expressões regular usando operadores SIMILAR TO e POSIX é computacionalmente cara. Recomendamos usar LIKE sempre que possível, especialmente ao processar um número muito grande de linhas. Por exemplo, as seguintes consultas são funcionalmente idênticas, mas a consulta que usa LIKE é executada várias vezes mais rápido do que a consulta que usa uma expressão regular:

```
select count(*) from event where eventname SIMILAR TO '%(Ring|Die)%';
select count(*) from event where eventname LIKE '%Ring%' OR eventname LIKE '%Die%';
```

LIKE

O operador LIKE compara uma expressão de string, tal como um nome de coluna, a um padrão que usa os caracteres curinga % (por cento) e _ (sublinhado). A correspondência de padrão LIKE sempre

abrange toda a string. Para corresponder uma sequência em qualquer lugar de uma string, o padrão deve começar e terminar com um sinal de por cento.

LIKE diferencia entre maiúsculas e minúsculas; ILIKE não diferencia entre maiúsculas e minúsculas.

Sintaxe

```
expression [ NOT ] LIKE | ILIKE pattern [ ESCAPE 'escape_char' ]
```

Argumentos

expressão

Uma expressão de caractere UTF-8 válida, tal como um nome de coluna.

LIKE | ILIKE

LIKE executa uma correspondência de padrão com diferenciação entre maiúsculas e minúsculas. ILIKE realiza uma correspondência de padrão sem diferenciação entre maiúsculas e minúsculas para caracteres de byte único UTF-8 (ASCII). Para realizar uma correspondência de padrões sem distinção entre maiúsculas e minúsculas para caracteres de vários bytes, use a função LOWER em expressão e padrão com uma condição LIKE.

Em contraste com predicados de comparação, como = e <>, predicados LIKE e ILIKE não ignoram implicitamente os espaços à direita. Para ignorar espaços à direita, use RTRIM ou converta explicitamente uma coluna CHAR em VARCHAR.

O operador ~~ é equivalente a LIKE, e ~~* é equivalente a ILIKE. Além disso, os operadores !~~ e !~~* são equivalentes a NOT LIKE e NOT ILIKE.

Padrão

Uma expressão de caractere UTF-8 válida com o padrão a ser correspondido.

escape_char

Uma expressão de caractere que irá escapar caracteres de metacaracteres no padrão. O padrão é duas barras invertidas ('\\').

Se o padrão não contém metacaracteres, o padrão representa somente a própria string; nesse caso, LIKE age da mesma forma que o operador de igualdade.

Qualquer uma das expressões de caracteres pode ser de tipos de dados CHAR ou VARCHAR. Se eles forem diferentes, o Amazon Redshift converterá o padrão no tipo de dados da expressão.

LIKE é compatível com os seguintes metacaracteres de correspondência de padrão:

Operador	Descrição
%	Corresponde a qualquer sequência de zero ou mais caracteres.
_	Corresponde a qualquer caractere único.

Exemplos

A seguinte tabela mostra exemplos de correspondência de padrão usando LIKE:

Expressão	Retornos
`'abc' LIKE 'abc'`	Verdadeiro
`'abc' LIKE 'a%'`	Verdadeiro
`'abc' LIKE '_B_'`	Falso
`'abc' ILIKE '_B_'`	Verdadeiro
`'abc' LIKE 'c%'`	Falso

O seguinte exemplo localiza todas as cidades cujos nomes começam com “E”:

```
select distinct city from users
where city like 'E%' order by city;
city
---------------
East Hartford
East Lansing
East Rutherford
East St. Louis
Easthampton
Easton
```

```
Eatontown
Eau Claire
...
```

O seguinte exemplo localiza usuários cujos sobrenomes contêm “ten”:

```
select distinct lastname from users
where lastname like '%ten%' order by lastname;
lastname
-------------
Christensen
Wooten
...
```

O exemplo a seguir demonstra como combinar vários padrões.

```
select distinct lastname from tickit.users
where lastname like 'Chris%' or lastname like '%Wooten' order by lastname;
lastname
-------------
Christensen
Christian
Wooten
...
```

O seguinte exemplo localiza as cidades cujos terceiros e quartos caracteres são “ea”. O comando usa ILIKE para demonstrar a não diferenciação entre maiúsculas e minúsculas:

```
select distinct city from users where city ilike '__EA%' order by city;
city
-------------
Brea
Clearwater
Great Falls
Ocean City
Olean
Wheaton
(6 rows)
```

O seguinte exemplo usa a string de escape padrão (\\) para pesquisar strings que incluem “start_” (o texto `start` seguido por um sublinhado _):

```
select tablename, "column" from pg_table_def
where "column" like '%start\\_%'
limit 5;

     tablename     |    column
-------------------+---------------
 stl_s3client      | start_time
 stl_tr_conflict   | xact_start_ts
 stl_undone        | undo_start_ts
 stl_unload_log    | start_time
 stl_vacuum_detail | start_row
(5 rows)
```

O seguinte exemplo especifica “^” como caractere de escape e o utiliza para pesquisar strings que incluem “start_” (o texto `start` seguido por um sublinhado _):

```
select tablename, "column" from pg_table_def
where "column" like '%start^_%' escape '^'
limit 5;

     tablename     |    column
-------------------+---------------
 stl_s3client      | start_time
 stl_tr_conflict   | xact_start_ts
 stl_undone        | undo_start_ts
 stl_unload_log    | start_time
 stl_vacuum_detail | start_row
(5 rows)
```

O exemplo a seguir usa o operador ~~* para fazer uma pesquisa sem distinção entre maiúsculas e minúsculas (ILIKE) de cidades que começam com “Ag”.

```
select distinct city from users where city ~~* 'Ag%' order by city;

city
------------
Agat
Agawam
Agoura Hills
Aguadilla
```

SIMILAR TO

O operador SIMILAR TO faz a correspondências de uma expressão de string, tal como um nome de coluna, com um padrão de expressão regular SQL. Um padrão de expressão regular SQL pode incluir um conjunto de metacaracteres de correspondência padrão, incluindo os dois compatíveis com o operador LIKE.

O operador SIMILAR TO retorna verdadeiro somente se seu padrão corresponder à string inteira, ao contrário do comportamento de expressão regular POSIX, onde o padrão pode corresponder a qualquer parte da string.

SIMILAR TO executa uma correspondência com diferenciação entre maiúsculas e minúsculas.

> **Note**
>
> A correspondência de expressões regular usando SIMILAR TO é computacionalmente cara. Recomendamos usar LIKE sempre que possível, especialmente ao processar um número muito grande de linhas. Por exemplo, as seguintes consultas são funcionalmente idênticas, mas a consulta que usa LIKE é executada várias vezes mais rápido do que a consulta que usa uma expressão regular:
>
> ```
> select count(*) from event where eventname SIMILAR TO '%(Ring|Die)%';
> select count(*) from event where eventname LIKE '%Ring%' OR eventname LIKE '%Die
> %';
> ```

Sintaxe

```
expression [ NOT ] SIMILAR TO pattern [ ESCAPE 'escape_char' ]
```

Argumentos

expressão

Uma expressão de caractere UTF-8 válida, tal como um nome de coluna.

SIMILAR TO

SIMILAR to executa uma correspondência de padrão com diferenciação entre maiúsculas e minúsculas para toda a string na expressão.

Padrão

Uma expressão válida de caractere UTF-8 que representa um padrão de expressão regular SQL.

escape_char

Uma expressão de caractere que irá escapar metacaracteres no padrão. O padrão é duas barras invertidas ('\\').

Se o padrão não contém metacaracteres, o padrão representa somente a própria string.

Qualquer uma das expressões de caracteres pode ser de tipos de dados CHAR ou VARCHAR. Se eles forem diferentes, o Amazon Redshift converterá o padrão no tipo de dados da expressão.

SIMILAR TO é compatível com os seguintes metacaracteres de correspondência de padrão:

Operador	Descrição
`%`	Corresponde a qualquer sequência de zero ou mais caracteres.
`_`	Corresponde a qualquer caractere único.
`|`	Denota a alternação (uma das duas alternativas).
`*`	Repita o item anterior zero ou mais vezes.
`+`	Repita o item anterior uma ou mais vezes.
`?`	Repita o item anterior zero ou uma vez.
`{m}`	Repita o item anterior exatamente m vezes.
`{m,}`	Repita o item anterior m ou mais vezes.
`{m,n}`	Repita o item anterior pelo menos m e não mais do que n vezes.
`()`	Parênteses agrupam itens em um único item lógico.
`[...]`	Uma expressão entre colchetes especifica uma classe de caractere, assim como em expressões regulares POSIX.

Exemplos

A seguinte tabela mostra exemplos de correspondência de padrão usando SIMILAR TO:

Expressão	Retornos
`'abc' SIMILAR TO 'abc'`	Verdadeiro
`'abc' SIMILAR TO '_b_'`	Verdadeiro
`'abc' SIMILAR TO '_A_'`	Falso
`'abc' SIMILAR TO '%(b|d)%'`	Verdadeiro
`'abc' SIMILAR TO '(b|c)%'`	Falso
`'AbcAbcdefgefg12efgefg12' SIMILAR TO '((Ab)?c)+d((efg)+(12))+'`	Verdadeiro
`'aaaaaab11111xy' SIMILAR TO 'a{6}_[0-9]{5}(x|y){2}'`	Verdadeiro
`'$0.87' SIMILAR TO '$[0-9]+(.[0-9][0-9])?'`	Verdadeiro

O seguinte exemplo encontra cidades cujos nomes contêm "E" ou "H":

```
SELECT DISTINCT city FROM users
WHERE city SIMILAR TO '%E%|%H%' ORDER BY city LIMIT 5;

      city
-----------------
 Agoura Hills
 Auburn Hills
 Benton Harbor
 Beverly Hills
 Chicago Heights
```

O seguinte exemplo usa a string de escape padrão ('\\') para pesquisar strings que contêm "_":

```
SELECT tablename, "column" FROM pg_table_def
WHERE "column" SIMILAR TO '%start\\_%'
ORDER BY tablename, "column" LIMIT 5;

        tablename         |       column
--------------------------+---------------------
 stcs_abort_idle          | idle_start_time
 stcs_abort_idle          | txn_start_time
 stcs_analyze_compression | start_time
 stcs_auto_worker_levels  | start_level
 stcs_auto_worker_levels  | start_wlm_occupancy
```

O seguinte exemplo especifica '^' como a string de escape e, então, usa a string de escape para pesquisar strings que contêm "_":

```
SELECT tablename, "column" FROM pg_table_def
WHERE "column" SIMILAR TO '%start^_%' ESCAPE '^'
ORDER BY tablename, "column" LIMIT 5;

        tablename         |       column
--------------------------+---------------------
 stcs_abort_idle          | idle_start_time
 stcs_abort_idle          | txn_start_time
 stcs_analyze_compression | start_time
 stcs_auto_worker_levels  | start_level
 stcs_auto_worker_levels  | start_wlm_occupancy
```

Operadores POSIX

Uma expressão regular POSIX é uma sequência de caracteres que especifica um padrão de correspondência. Uma string estabelece correspondência com uma expressão regular se for membro do conjunto regular descrito pela expressão regular.

As expressões regulares POSIX fornecem uma forma mais poderosa para a correspondência de padrão do que os operadores LIKE e SIMILAR TO. Padrões de expressão regular POSIX podem corresponder a qualquer porção de uma string, diferente do operador SIMILAR TO, que retorna verdadeiro somente se seu padrão corresponder a toda a string.

> **Note**
>
> A correspondência de expressões regular usando operadores POSIX é computacionalmente cara. Recomendamos usar LIKE sempre que possível, especialmente ao processar um número muito grande de linhas. Por exemplo, as seguintes consultas são funcionalmente idênticas, mas a consulta que usa LIKE é executada várias vezes mais rápido do que a consulta que usa uma expressão regular:
>
> ```
> select count(*) from event where eventname ~ '.*(Ring|Die).*';
> select count(*) from event where eventname LIKE '%Ring%' OR eventname LIKE '%Die
> %';
> ```

Sintaxe

```
expression [ ! ] ~ pattern
```

Argumentos

expressão

Uma expressão de caractere UTF-8 válida, tal como um nome de coluna.

!

Operador de negação. Não estabelece correspondência com a expressão regular.

~

Execute uma correspondência com diferenciação entre maiúsculas e minúsculas para qualquer substring de expressão.

> **Note**
>
> ~~ é sinônimo de LIKE.

Padrão

Um literal de string que representa um padrão de expressão regular.

Se o padrão não contém caracteres curinga, o padrão representa somente a própria string.

Para pesquisar strings que contêm metacaracteres, tal como '`. * | ?` ', e assim por diante, escapam o caractere usando duas barras invertidas (' `\\` '). Ao contrário de `SIMILAR TO` e `LIKE`, a sintaxe de expressão regular POSIX não é compatível com um caractere de escape definido pelo usuário.

Qualquer uma das expressões de caracteres pode ser de tipos de dados CHAR ou VARCHAR. Se eles forem diferentes, o Amazon Redshift converterá o padrão no tipo de dados da expressão.

Todas as expressões de caracteres podem ser de tipos de dados CHAR ou VARCHAR. Se as expressões forem diferentes no tipo de dados, o Amazon Redshift os converterá no tipo de dados da expressão.

A correspondência de padrão POSIX é compatível com os seguintes metacaracteres:

POSIX	Descrição
.	Corresponde a qualquer caractere único.
*	Corresponde a zero ou mais ocorrências.
+	Corresponde a uma ou mais ocorrências.
?	Corresponde a zero ou uma ocorrência.
|	Especifica correspondências alternativas; por exemplo, `E | H` significa `E` ou `H`.
^	Corresponde ao caractere de início de linha.
$	Corresponde ao caractere de final de linha.
$	Corresponde ao final da string.
[ ]	Colchetes especificam uma lista de correspondência, que deve correspon der a uma expressão na lista. Um acento circunflexo (^) precede uma lista não correspondência, que corresponde a qualquer caractere exceto para as expressões representadas na lista.
( )	Parênteses agrupam itens em um único item lógico.

POSIX	Descrição
`{m}`	Repita o item anterior exatamente m vezes.
`{m,}`	Repita o item anterior m ou mais vezes.
`{m,n}`	Repita o item anterior pelo menos m e não mais do que n vezes.
`[: :]`	Corresponde a qualquer caractere em uma classe de caracteres POSIX. Nas seguintes classes de caracteres, o Amazon Redshift oferece suporte apenas a caracteres ASCII: , , e: `[:alnum:]` , `[:alpha:]` , `[:lower:]` , `[:upper:]`

O Amazon Redshift oferece suporte às classes de caracteres POSIX a seguir.

Classe do caractere	Descrição
`[[:alnum:]]`	Todos os caracteres ASCII alfanuméricos
`[[:alpha:]]`	Todos os caracteres ASCII alfabéticos
`[[:blank:]]`	Todos os caracteres de espaço em branco
`[[:cntrl:]]`	Todos os caracteres de controle (não imprimíveis)
`[[:digit:]]`	Todos os dígitos numéricos
`[[:lower:]]`	Todos os caracteres ASCII alfabéticos em letras minúsculas
`[[:punct:]]`	Todos os caracteres de pontuação
`[[:space:]]`	Todos os caracteres de espaço (não imprimíveis)
`[[:upper:]]`	Todos os caracteres ASCII alfabéticos em letras maiúsculas
`[[:xdigit:]]`	Todos os caracteres hexadecimais válidos

O Amazon Redshift oferece suporte aos operadores influenciados por Perl em expressões regulares a seguir. Escape o operador usando duas barras invertidas (‘\\’).

Operador	Descrição	Expressão equivalente da classe do caractere
\\d	Um caractere de dígitos	[[:digit:]]
\\D	Um caractere não dígito	[^[:digit:]]
\\w	Um caractere de palavra	[[:word:]]
\\W	Um caractere não palavra	[^[:word:]]
\\s	Um caractere de espaço em branco	[[:space:]]
\\S	Um caractere de espaço não em branco	[^[:space:]]
\\b	Uma palavra de limite	

Exemplos

A seguinte tabela mostra exemplos de correspondência de padrão usando operadores POSIX:

Expressão	Retornos
'abc' ~ 'abc'	Verdadeiro
'abc' ~ 'a'	Verdadeiro
'abc' ~ 'A'	Falso
'abc' ~ '.*(b|d).*'	Verdadeiro
'abc' ~ '(b|c).*'	Verdadeiro
'AbcAbcdefgefg12efgefg12' ~ '((Ab)?c)+d((efg)+(12))+'	Verdadeiro

Expressão	Retornos
`'aaaaaab11111xy' ~ 'a{6}.[1]{5}(x|y){2}'`	Verdadeiro
`'$0.87' ~ '\\$[0-9]+(\\.[0-9][0-9])?'`	Verdadeiro
`'ab c' ~ '[[:space:]]'`	Verdadeiro
`'ab c' ~ '\\s'`	Verdadeiro
`' ' ~ '\\S'`	Falso

O seguinte exemplo localiza cidades cujos nomes contêm E ou H:

```
SELECT DISTINCT city FROM users
WHERE city ~ '.*E.*|.*H.*' ORDER BY city LIMIT 5;

      city
-----------------
 Agoura Hills
 Auburn Hills
 Benton Harbor
 Beverly Hills
 Chicago Heights
```

O exemplo a seguir encontra cidades cujos nomes não contêm E ou H:

```
SELECT DISTINCT city FROM users WHERE city !~ '.*E.*|.*H.*' ORDER BY city LIMIT 5;

      city
----------------
 Aberdeen
 Abilene
 Ada
 Agat
 Agawam
```

O seguinte exemplo usa a string de escape padrão ('\\') para pesquisar strings que contêm um período.

```
SELECT venuename FROM venue
WHERE venuename ~ '.*\\..*'
ORDER BY venueid;

          venuename
------------------------------
 St. Pete Times Forum
 Jobing.com Arena
 Hubert H. Humphrey Metrodome
 U.S. Cellular Field
 Superpages.com Center
 E.J. Nutter Center
 Bernard B. Jacobs Theatre
 St. James Theatre
```

Condição de intervalo BETWEEN

Uma condição BETWEEN testa expressões para a inclusão em um intervalo de valores, usando as palavras chave BETWEEN e AND.

Sintaxe

```
expression [ NOT ] BETWEEN expression AND expression
```

As expressões podem ser numéricas, caracteres ou tipos de dados de data e hora, mas elas devem ser compatíveis. O intervalo é inclusivo.

Exemplos

O primeiro exemplo conta quantas transações registraram vendas de 2, 3 ou 4 ingressos:

```
select count(*) from sales
where qtysold between 2 and 4;

count
--------
104021
(1 row)
```

A condição de intervalo inclui os valores de começo e de término.

```
select min(dateid), max(dateid) from sales
where dateid between 1900 and 1910;

min  | max
-----+-----
1900 | 1910
```

A primeira expressão em uma condição de intervalo deve ser o menor valor e a segunda expressão o maior valor. O seguinte exemplo retornará sempre zero linhas devido aos valores das expressões:

```
select count(*) from sales
where qtysold between 4 and 2;

count
-------
0
(1 row)
```

Contudo, a aplicação do modificador NOT inverterá a lógica e produzirá uma contagem de todas as linhas:

```
select count(*) from sales
where qtysold not between 4 and 2;

count
--------
172456
(1 row)
```

A seguinte consulta retorna uma lista de locais com 20.000 a 50.000 assentos:

```
select venueid, venuename, venueseats from venue
where venueseats between 20000 and 50000
order by venueseats desc;

venueid |       venuename               | venueseats
---------+-------------------------------+------------
116 | Busch Stadium                 |      49660
106 | Rangers BallPark in Arlington |      49115
```

```
96 | Oriole Park at Camden Yards   |      48876
...
(22 rows)
```

O exemplo a seguir demonstra o uso de BETWEEN para valores de data:

```
select salesid, qtysold, pricepaid, commission, saletime
from sales
where eventid between 1000 and 2000
   and saletime between '2008-01-01' and '2008-01-03'
order by saletime asc;

salesid | qtysold | pricepaid | commission |   saletime
--------+---------+-----------+------------+---------------
  65082 |       4 |       472 |       70.8 | 1/1/2008 06:06
 110917 |       1 |       337 |      50.55 | 1/1/2008 07:05
 112103 |       1 |       241 |      36.15 | 1/2/2008 03:15
 137882 |       3 |      1473 |     220.95 | 1/2/2008 05:18
  40331 |       2 |        58 |        8.7 | 1/2/2008 05:57
 110918 |       3 |      1011 |     151.65 | 1/2/2008 07:17
  96274 |       1 |       104 |       15.6 | 1/2/2008 07:18
 150499 |       3 |       135 |      20.25 | 1/2/2008 07:20
  68413 |       2 |       158 |       23.7 | 1/2/2008 08:12
```

Observe que, embora o intervalo de BETWEEN seja inclusivo, as datas têm como padrão um valor de hora de 00:00:00. A única linha válida de 3 de janeiro para a consulta de amostra seria uma linha com hora de venda de `1/3/2008 00:00:00`.

Condição null

A condição null testa quanto a nulos quando um valor está ausente ou é desconhecido.

Sintaxe

```
expression IS [ NOT ] NULL
```

Argumentos

expressão

Qualquer expressão, tal como uma coluna.

IS NULL

É verdadeiro quando o valor de expressão é nulo e falso quando ele tem um valor.

IS NOT NULL

É falso quando o valor de expressão é nulo e verdadeiro quando ele tem um valor.

Exemplo

Este exemplo indica quantas vezes a tabela SALES contém null no campo QTYSOLD:

```
select count(*) from sales
where qtysold is null;
count
-------
0
(1 row)
```

Condição EXISTS

As condições EXISTS testam a existência de linhas em uma subconsulta e retornam verdadeiro se uma subconsulta retornar pelo menos uma linha. Se NOT estiver especificado, a condição retorna verdadeiro se uma subconsulta não retornar qualquer linha.

Sintaxe

```
[ NOT ] EXISTS (table_subquery)
```

Argumentos

EXISTS

É verdadeiro quando table_subquery retorna pelo menos uma linha.

NOT EXISTS

É verdadeiro quando table_subquery não retorna qualquer linha.

table_subquery

Uma subconsulta que avalia em uma tabela com uma ou mais colunas e uma ou mais linhas.

Exemplo

Este exemplo retorna todos os identificadores de data, um de cada vez, para cada data teve uma venda de qualquer tipo:

```
select dateid from date
where exists (
select 1 from sales
where date.dateid = sales.dateid
)
order by dateid;

dateid
--------
1827
1828
1829
...
```

Condição IN

Uma condição IN testa um valor para associação em um conjunto de valores ou uma subconsulta.

Sintaxe

```
expression [ NOT ] IN (expr_list | table_subquery)
```

Argumentos

expressão

Uma expressão numérica, de caractere ou de data e hora que é avaliada em relação a expr_list ou table_subquery e deve ser compatível com o tipo de dados daquela lista ou subconsulta.

expr_list

Uma ou várias expressões delimitadas por vírgula ou um ou mais conjuntos de expressões delimitadas por vírgula entre parênteses.

table_subquery

Uma subconsulta que avalia em uma tabela com uma ou mais linhas, mas é limitada a somente uma coluna em sua lista de seleção.

IN | NOT IN

IN retorna verdadeiro se a expressão é um membro da lista de expressão ou consulta. NOT IN retorna verdadeiro se a expressão não é um membro. IN e NOT IN retornam null e nenhuma linha é retornada nos seguintes casos: Se a expressão resulta em nulo; ou se não há valores expr_list ou table_subquery correspondentes e pelo menos uma dessas linhas de comparação resulta em null.

Exemplos

As seguintes condições são verdadeiras somente para os valores listados:

```
qtysold in (2, 4, 5)
date.day in ('Mon', 'Tues')
date.month not in ('Oct', 'Nov', 'Dec')
```

Otimização para grandes listas IN

Para otimizar a performance da consulta, uma lista IN que inclua mais do que 10 valores é internamente avaliada como uma matriz escalar. Listas IN com menos do que 10 valores são avaliadas como uma série de predicados OR. Essa otimização é compatível com os tipos de dados SMALLINT, INTEGER, BIGINT, REAL, DOUBLE PRECISION, BOOLEAN, CHAR, VARCHAR, DATE, TIMESTAMP e TIMESTAMPTZ.

Observe a saída EXPLAIN para a consulta para visualizar o efeito desta otimização. Por exemplo:

```
explain select * from sales
QUERY PLAN
------------------------------------------------------------------------
XN Seq Scan on sales  (cost=0.00..6035.96 rows=86228 width=53)
Filter: (salesid = ANY ('{1,2,3,4,5,6,7,8,9,10,11}'::integer[]))
(2 rows)
```

Comandos SQL

A linguagem SQL consiste em comandos usados para criar e manipular objetos de banco de dados, executar consultas, carregar tabelas e modificar os dados nas tabelas.

O Amazon Redshift é baseado no PostgreSQL. O Amazon Redshift e o PostgreSQL têm inúmeras diferenças importantes das quais você deve estar ciente ao projetar e desenvolver suas aplicações

de data warehouse. Para mais informações sobre como o SQL do Amazon Redshift difere do PostgreSQL, consulte Amazon Redshift e PostgreSQL.

> **Note**
>
> O tamanho máximo de uma única instrução SQL é 16 MB.

Tópicos

- ABORT
- ALTER DATABASE
- ALTER DATASHARE
- ALTER DEFAULT PRIVILEGES
- ALTER EXTERNAL SCHEMA
- ALTER EXTERNAL VIEW
- ALTER FUNCTION
- ALTER GROUP
- ALTER IDENTITY PROVIDER
- ALTER MASKING POLICY
- ALTER MATERIALIZED VIEW
- ALTER RLS POLICY
- ALTER ROLE
- ALTER PROCEDURE
- ALTER SCHEMA
- ALTER SYSTEM
- ALTER TABLE
- ALTER TABLE APPEND
- ALTER TEMPLATE
- ALTER USER
- ANALYZE
- ANALYZE COMPRESSION

- ATTACH MASKING POLICY
- ATTACH RLS POLICY
- BEGIN
- CALL
- CANCEL
- CLOSE
- COMMENT
- COMMIT
- COPY
- CREATE DATABASE
- CREATE DATASHARE
- CREATE EXTERNAL FUNCTION
- CREATE EXTERNAL MODEL
- CREATE EXTERNAL SCHEMA
- CREATE EXTERNAL TABLE
- CREATE EXTERNAL VIEW
- CREATE FUNCTION
- CREATE GROUP
- CREATE IDENTITY PROVIDER
- CREATE LIBRARY
- CREATE MASKING POLICY
- CREATE MATERIALIZED VIEW
- CREATE MODEL
- CREATE PROCEDURE
- CREATE RLS POLICY
- CREATE ROLE
- CREATE SCHEMA
- CRIAR TABELA
- CREATE TABLE AS

- CREATE TEMPLATE
- CREATE USER
- CREATE VIEW
- DEALLOCATE
- DECLARE
- DELETE
- DESC DATASHARE
- DESC IDENTITY PROVIDER
- DETACH MASKING POLICY
- DETACH RLS POLICY
- DROP DATABASE
- DROP DATASHARE
- DROP EXTERNAL VIEW
- DROP FUNCTION
- DROP GROUP
- DROP IDENTITY PROVIDER
- DROP LIBRARY
- DROP MASKING POLICY
- DROP MODEL
- DROP MATERIALIZED VIEW
- DROP PROCEDURE
- DROP RLS POLICY
- DROP ROLE
- DROP SCHEMA
- DESCARTAR TABELA
- DROP
- DROP USER
- DROP VIEW
- END

- EXECUTE
- EXPLAIN
- FETCH
- GRANT
- INSERT
- INSERT (tabela externa)
- LOCK
- MERGE
- PREPARE
- REFRESH MATERIALIZED VIEW
- RESET
- REVOKE
- ROLLBACK
- SELECT
- SELECT INTO
- SET
- SET SESSION AUTHORIZATION
- SET SESSION CHARACTERISTICS
- SHOW
- SHOW USER LOCKOUT
- SHOW COLUMN GRANTS
- SHOW COLUMNS
- SHOW CONSTRAINTS
- SHOW EXTERNAL TABLE
- SHOW DATABASES
- SHOW FUNCTIONS
- SHOW GRANTS
- SHOW MODEL
- SHOW DATASHARES

- SHOW PARAMETERS
- SHOW POLICIES
- SHOW PROCEDURE
- SHOW PROCEDURES
- SHOW SCHEMAS
- SHOW TABLE
- SHOW TABLES
- SHOW TEMPLATE
- SHOW TEMPLATES
- SHOW VIEW
- START TRANSACTION
- TRUNCATE
- UNLOAD
- UPDATE
- USE
- VACUUM

ABORT

Interrompe a transação que está sendo executada e descarta todas as atualizações feitas por essa transação. ABORT não afeta transações já concluídas.

Este comando executa a mesma função que o comando ROLLBACK. Para mais informações, consulte ROLLBACK.

Sintaxe

```
ABORT [ WORK | TRANSACTION ]
```

Parâmetros

WORK

Palavra-chave opcional.

TRANSACTION

Palavra-chave opcional; WORK e TRANSACTION são sinônimos.

Exemplo

O exemplo a seguir cria uma tabela, depois inicia uma transação com a inserção de dados na tabela. O comando ABORT então reverte a inserção de dados para deixar a tabela vazia.

O comando a seguir cria uma tabela de exemplo denominada MOVIE_GROSS:

```
create table movie_gross( name varchar(30), gross bigint );
```

O próximo conjunto de comandos inicia uma transação que insere duas linhas de dados na tabela:

```
begin;

insert into movie_gross values ( 'Raiders of the Lost Ark', 23400000);

insert into movie_gross values ( 'Star Wars', 10000000 );
```

Depois, o comando seleciona os dados da tabela para mostrar que eles foram inseridos com êxito:

```
select * from movie_gross;
```

A saída do comando mostra que ambas as linhas foram inseridas com êxito:

```
         name           |  gross
------------------------+----------
Raiders of the Lost Ark | 23400000
Star Wars               | 10000000
(2 rows)
```

Agora este comando reverte as alterações de dados para onde a transação foi iniciada:

```
abort;
```

Selecionar dados na tabela agora exibe uma tabela vazia:

```
select * from movie_gross;

 name | gross
------+-------
(0 rows)
```

ALTER DATABASE

Altera os atributos de um banco de dados.

Privilégios obrigatórios

Para usar ALTER DATABASE, um dos privilégios a seguir é necessário.

- Superusuário
- Usuários com o privilégio ALTER DATABASE.
- Proprietário do banco de dados

Sintaxe

```
ALTER DATABASE database_name
{
  RENAME TO new_name
  | OWNER TO new_owner
  | [ CONNECTION LIMIT { limit | UNLIMITED } ]
    [ COLLATE { CASE_SENSITIVE | CS | CASE_INSENSITIVE | CI } ]
    [ ISOLATION LEVEL { SNAPSHOT | SERIALIZABLE } ]
| INTEGRATION
 {
  REFRESH { { ALL | INERROR | REMEDIABLE } TABLES [ IN SCHEMA schema [, ...] ] | TABLE
 schema.table [, ...] }
   | SET
     [ QUERY_ALL_STATES [=] { TRUE | FALSE } ]
     [ ACCEPTINVCHARS [=] { TRUE | FALSE } ]
     [ REFRESH_INTERVAL <interval> ]
     [ TRUNCATECOLUMNS [=] { TRUE | FALSE } ]
     [ HISTORY_MODE [=] {TRUE | FALSE} [ FOR { {ALL} TABLES [IN SCHEMA schema [, ...] ]
 | TABLE schema.table [, ...] } ] ]
     [ AUTO_REMEDIATION [=] { TRUE | FALSE } ]
 }
```

```
}
```

Parâmetros

database_name

Nome do banco de dados a ser alterado. Normalmente, você deve alterar um banco de dados ao qual não estiver conectado. De qualquer maneira, as alterações somente entram em vigor nas próximas sessões. É possível alterar o proprietário do banco de dados atual, mas não é possível renomeá-lo:

```
alter database tickit rename to newtickit;
ERROR:  current database may not be renamed
```

RENAME TO

Renomeia o banco de dados especificado. Para obter mais informações sobre nomes válidos, consulte Nomes e identificadores. Você não pode renomear os bancos de dados dev, padb_harvest, template0, template1 ou sys:internal, além do banco de dados atual. Somente o proprietário do banco de dados ou um superuser (p. 1124) pode renomear o banco de dados. Proprietários que não forem superusuários também devem ter o privilégio CREATEDB.

new_name

Novo nome do banco de dados.

OWNER TO

Altera o proprietário do banco de dados especificado. É possível alterar o proprietário do banco de dados atual ou de outro banco de dados. Somente um superusuário pode alterar o proprietário.

new_owner

Novo proprietário do banco de dados. O novo proprietário deve ser um usuário existente de bancos de dados com privilégios de gravação. Para obter mais informações sobre privilégios do usuário, consulte GRANT.

CONNECTION LIMIT { limite | UNLIMITED }

Número máximo de conexões de banco de dados que os usuários podem abrir simultaneamente. Não há aplicação de limite para superusuários. Use a palavra-chave UNLIMITED para permitir o número máximo de conexões simultâneas. Um limite no número de conexões para cada usuário

também pode ser aplicável. Para obter mais informações, consulte CREATE USER. O valor padrão é UNLIMITED. Para visualizar as conexões atuais, consulte a exibição STV_SESSIONS do sistema.

> **Note**
>
> Se limites de usuário e de conexão de banco de dados forem aplicáveis, um slot de conexão não utilizado que esteja dentro de ambos os limites deve estar disponível quando um usuário tenta se conectar.

COLLATE { CASE_SENSITIVE | CS | CASE_INSENSITIVE | CI }

Uma cláusula que especifica se a pesquisa ou comparação de string diferencia maiúsculas e minúsculas ou não.

É possível alterar a distinção entre maiúsculas e minúsculas do banco de dados atual se ele estiver vazio.

Você deve ter a permissão ALTER para o banco de dados atual para alterar a distinção entre maiúsculas e minúsculas. Os superusuários ou proprietários de bancos de dados com a permissão CREATE DATABASE também podem alterar a distinção.

CASE_SENSITIVE e CS são intercambiáveis e geram os mesmos resultados. Da mesma forma, CASE_INSENSITIVE e CI são intercambiáveis e geram os mesmos resultados.

Para verificar o agrupamento atual de um banco de dados, use a função DB_COLLATION.

ISOLATION LEVEL { SNAPSHOT | SERIALIZABLE }

Uma cláusula que especifica o nível de isolamento usado quando são executadas consultas em um banco de dados. Para acessar mais informações sobre níveis de isolamento, consulte Níveis de isolamento no Amazon Redshift.

- Isolamento SNAPSHOT: fornece um nível de isolamento com proteção contra conflitos de atualização e exclusão.
- Isolamento SERIALIZABLE: fornece serialização total para transações simultâneas.

Considere os seguintes itens ao alterar o nível de isolamento de um banco de dados:

- Você deve ter o privilégio de superusuário ou CREATE DATABASE para o banco de dados atual a fim de alterar o nível de isolamento do banco de dados.

- Você não pode alterar o nível de isolamento do banco de dados `dev`.
- Você não pode alterar o nível de isolamento em um bloco de transação.
- O comando para alterar o nível de isolamento falhará se outros usuários estiverem conectados ao banco de dados.
- O comando para alterar o nível de isolamento pode alterar as configurações de nível de isolamento da sessão atual.

INTEGRATION

Altere um banco de dados de integração ETL zero.

REFRESH {{ ALL | INERROR | REMEDIABLE } TABLES [IN SCHEMA schema [, ...]] | TABLE schema.table [, ...]}

Uma cláusula que especifica quais tabelas o Amazon Redshift atualiza. É possível especificar todas as tabelas, tabelas com erros ou tabelas afetadas por linhas duplicadas. A atualização replica completamente as tabelas do banco de dados de origem.

`REMEDIABLE` aplica-se a tabelas no estado `Synced` que foram afetadas por linhas duplicadas. É possível verificar quais tabelas foram afetadas consultando SVV_INTEGRATION_TABLE_STATE.

Consulte mais informações em Integrações ETL zero no Guia de gerenciamento do Amazon Redshift. Para obter mais informações sobre estados de integração, consulte SVV_INTEGRATION_TABLE_STATE e SVV_INTEGRATION.

QUERY_ALL_STATES [=] { TRUE | FALSE }

A cláusula QUERY_ALL_STATES define se as tabelas de integração ETL zero podem ser consultadas em todos os estados (`Synced`, `Failed`, `ResyncRequired` e `ResyncInitiated`). Por padrão, uma tabela de integração ETL zero só pode ser consultada no estado `Synced`.

ACCEPTINVCHARS [=] { TRUE | FALSE }

A cláusula ACCEPTINVCHARS define se as tabelas de integração ETL zero devem continuar com a ingestão quando caracteres inválidos são detectados para o tipo de dados VARCHAR. Quando um caractere inválido é encontrado, ele é substituído por um caractere padrão `?`.

REFRESH_INTERVAL <intervalo>

A cláusula REFRESH_INTERVAL define o intervalo aproximado, em segundos, que o Amazon Redshift aguarda após a conclusão de um ciclo de atualização para iniciar o próximo. Cada ciclo atualiza os dados da origem da integração ETL zero para o banco de dados de destino, aplicando todas as alterações acumuladas desde o término do ciclo anterior. O valor 0 inicia o próximo

ciclo assim que o anterior termina (replicação quase em tempo real). Um valor maior aumenta o intervalo entre os ciclos, reduzindo a sobrecarga das atualizações em troca de uma menor atualização imediata dos dados. O Amazon Redshift aguarda entre os ciclos somente quando a ingestão alcança o ritmo das alterações. Quando as alterações se acumulam mais rapidamente do que podem ser aplicadas, os ciclos são executados sem intervalo de espera.

O `interval` pode ser definido entre 0 e 432.000 segundos (5 dias) para integrações ETL zero cuja origem seja Aurora MySQL, Aurora PostgreSQL ou um dos mecanismos do RDS compatíveis. O valor padrão é 0. Para integrações ETL zero do Amazon DynamoDB, o `interval` pode ser definido entre 900 e 432.000 segundos (15 minutos a 5 dias), e o valor padrão é 900 segundos (15 minutos). O valor 0 não é compatível.

> **Note**
>
> Para integrações com alta intensidade de gravação, exceto as do DynamoDB, que geram um grande volume de alterações, defina REFRESH_INTERVAL como um pequeno valor diferente de zero (por exemplo, 60 a 120 segundos). Agrupar mais alterações em cada ciclo reduz a sobrecarga geral de computação da replicação.

Consulte mais informações sobre a criação de bancos de dados com integrações ETL zero em Criar bancos de dados de destino no Amazon Redshift no Guia de gerenciamento do Amazon Redshift.

TRUNCATECOLUMNS [=] { TRUE | FALSE }

A cláusula TRUNCATECOLUMNS define se as tabelas de Integração ETL zero devem continuar com a ingestão quando os valores dos atributos das colunas VARCHAR ou SUPER estão além do limite. Quando TRUE, os valores são truncados para caber na coluna e os valores dos atributos JSON em excesso são truncados para caber na coluna SUPER.

HISTORY_MODE [=] {TRUE | FALSE} [FOR { {ALL} TABLES [IN SCHEMA schema [, ...]] | TABLE schema.table [, ...]}]

Uma cláusula que especifica se o Amazon Redshift definirá o modo histórico para todas as tabelas ou para tabelas no esquema especificado que participam da integração ETL zero. Essa opção é aplicável somente para bancos de dados criados para integração ETL zero.

A cláusula HISTORY_MODE pode ser definida como TRUE ou FALSE. O padrão é FALSE. Ativar e desativar o modo histórico só é aplicável às tabelas que estão no estado Synced. Consulte

informações sobre HISTORY_MODE em History mode no Guia de gerenciamento do Amazon Redshift.

AUTO_REMEDIATION [=] { TRUE | FALSE }

Especifica se o Amazon Redshift deve sincronizar novamente de forma automática as tabelas afetadas por linhas duplicadas. Quando definido como TRUE, o Amazon Redshift marca as tabelas afetadas para ressincronização sem exigir intervenção manual. O padrão é FALSE.

É possível monitorar quais tabelas foram sinalizadas consultando SVV_INTEGRATION_TABLE_STATE. A configuração atual fica visível na coluna `auto_remediation` de SVV_INTEGRATION.

Observações de uso

Comandos ALTER DATABASE são aplicáveis a sessões subsequentes, e não às atuais. Você precisa se reconectar ao banco de dados alterado para visualizar as alterações implementadas.

Exemplos

O exemplo a seguir renomeia o banco de dados denominado TICKIT_SANDBOX para TICKIT_TEST:

```
alter database tickit_sandbox rename to tickit_test;
```

O exemplo a seguir altera o proprietário do banco de dados TICKIT (o banco de dados atual) para DWUSER:

```
alter database tickit owner to dwuser;
```

O seguinte exemplo altera a distinção entre maiúsculas de minúsculas do banco de dados sampledb:

```
ALTER DATABASE sampledb COLLATE CASE_INSENSITIVE;
```

O exemplo a seguir altera um banco de dados chamado **`sampledb`** com o nível de isolamento SNAPSHOT.

```
ALTER DATABASE sampledb ISOLATION LEVEL SNAPSHOT;
```

O exemplo a seguir atualiza as tabelas **schema1.sample_table1** e **schema2.sample_table2** no banco de dados **sample_integration_db** na integração ETL zero.

```
ALTER DATABASE sample_integration_db INTEGRATION REFRESH TABLE schema1.sample_table1,
 schema2.sample_table2;
```

O exemplo a seguir atualiza todas as tabelas sincronizadas e com falha na integração ETL zero.

```
ALTER DATABASE sample_integration_db INTEGRATION REFRESH ALL tables;
```

O exemplo a seguir define o intervalo de atualização para integrações ETL zero como 600 segundos.

```
ALTER DATABASE sample_integration_db INTEGRATION SET REFRESH_INTERVAL 600;
```

O exemplo a seguir atualiza todas as tabelas presentes em `ErrorState` no esquema **sample_schema**.

```
ALTER DATABASE sample_integration_db INTEGRATION REFRESH INERROR TABLES in SCHEMA
 sample_schema;
```

O exemplo a seguir ativa o modo histórico para a tabela `myschema.table1`.

```
ALTER DATABASE sample_integration_db INTEGRATION SET HISTORY_MODE = true FOR TABLE
 myschema.table1
```

O exemplo a seguir ativa o modo histórico para todas as tabelas em `myschema`.

```
ALTER DATABASE sample_integration_db INTEGRATION SET HISTORY_MODE = true for ALL TABLES
 IN SCHEMA myschema
```

O exemplo a seguir habilita a correção automática de linhas duplicadas para um banco de dados de uma integração ETL zero.

```
ALTER DATABASE sample_integration_db INTEGRATION SET AUTO_REMEDIATION = true;
```

O exemplo a seguir atualiza todas as tabelas afetadas por linhas duplicadas no banco de dados da integração ETL zero.

```
ALTER DATABASE sample_integration_db INTEGRATION REFRESH REMEDIABLE TABLES;
```

O exemplo a seguir atualiza as tabelas afetadas por linhas duplicadas no esquema `myschema`.

```
ALTER DATABASE sample_integration_db INTEGRATION REFRESH REMEDIABLE TABLES IN SCHEMA
 myschema;
```

ALTER DATASHARE

Altera a definição de um datashare. Você pode adicionar ou remover objetos usando ALTER DATASHARE. Você só pode alterar uma unidade de compartilhamento de dados no banco de dados atual. Adicione ou remova objetos do banco de dados associado a uma unidade de compartilhamento de dados. O proprietário do datashare com as permissões necessárias nos objetos de datashare a serem adicionados ou removidos pode alterar o datashare.

Privilégios obrigatórios

A seguir estão os privilégios obrigatórios para ALTER DATASHARE:

- Superusuário.
- Usuário com o privilégio ALTER DATASHARE.
- Usuários que tenham privilégios ALTER ou ALL na unidade de compartilhamento de dados.
- Para adicionar objetos específicos a uma unidade de compartilhamento de dados, os usuários devem ter o privilégio nos objetos. Neste caso, os usuários devem ser os proprietários de objetos ou ter privilégios SELECT, USAGE ou ALL nos objetos.

Sintaxe

A sintaxe a seguir ilustra como adicionar ou remover objetos da unidade de compartilhamento de dados.

```
ALTER DATASHARE datashare_name { ADD | REMOVE } {
TABLE schema.table [, ...]
| SCHEMA schema [, ...]
| FUNCTION schema.sql_udf (argtype,...) [, ...]
| ALL TABLES IN SCHEMA schema [, ...]
| ALL FUNCTIONS IN SCHEMA schema [, ...] }
```

A sintaxe a seguir ilustra como configurar as propriedades da unidade de compartilhamento de dados.

```
ALTER DATASHARE datashare_name {
[ SET PUBLICACCESSIBLE [=] TRUE | FALSE ]
[ SET INCLUDENEW [=] TRUE | FALSE FOR SCHEMA schema ] }
```

Parâmetros

datashare_name

Nome do datashare a ser alterado.

ADD | REMOVE

Uma cláusula que especifica se deve adicionar ou remover objetos do datashare.

TABLE schema.table [, ...]

O nome da tabela ou da exibição no esquema especificado a ser adicionado ao datashare.

SCHEMA schema [, ...]

O nome do esquema a ser adicionado ao datashare.

FUNCTION schema.sql_udf (argtype,...) [, ...]

O nome da função SQL definida pelo usuário a ser adicionada à unidade de compartilhamento de dados.

ALL TABLES IN SCHEMA schema [, ...]

Uma cláusula que especifica se deve adicionar todas as tabelas e exibições no esquema especificado ao datashare.

ALL FUNCTIONS IN SCHEMA schema [, ...] }

Uma cláusula que especifica a adição de todas as funções no esquema especificado ao datashare.

[SET PUBLICACCESSIBLE [=] TRUE | FALSE]

Uma cláusula que especifica se um datashare pode ser compartilhado para clusters acessíveis publicamente.

[SET INCLUDENEW [=] TRUE | FALSE FOR SCHEMA schema]

Uma cláusula que especifica se deseja adicionar futuras tabelas, exibições ou funções definidas pelo usuário (UDFs) do SQL criadas no esquema especificado ao datashare. Tabelas atuais, exibições ou UDFs SQL no esquema especificado não são adicionadas ao datashare. Somente superusuários podem alterar essa propriedade para cada par datashare-schema. Por padrão, a cláusula INCLUDENEW está definida como falsa.

Observações de uso do ALTER DATASHARE

- Os seguintes usuários podem alterar um datashare:
  - Um superusuário
  - O proprietário do datashare
  - Usuários com privilégios ALTER ou ALL na unidade de compartilhamento de dados
- Para adicionar objetos específicos a uma unidade de compartilhamento de dados, esses usuários devem ter o privilégio correto nos objetos. Os usuários devem ser os proprietários de objetos ou ter privilégios SELECT, USAGE ou ALL nos objetos.
- Você pode compartilhar esquemas, tabelas, exibições regulares, exibições de vinculação tardia, visões materializadas e funções definidas pelo usuário (UDFs) do SQL. Adicione um esquema à unidade de compartilhamento de dados antes de adicionar outros objetos ao esquema.

  Quando você adiciona um esquema, o Amazon Redshift não adiciona todos os objetos abaixo dele. Você deve adicioná-las explicitamente.
- Recomendamos criar unidades de compartilhamento de dados AWS Data Exchange com a configuração publicamente acessível ativada.
- Em geral, recomendamos não alterar uma unidade de compartilhamento de dados AWS Data Exchange para desativar a acessibilidade pública usando a instrução ALTER DATASHARE. Se você fizer isso, as Contas da AWS que têm acesso à unidade de compartilhamento de dados perderão o acesso se os clusters forem acessíveis ao público. Executar esse tipo de alteração pode violar os termos do produto de dados no AWS Data Exchange. Veja a seguir uma exceção a essa recomendação.

  O exemplo a seguir mostra um erro quando uma unidade de compartilhamento de dados do AWS Data Exchange é criada com a configuração desativada.

```
ALTER DATASHARE salesshare SET PUBLICACCESSIBLE FALSE;
```

```
ERROR:  Alter of ADX-managed datashare salesshare requires session variable
 datashare_break_glass_session_var to be set to value 'c670ba4db22f4b'
```

Para permitir a alteração de uma unidade de compartilhamento de dados do AWS Data Exchange para desativar a configuração publicamente acessível, definir a variável a seguir e executar a instrução ALTER DATASHARE novamente.

```
SET datashare_break_glass_session_var to 'c670ba4db22f4b';
```

```
ALTER DATASHARE salesshare SET PUBLICACCESSIBLE FALSE;
```

Nesse caso, o Amazon Redshift gera um valor único aleatório para definir a variável de sessão para permitir ALTER DATASHARE SET PUBLICACCESSIBLE FALSE para uma unidade de compartilhamento de dados do AWS Data Exchange.

Exemplos

O exemplo a seguir adiciona o esquema `public` à unidade de compartilhamento de dados `salesshare`.

```
ALTER DATASHARE salesshare ADD SCHEMA public;
```

O exemplo a seguir adiciona a tabela `public.tickit_sales_redshift` à unidade de compartilhamento de dados `salesshare`.

```
ALTER DATASHARE salesshare ADD TABLE public.tickit_sales_redshift;
```

O exemplo a seguir adiciona todas as tabelas à unidade de compartilhamento de dados `salesshare`.

```
ALTER DATASHARE salesshare ADD ALL TABLES IN SCHEMA PUBLIC;
```

O exemplo a seguir remove a tabela `public.tickit_sales_redshift` da unidade de compartilhamento de dados `salesshare`.

```
ALTER DATASHARE salesshare REMOVE TABLE public.tickit_sales_redshift;
```

ALTER DEFAULT PRIVILEGES

Define o conjunto padrão de permissões de acesso que será aplicado a objetos criados no futuro pelo usuário especificado. Por padrão, os usuários podem alterar somente suas próprias permissões de acesso padrão. Somente um superusuário pode especificar permissões padrão para outros usuários.

Privilégios padrão podem ser aplicados a funções, usuários ou grupos de usuários. É possível definir permissões padrão globalmente para todos os objetos criados no banco de dados atual ou para os objetos criados somente nos esquemas especificados.

As permissões padrão se aplicam somente a novos objetos. Executar ALTER DEFAULT PRIVILEGES não altera as permissões em objetos existentes. Para conceder permissões em todos os objetos atuais e futuros criados por qualquer usuário em um banco de dados ou esquema, consulte Scoped permissions.

Para visualizar informações sobre os privilégios padrão para usuários de bancos de dados, consulte a PG_DEFAULT_ACL tabela de catálogos do sistema.

Para obter mais informações sobre privilégios, consulte GRANT.

Privilégios obrigatórios

A seguir estão os privilégios obrigatórios para ALTER DEFAULT PRIVILEGES:

- Superusuário
- Usuários com o privilégio ALTER DEFAULT PRIVILEGES
- Usuários que alterem seus próprios privilégios de acesso padrão
- Usuários que definam privilégios para esquemas aos quais eles têm privilégios de acesso

Sintaxe

```
ALTER DEFAULT PRIVILEGES
    [ FOR USER target_user [, ...] ]
    [ IN SCHEMA schema_name [, ...] ]
    grant_or_revoke_clause

where grant_or_revoke_clause is one of:
```

```
GRANT { { SELECT | INSERT | UPDATE | DELETE | DROP | REFERENCES | TRUNCATE } [,...] |
 ALL [ PRIVILEGES ] }
 ON TABLES
 TO { user_name [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

GRANT { EXECUTE | ALL [ PRIVILEGES ] }
 ON FUNCTIONS
 TO { user_name [ WITH GRANT OPTION ] |  ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

GRANT { EXECUTE | ALL [ PRIVILEGES ] }
 ON PROCEDURES
 TO { user_name [ WITH GRANT OPTION ] |  ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

REVOKE [ GRANT OPTION FOR ] { { SELECT | INSERT | UPDATE | DELETE | REFERENCES |
 TRUNCATE } [,...] | ALL [ PRIVILEGES ] }
 ON TABLES
 FROM user_name [, ...] [ RESTRICT ]

REVOKE  { { SELECT | INSERT | UPDATE | DELETE | REFERENCES | TRUNCATE } [,...] | ALL
 [ PRIVILEGES ] }
 ON TABLES
 FROM { ROLE role_name | GROUP group_name | PUBLIC } [, ...] [ RESTRICT ]

REVOKE [ GRANT OPTION FOR ] { EXECUTE | ALL [ PRIVILEGES ] }
 ON FUNCTIONS
 FROM user_name [, ...] [ RESTRICT ]

REVOKE { EXECUTE | ALL [ PRIVILEGES ] }
 ON FUNCTIONS
 FROM { ROLE role_name | GROUP group_name | PUBLIC } [, ...] [ RESTRICT ]

REVOKE [ GRANT OPTION FOR ] { EXECUTE | ALL [ PRIVILEGES ] }
 ON PROCEDURES
 FROM user_name [, ...] [ RESTRICT ]

REVOKE { EXECUTE | ALL [ PRIVILEGES ] }
 ON PROCEDURES
 FROM { ROLE role_name | GROUP group_name | PUBLIC } [, ...] [ RESTRICT ]
```

Parâmetros

FOR USER target_user

Opcional. Nome do usuário para o qual os privilégios padrão são definidos. Somente os superusuários podem especificar privilégios padrão para outros usuários. O valor padrão é o usuário atual.

IN SCHEMA schema_name

Opcional. Se uma cláusula IN SCHEMA for exibida, os privilégios padrão especificados são aplicados a novos objetos criados com o schema_name especificado. Nesse caso, o usuário ou grupo de usuários de destino do comando ALTER DEFAULT PRIVILEGES deve usar o comando CREATE para criar o privilégio para o esquema especificado. Os privilégios padrão específicos de um esquema são adicionados a privilégios padrão globais existentes. Por padrão, os privilégios padrão são aplicados globalmente ao banco de dados inteiro.

GRANT

O conjunto de privilégios a ser concedido a usuários ou grupos especificados para todas as novas tabelas, visualizações, funções ou procedimentos armazenados criados pelo usuário especificado. É possível definir os mesmos privilégios e opções com a cláusula GRANT que você pode definir com o comando GRANT.

WITH GRANT OPTION

Cláusula que indica que o usuário que recebe os privilégios por sua vez pode conceder os mesmos privilégios a outros. Você não pode conceder WITH GRANT OPTION a um grupo ou a PUBLIC.

TO user_name | ROLE role_name | GROUP group_name

Nome do usuário, da função ou do grupo de usuários ao qual os privilégios padrão especificados são aplicados.

REVOKE

Conjunto de privilégios a serem revogados dos usuários ou grupos especificados para todas as novas tabelas, funções ou procedimentos armazenados criados pelo usuário especificado. É possível definir os mesmos privilégios e opções com a cláusula REVOKE que você pode definir com o comando REVOKE.

GRANT OPTION FOR

Cláusula que revoga somente a opção de conceder um privilégio especificado a outros usuários e não revoga o próprio privilégio. Não é possível revogar GRANT OPTION de um grupo ou de PUBLIC.

FROM user_name | ROLE role_name | GROUP group_name

O nome do usuário, da função ou do grupo de usuários do qual os privilégios especificados são revogados por padrão.

RESTRICT

A opção RESTRICT revoga somente os privilégios que o usuário concedeu diretamente. Esse é o padrão.

Exemplos

Digamos que você deseja permitir que qualquer usuário do grupo `report_readers` visualize todas as tabelas e visualizações criadas pelo usuário `report_admin`. Nesse caso, execute o comando a seguir como superusuário.

```
alter default privileges for user report_admin grant select on tables to group
 report_readers;
```

No exemplo a seguir, o primeiro comando concede o privilégio SELECT a todas as novas tabelas que você criar. Sempre que criar uma visão, você deverá conceder explicitamente privilégios à visão ou executar novamente o comando `alter default privileges`.

```
alter default privileges grant select on tables to public;
```

O exemplo a seguir concede o privilégio INSERT ao grupo de usuários `sales_admin` para todas as novas tabelas e exibições que você criar no esquema `sales`.

```
alter default privileges in schema sales grant insert on tables to group sales_admin;
```

O exemplo a seguir inverte o comando ALTER DEFAULT PRIVILEGES do exemplo anterior.

```
alter default privileges in schema sales revoke insert on tables from group
 sales_admin;
```

Por padrão, o grupo de usuários PUBLIC tem permissão de execução para todas as novas funções definidas por usuário. Para revogar permissões de execução `public` para suas novas funções e depois conceder a permissão de execução somente para o grupo de usuários `dev_test`, execute os comandos a seguir.

```
alter default privileges revoke execute on functions from public;
alter default privileges grant execute on functions to group dev_test;
```

ALTER EXTERNAL SCHEMA

Altera um esquema externo existente no banco de dados atual. Somente proprietários de esquemas, superusuários ou usuários com privilégios ALTER no esquema podem alterá-lo. Somente esquemas externos criados no CATÁLOGO DE DADOS, KAFKA ou MSK podem ser alterados.

O proprietário deste esquema é o emissor do comando CREATE EXTERNAL SCHEMA. Para transferir a propriedade de um esquema externo, use ALTER SCHEMA para alterar o proprietário. Para conceder acesso ao esquema a outros usuários ou grupos de usuário, use o comando GRANT.

Você não pode usar os comandos GRANT ou REVOKE para permissões em uma tabela externa. Em vez disso, conceda ou revogue permissões no esquema externo.

Para saber mais, consulte:

- ALTER SCHEMA
- GRANT
- REVOKE
- CREATE EXTERNAL SCHEMA
- Habilitar a autenticação mTLS para um esquema externo existente

Para visualizar detalhes dos esquemas externos, consulte a visualização de sistema SVV_EXTERNAL_SCHEMAS. Para obter mais informações, consulte SVV_EXTERNAL_SCHEMAS.

Sintaxe

```
ALTER EXTERNAL SCHEMA schema_name
[ IAM_ROLE [ default | 'SESSION' | 'arn:aws:iam::<account-id>:role/<role-name>' ] ]
[ AUTHENTICATION [ none | iam | mtls] ]
```

```
[ AUTHENTICATION_ARN 'acm-certificate-arn' | SECRET_ARN 'asm-secret-arn' ]
[ URI 'Kafka bootstrap URL' ]
```

Se você tiver um esquema externo usado para ingestão de streaming e quiser implementar o TLS mútuo para autenticação, poderá executar um comando como o apresentado a seguir, que especifica a autenticação mTLS e o ARN do certificado do ACM no ACM.

```
ALTER EXTERNAL SCHEMA schema_name
AUTHENTICATION mtls
AUTHENTICATION_ARN 'arn:aws:acm:us-east-1:444455556666:certificate/certificate_ID';
```

Ou você pode especificar a autenticação mTLS, com referência ao ARN do segredo no Secrets Manager.

```
ALTER EXTERNAL SCHEMA schema_name
AUTHENTICATION mtls
SECRET_ARN 'arn:aws:secretsmanager:us-east-1:012345678910:secret:myMTLSSecret';
```

O seguinte exemplo mostra como modificar o URI para ALTER EXTERNAL SCHEMA:

```
ALTER EXTERNAL SCHEMA schema_name
URI 'lkc-ghidef-67890.centralus.azure.glb.confluent.cloud:9092';
```

O seguinte exemplo mostra como modificar o perfil do IAM para ALTER EXTERNAL SCHEMA:

```
ALTER EXTERNAL SCHEMA schema_name
IAM_ROLE 'arn:aws:iam::012345678901:role/testrole';
```

Parâmetros

IAM_ROLE[default | 'SESSION' | 'arn:aws:iam::<AWS account-id>:role/<role-name>']

Use a palavra-chave `default` para que o Amazon Redshift use o perfil do IAM que está definido como padrão.

Use `'SESSION'` se você se conectar ao cluster do Amazon Redshift usando uma identidade federada e acesse as tabelas do esquema externo criado usando esse comando.

Para ter mais informações, consulte CREATE EXTERNAL SCHEMA.

AUTHENTICATION

O tipo de autenticação definido para ingestão de streaming. A ingestão de streaming com tipos de autenticação funciona com o Apache, Kafka, Confluent Cloud e Amazon Managed Streaming for Apache Kafka. Para ter mais informações, consulte CREATE EXTERNAL SCHEMA.

AUTHENTICATION_ARN

O ARN do certificado do Gerenciador de certificados da AWS usado pelo Amazon Redshift para autenticação mTLS com Apache Kafka, Confluent Cloud ou Amazon Managed Streaming for Apache Kafka (Amazon MSK). O ARN está disponível no console do ACM quando você escolhe o certificado emitido.

SECRET_ARN

O nome do recurso da Amazon (ARN) de um segredo compatível criado por meio do AWS Secrets Manager. Para ter informações sobre como criar e recuperar um ARN de um segredo, consulte Manage secrets with AWS Secrets Manager no Guia do usuário do AWS Secrets Manager e Recuperar o nome do recurso da Amazon (ARN) do segredo no Amazon Redshift.

URI

O URL de bootstrap do cluster do Apache Kafka, Confluent Cloud ou Amazon Managed Streaming for Apache Kafka (Amazon MSK). O endpoint deve ser acessível (roteável) pelo cluster do Amazon Redshift. Para ter mais informações, consulte CREATE EXTERNAL SCHEMA.

ALTER EXTERNAL VIEW

Use o comando ALTER EXTERNAL VIEW para atualizar a exibição externa. Dependendo dos parâmetros usados, outros mecanismos SQL, como Amazon Athena e Amazon EMR Spark, que também podem referenciar essa exibição, podem ser afetados. Para obter mais informações sobre visualizações do Catálogo de Dados, consulte Visualizações do AWS Glue Data Catalog.

Sintaxe

```
ALTER EXTERNAL VIEW schema_name.view_name
{catalog_name.schema_name.view_name | awsdatacatalog.dbname.view_name |
 external_schema_name.view_name}
[FORCE] { AS (query_definition) | REMOVE DEFINITION }
```

Parâmetros

schema_name.view_name

O esquema anexado ao banco de dados do AWS Glue, seguido do nome da exibição.

catalog_name.schema_name.view_name | awsdatacatalog.dbname.view_name | external_schema_name.view_name

A notação do esquema a ser usado durante a alteração da exibição. Você pode especificar o uso do AWS Glue Data Catalog, um banco de dados do Glue criado por você, ou um esquema externo também criado por você. Consulte CREATE DATABASE e CREATE EXTERNAL SCHEMA para obter mais informações.

FORCE

Se AWS Lake Formation deve atualizar a definição da exibição mesmo que os objetos referenciados na tabela estejam inconsistentes com outros mecanismos SQL. Se o Lake Formation atualizar a exibição, esta será considerada obsoleta para os outros mecanismos SQL até esses mecanismos também serem atualizados.

AS query_definition

A definição da consulta SQL executada pelo Amazon Redshift para alterar a exibição.

REMOVE DEFINITION

Se é necessário descartar e recriar as exibições. As exibições devem ser descartadas e recriadas para marcá-las como PROTECTED.

Exemplos

O exemplo a seguir altera uma exibição do Data Catalog chamada sample_schema.glue_data_catalog_view.

```
ALTER EXTERNAL VIEW sample_schema.glue_data_catalog_view
FORCE
REMOVE DEFINITION
```

ALTER FUNCTION

Renomeia uma função ou altera o proprietário. Tanto o nome da função quanto os tipos de dados são obrigatórios. Somente o proprietário ou um superusuário pode renomear uma função. Somente um superusuário pode alterar o proprietário de uma função.

Sintaxe

```
ALTER FUNCTION function_name ( { [ py_arg_name py_arg_data_type | sql_arg_data_type }
 [ , ... ] ] )
    RENAME TO new_name
```

```
ALTER FUNCTION function_name ( { [ py_arg_name py_arg_data_type | sql_arg_data_type }
 [ , ... ] ] )
    OWNER TO { new_owner | CURRENT_USER | SESSION_USER }
```

Parâmetros

function_name

Nome da função a ser alterada. Especifique o nome da função no caminho de pesquisa atual ou use o formato `schema_name.function_name` para usar um esquema específico.

py_arg_name py_arg_data_type | sql_arg_data_type

Opcional. Uma lista de nomes de argumentos de entrada e tipos de dados para a função definida pelo usuário do Python ou uma lista de tipos de dados de argumentos de entrada para a função SQL definida pelo usuário.

new_name

Um novo nome da função definida pelo usuário.

new_owner | CURRENT_USER | SESSION_USER

Um novo proprietário da função definida pelo usuário.

Exemplos

O exemplo a seguir altera o nome de uma função de `first_quarter_revenue` para `quarterly_revenue`.

```
ALTER FUNCTION first_quarter_revenue(bigint, numeric, int)
         RENAME TO quarterly_revenue;
```

O exemplo a seguir altera o proprietário da função `quarterly_revenue` para `etl_user`.

```
ALTER FUNCTION quarterly_revenue(bigint, numeric) OWNER TO etl_user;
```

ALTER GROUP

Altera um grupo de usuários. Use este comando para adicionar usuários a um grupo, excluir usuários de um grupo ou renomear o grupo.

Sintaxe

```
ALTER GROUP group_name
{
ADD USER username [, ... ] |
DROP USER username [, ... ] |
RENAME TO new_name
}
```

Parâmetros

nome_grupo

Nome do grupo de usuários a ser modificado.

ADD

Adiciona um usuário a um grupo de usuários.

DROP

Remove um usuário de um grupo de usuários.

Nome de usuário do

Nome do usuário a ser adicionado ou excluído do grupo.

RENAME TO

Renomeia o grupo de usuários. Nomes de grupos que começam com dois sublinhados são reservados para uso interno do Amazon Redshift. Para obter mais informações sobre nomes válidos, consulte Nomes e identificadores.

new_name

Novo nome do grupo de usuários.

Exemplos

O exemplo a seguir adiciona um usuário denominado DWUSER ao grupo ADMIN_GROUP.

```
ALTER GROUP admin_group
ADD USER dwuser;
```

O exemplo a seguir renomeia o grupo ADMIN_GROUP para ADMINISTRATORS:

```
ALTER GROUP admin_group
RENAME TO administrators;
```

O exemplo a seguir adiciona dois usuários ao grupo ADMIN_GROUP.

```
ALTER GROUP admin_group
ADD USER u1, u2;
```

O exemplo a seguir descarta dois usuários do grupo ADMIN_GROUP.

```
ALTER GROUP admin_group
DROP USER u1, u2;
```

ALTER IDENTITY PROVIDER

Altera um provedor de identidades para atribuir novos parâmetros e valores. Quando você executa esse comando, todos os valores de parâmetro definidos anteriormente são excluídos antes de os novos valores serem atribuídos. Somente um superusuário pode alterar um provedor de identidades.

Sintaxe

```
ALTER IDENTITY PROVIDER identity_provider_name
[PARAMETERS parameter_string]
[NAMESPACE namespace]
[IAM_ROLE iam_role]
```

```
[AUTO_CREATE_ROLES
    [ TRUE [ { INCLUDE | EXCLUDE } GROUPS LIKE filter_pattern] |
      FALSE
    ]
[DISABLE | ENABLE]
```

Parâmetros

identity_provider_name

O nome do provedor de identidades. Para obter mais informações sobre nomes válidos, consulte Nomes e identificadores.

parameter_string

Uma string com um objeto JSON formatado corretamente que contém os parâmetros e valores necessários para o provedor de identidades específico.

Namespace do

O namespace da organização.

iam_role

O perfil do IAM que fornece permissões para a conexão com o Centro de Identidade do IAM. Esse parâmetro é aplicável somente quando o tipo de provedor de identidades é AWSIDC.

auto_create_roles

Habilita ou desabilita o atributo de criação automática de perfil. Se o valor for TRUE, o Amazon Redshift habilitará o recurso de criação automática de perfil. Se o valor for FALSE, o Amazon Redshift desabilitará o recurso de criação automática de perfil. Se o valor desse parâmetro não for especificado, o Amazon Redshift o determinará usando a seguinte lógica:

- Se `AUTO_CREATE_ROLES` for habilitado, mas o valor não for especificado, o valor será definido como TRUE.
- Se `AUTO_CREATE_ROLES` não for habilitado e o provedor de identidades for AWSIDC, o valor será definido como FALSE.
- Se `AUTO_CREATE_ROLES` não for habilitado e o provedor de identidades for o Azure, o valor será definido como TRUE.

Para incluir grupos, especifique `INCLUDE`. O padrão é vazio, o que significa incluir todos os grupos se `AUTO_CREATE_ROLES` estiver ativado.

Para excluir grupos, especifique `EXCLUDE`. O padrão é vazio, o que significa não excluir nenhum grupo se `AUTO_CREATE_ROLES` estiver ativado.

filter_pattern

Uma expressão de caractere UTF-8 válida com o padrão para estabelecer correspondência com os nomes de grupo. A opção LIKE executa uma correspondência com distinção entre letras maiúsculas e minúsculas compatível com os seguintes metacaracteres de correspondência de padrões:

Metacaractere	Descrição
%	Corresponde a qualquer sequência de zero ou mais caracteres.
_	Corresponde a qualquer caractere único.

Se filter_pattern não contiver metacaracteres, o padrão representará somente a própria string. Nesse caso, LIKE age da mesma forma que o operador de igualdade.

filter_pattern aceita os seguintes caracteres:

- Caracteres alfabéticos maiúsculos e minúsculos (A-Z e a-z)
- Números (0-9)
- Os seguintes caracteres especiais:

```
_ % ^ * + ? { } , $
```

DISABLE ou ENABLE

Ativa e desativa o provedor de identidades. O padrão é ENABLE.

Exemplos

O exemplo a seguir altera um provedor de identidades chamado oauth_standard. Aplica-se especificamente quando o Microsoft Azure AD é o provedor de identidades.

```
ALTER IDENTITY PROVIDER oauth_standard
PARAMETERS '{"issuer":"https://sts.windows.net/2sdfdsf-d475-420d-b5ac-667adad7c702/",
"client_id":"87f4aa26-78b7-410e-bf29-57b39929ef9a",
```

```
"client_secret":"BUAH~ewrqewrqwerUUY^%tHe1oNZShoiU7",
"audience":["https://analysis.windows.net/powerbi/connector/AmazonRedshift"]
}'
```

O exemplo a seguir mostra como definir o namespace do provedor de identidades. Isso poderá ser aplicado ao Microsoft Azure AD, se ele seguir uma declaração como o exemplo anterior, ou a outro provedor de identidades. Também poderá ser aplicado a um caso em que você conecte um cluster provisionado existente do Amazon Redshift ou um grupo de trabalho do Amazon Redshift sem servidor ao Centro de Identidade do IAM, se tiver uma conexão configurada por meio de uma aplicação gerenciada.

```
ALTER IDENTITY PROVIDER "my-redshift-idc-application"
NAMESPACE 'MYCO';
```

O exemplo a seguir define o perfil do IAM e funciona no caso de uso para configurar a integração do Redshift com o Centro de Identidade do IAM.

```
ALTER IDENTITY PROVIDER "my-redshift-idc-application"
IAM_ROLE 'arn:aws:iam::123456789012:role/myadministratorrole';
```

Para obter mais informações sobre a configuração de uma conexão do Redshift ao Centro de Identidade do IAM, consulte Conectar o Redshift ao IAM Identity Center para proporcionar aos usuários uma experiência de logon único.

Desabilitar um provedor de identidades

A declaração de exemplo a seguir mostra como desabilitar um provedor de identidades. Quando desabilitado, os usuários federados do provedor de identidades não podem fazer login no cluster enquanto ele não for habilitado novamente.

```
ALTER IDENTITY PROVIDER "redshift-idc-app" DISABLE;
```

ALTER MASKING POLICY

Altera uma política de mascaramento de dados dinâmica existente. Para obter mais informações sobre mascaramento dinâmico de dados, consulte Mascaramento dinâmico de dados.

Superusuários e usuários ou perfis que têm o perfil sys:secadmin podem alterar uma política de mascaramento.

Sintaxe

```
ALTER MASKING POLICY
{ policy_name | database_name.policy_name }
USING (masking_expression);
```

Parâmetros

policy_name

O nome da política de mascaramento. Deve ser o nome de uma política de mascaramento que já existe no banco de dados.

database_name

O nome do banco de dados no qual a política é criada. O banco de dados pode ser o conectado ou um banco de dados que comporte as permissões federadas do Amazon Redshift.

masking_expression

A expressão SQL usada para transformar as colunas de destino. Ela pode ser escrita usando funções de manipulação de dados, como funções de manipulação de strings, ou em conjunto com funções definidas pelo usuário escritas em SQL, em Python ou com o AWS Lambda.

A expressão deve corresponder às colunas de entrada e aos tipos de dados da expressão original. Por exemplo, se as colunas de entrada da política de mascaramento original fossem `sample_1 FLOAT` e `sample_2 VARCHAR(10)`, não seria possível alterar a política de mascaramento para receber uma terceira coluna ou fazer com que a política recebesse um FLOAT e um BOOLEAN. Se você usar uma constante como sua expressão de mascaramento, deverá convertê-la explicitamente em um tipo que corresponda ao tipo de entrada.

Você deve ter a permissão USAGE em todas as funções definidas pelo usuário que você usa na expressão de mascaramento.

Para o uso da ALTER MASKING POLICY no Catálogo de Permissões Federadas do Amazon Redshift, consulte Gerenciar o controle de acesso com permissões federadas do Amazon Redshift.

ALTER MATERIALIZED VIEW

Altera os atributos de uma visão materializada.

Sintaxe

```
ALTER MATERIALIZED VIEW mv_name
{
AUTO REFRESH { YES | NO }
| ALTER DISTKEY column_name
| ALTER DISTSTYLE ALL
| ALTER DISTSTYLE EVEN
| ALTER DISTSTYLE KEY DISTKEY column_name
| ALTER DISTSTYLE AUTO
| ALTER [COMPOUND] SORTKEY ( column_name [,...] )
| ALTER SORTKEY AUTO
| ALTER SORTKEY NONE
| ROW LEVEL SECURITY { ON | OFF } [ CONJUNCTION TYPE { AND | OR } ] [FOR DATASHARES]
};
```

Parâmetros

mv_name

O nome da visão materializada a ser alterada.

AUTO REFRESH { YES | NO }

Uma cláusula que ativa ou desativa a atualização automática de uma visão materializada. Para obter informações sobre atualização automática ou visões materializadas, consulte Atualizar uma visualização materializada.

ALTER DISTSTYLE ALL

Uma cláusula que altera o estilo de distribuição existente de uma tabela para ALL. Considere o seguinte:

- Um ALTER DISTSTYLE, ALTER SORTKEY e VACUUM não podem ser executados simultaneamente na mesma relação.
  - Se VACUUM estiver sendo executado no momento, a execução ALTER DISTSTYLE ALL retornará um erro.
  - Se ALTER DISTSTYLE ALL estiver em execução, um vacuum em segundo plano não será iniciado em uma relação.
- O comando ALTER DISTSTYLE ALL não é compatível para relações com chaves de classificação intercaladas e tabelas temporárias.

- Se o estilo de distribuição tiver sido definido anteriormente como AUTO, a relação não será mais considerada para otimização automática de tabela.

Para obter mais informações sobre DISTSTYLE ALL, acesse CREATE MATERIALIZED VIEW.

ALTER DISTSTYLE EVEN

Uma cláusula que altera o estilo de distribuição existente de uma tabela para EVEN. Considere o seguinte:

- Um ALTER DISTSYTLE, ALTER SORTKEY e VACUUM não podem ser executados simultaneamente na mesma relação.
  - Se VACUUM estiver sendo executado no momento, a execução ALTER DISTSTYLE EVEN retornará um erro.
  - Se ALTER DISTSTYLE EVEN estiver em execução, um vacuum em segundo plano não será iniciado na relação.
- O comando ALTER DISTSTYLE EVEN não é compatível para relações com chaves de classificação intercaladas e tabelas temporárias.
- Se o estilo de distribuição tiver sido definido anteriormente como AUTO, a relação não será mais considerada para otimização automática de tabela.

Para obter mais informações sobre DISTSTYLE EVEN, acesse CREATE MATERIALIZED VIEW.

ALTER DISTKEY column_name ou ALTER DISTSTYLE KEY DISTKEY column_name

Uma cláusula que altera a coluna usada como chave de distribuição de uma relação. Considere o seguinte:

- VACUUM e ALTER DISTKEY não podem ser executados simultaneamente na mesma relação.
  - Se VACUUM já estiver sendo executado, então, ALTER DISTKEY retornará um erro.
  - Se ALTER DISTKEY estiver em execução, o vacuum em segundo plano não será iniciado em uma relação.
  - Se ALTER DISTKEY estiver sendo executado, a limpeza em primeiro plano retornará um erro.
- Você só pode executar um comando ALTER DISTKEY em uma relação por vez.
- O comando ALTER DISTKEY não é compatível para relações com chaves de classificação intercaladas.
- Se o estilo de distribuição tiver sido definido anteriormente como AUTO, a relação não será mais considerada para otimização automática de tabela.

Ao especificar DISTSTYLE KEY, os dados são distribuídos pelos valores na coluna DISTKEY. Para obter mais informações sobre DISTSTYLE, consulte CREATE MATERIALIZED VIEW.

ALTER DISTSTYLE AUTO

Uma cláusula que altera o estilo de distribuição existente de uma relação para AUTO.

Quando você altera um estilo de distribuição para AUTO, o estilo de distribuição da relação é definido como o seguinte:

- Uma pequena relação com DISTSTYLE ALL é convertida em AUTO(ALL).
- Uma pequena relação com DISTSTYLE EVEN é convertida em AUTO(ALL).
- Uma pequena relação com DISTSTYLE KEY é convertida em AUTO(ALL).
- Uma grande relação com DISTSTYLE ALL é convertida em AUTO(EVEN).
- Uma grande relação com DISTSTYLE EVEN é convertida em AUTO(EVEN).
- Uma grande relação com DISTSTYLE KEY é convertida em AUTO(KEY) e a DISTKEY é preservada. Nesse caso, o Amazon Redshift não faz alterações na relação.

Se o Amazon Redshift determinar que um novo estilo de distribuição ou chave melhorará a performance das consultas, o Amazon Redshift poderá alterar o estilo de distribuição ou a chave da relação no futuro. Por exemplo, o Amazon Redshift pode converter uma relação com um DISTSTYLE de AUTO(KEY) em AUTO(EVEN) ou vice-versa. Para obter mais informações sobre o comportamento quando as chaves de distribuição são alteradas, incluindo redistribuição de dados e bloqueios, acesse Recomendações do Amazon Redshift Advisor.

Para mais informações sobre DISTSTYLE AUTO, acesse CREATE MATERIALIZED VIEW.

Para visualizar o estilo de distribuição de uma relação, consulte a visualização de catálogo SVV_TABLE_INFO do sistema. Para obter mais informações, acesse SVV_TABLE_INFO. Para visualizar as recomendações do Amazon Redshift Advisor para relações, consulte a visualização de catálogo SVV_ALTER_TABLE_RECOMMENDATIONS do sistema. Para obter mais informações, acesse SVV_ALTER_TABLE_RECOMMENDATIONS. Para exibir as ações executadas pelo Amazon Redshift, consulte a visualização do catálogo do sistema SVL_AUTO_WORKER_ACTION. Para obter mais informações, acesse SVL_AUTO_WORKER_ACTION.

ALTER [COMPOUND] SORTKEY (column_name [,...])

Uma cláusula que altera ou adiciona a chave de classificação usada para uma relação. ALTER SORTKEY não é compatível com tabelas temporárias.

Quando você altera uma chave de classificação, a codificação de compactação de colunas na chave de classificação nova ou original pode ser alterada. Se nenhuma codificação for definida explicitamente para a relação, o Amazon Redshift atribuirá automaticamente codificações de compactação da seguinte maneira:

- Colunas que são definidas como chaves de classificação são designadas a compactação RAW.
- Colunas que são definidas como tipos de dados BOOLEAN, REAL ou DOUBLE PRECISION recebem a compactação RAW.
- As colunas definidas como SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIME, TIMETZ, TIMESTAMP ou TIMESTAMPTZ recebem a compactação AZ64.
- As colunas definidas como CHAR ou VARCHAR recebem a compactação LZO.

Considere o seguinte:

- Você pode definir no máximo 400 colunas para uma chave de classificação por relação.
- É possível alterar uma chave de classificação intercalada para uma chave de classificação composta ou nenhuma chave de classificação. Porém, não é possível alterar uma chave de classificação composta para uma chave de classificação intercalada.
- Se a chave de classificação tiver sido definida anteriormente como AUTO, a relação não será mais considerada para otimização automática de tabela.
- O Amazon Redshift recomenda o uso de codificação RAW (sem compactação) para colunas definidas como chaves de classificação. Quando você altera uma coluna para escolhê-la como uma chave de classificação, a compactação da coluna é alterada para compactação RAW (sem compactação). Isso pode aumentar a quantidade de armazenamento necessária para a relação. Quanto o tamanho da relação aumenta depende da definição específica da relação e do conteúdo da relação. Para obter mais informações sobre compactação, acesse Codificações de compactação.

Quando os dados são carregados em uma relação, eles são carregados na ordem da chave de classificação. Quando você altera a chave de classificação, o Amazon Redshift reorganiza os dados. Para obter mais informações sobre SORTKEY, consulte CREATE MATERIALIZED VIEW.

ALTER SORTKEY AUTO

Uma cláusula que altera ou adiciona a chave de classificação da relação de destino para AUTO. ALTER SORTKEY AUTO não é compatível com tabelas temporárias.

Quando você altera uma chave de classificação para AUTO, o Amazon Redshift preserva a chave de classificação existente da relação.

Se o Amazon Redshift determinar que uma nova chave de classificação melhorará a performance das consultas, o Amazon Redshift poderá alterar a chave de classificação da sua relação no futuro.

Para obter mais informações sobre SORTKEY AUTO, consulte CREATE MATERIALIZED VIEW.

Para visualizar a chave de classificação de uma relação, consulte a visualização de catálogo SVV_TABLE_INFO do sistema. Para obter mais informações, acesse SVV_TABLE_INFO. Para visualizar as recomendações do Amazon Redshift Advisor para relações, consulte a visualização de catálogo SVV_ALTER_TABLE_RECOMMENDATIONS do sistema. Para obter mais informações, acesse SVV_ALTER_TABLE_RECOMMENDATIONS. Para exibir as ações executadas pelo Amazon Redshift, consulte a visualização do catálogo do sistema SVL_AUTO_WORKER_ACTION. Para obter mais informações, acesse SVL_AUTO_WORKER_ACTION.

ALTER SORTKEY NONE

Uma cláusula que remove a chave de classificação da relação de destino.

Se a chave de classificação tiver sido definida anteriormente como AUTO, a relação não será mais considerada para otimização automática de tabela.

ROW LEVEL SECURITY { ON | OFF } [CONJUNCTION TYPE { AND | OR }] [FOR DATASHARES]

Uma cláusula que ativa ou desativa a segurança no nível da linha para uma relação.

Quando a segurança no nível da linha é ativada para uma relação, você só pode ler as linhas às quais a política de segurança no nível da linha permite acesso. Quando não há política que conceda acesso à relação, você não consegue ver nenhuma linha da tabela. Somente superusuários e usuários ou perfis que tenham o perfil `sys:secadmin` podem definir a cláusula ROW LEVEL SECURITY. Para obter mais informações, consulte Segurança por linha.

- [CONJUNCTION TYPE { AND | OR }]

  Uma cláusula usada para escolher o tipo de conjunção da política de segurança por linha para uma relação. Quando várias políticas de segurança no nível da linha são anexadas a uma relação, você pode combinar as políticas com a cláusula AND ou OR. Por padrão, o Amazon Redshift combina políticas RLS com a cláusula AND. Superusuários, usuários ou funções que tenham a função `sys:secadmin` podem usar essa cláusula para definir o tipo de conjunção da política de segurança no nível da linha para uma relação. Para obter mais informações, consulte Combinar várias políticas por usuário.
- PARA UNIDADES DE COMPARTILHAMENTO DE DADOS

Uma cláusula que determina se uma relação protegida por RLS pode ser acessada por meio das unidades de compartilhamento de dados. Por padrão, uma relação protegida por RLS não pode ser acessada por meio de uma unidade de compartilhamento de dados. Um comando ALTER MATERIALIZED VIEW ROW LEVEL SECURITY executado com essa cláusula só afeta a propriedade de acessibilidade da unidade de compartilhamento de dados da relação. A propriedade ROW LEVEL SECURITY não foi alterada.

Se você tornar uma relação protegida por RLS acessível por meio de unidades de compartilhamentos de dados, a relação não terá segurança no nível da linha no banco de dados compartilhado no lado do consumidor. A relação mantém a propriedade RLS do lado do produtor.

Exemplos

O exemplo a seguir habilita a visão materializada `tickets_mv` a ser atualizada automaticamente.

```
ALTER MATERIALIZED VIEW tickets_mv AUTO REFRESH YES
```

Exemplos de DISTSTYLE e SORTKEY para ALTER MATERIALIZED VIEW

Os exemplos neste tópico mostram como realizar alterações DISTSTYLE e SORTKEY utilizando ALTER MATERIALIZED VIEW.

Os seguintes exemplos de consulta mostram como alterar uma coluna DISTSTYLE KEY DISTKEY usando um exemplo de tabela base:

```
CREATE TABLE base_inventory(
  inv_date_sk int4 NOT NULL,
  inv_item_sk int4 NOT NULL,
  inv_warehouse_sk int4 NOT NULL,
  inv_quantity_on_hand int4
);

INSERT INTO base_inventory VALUES(1,1,1,1);

CREATE materialized VIEW inventory diststyle even AS SELECT * FROM base_inventory;
SELECT "table", diststyle FROM svv_table_info WHERE "table" = 'inventory';

ALTER materialized VIEW inventory ALTER diststyle KEY distkey inv_warehouse_sk;
SELECT "table", diststyle FROM svv_table_info WHERE "table" = 'inventory';
```

```
ALTER materialized VIEW inventory ALTER distkey inv_item_sk;
SELECT "table", diststyle FROM svv_table_info WHERE "table" = 'inventory';

DROP TABLE base_inventory CASCADE;
```

Altere uma visão materializada para DISTSTYLE ALL:

```
CREATE TABLE base_inventory(
  inv_date_sk int4 NOT NULL,
  inv_item_sk int4 NOT NULL,
  inv_warehouse_sk int4 NOT NULL,
  inv_quantity_on_hand int4
);

INSERT INTO base_inventory VALUES(1,1,1,1);

CREATE materialized VIEW inventory diststyle even AS SELECT * FROM base_inventory;
SELECT "table", diststyle FROM svv_table_info WHERE "table" = 'inventory';

ALTER MATERIALIZED VIEW inventory ALTER diststyle ALL;
SELECT "table", diststyle FROM svv_table_info WHERE "table" = 'inventory';

DROP TABLE base_inventory CASCADE;
```

Os seguintes comandos mostram exemplos de ALTER MATERIALIZED VIEW SORTKEY usando uma tabela base de amostra:

```
CREATE TABLE base_inventory (c0 int, c1 int);

INSERT INTO base_inventory VALUES(1,1);

CREATE materialized VIEW inventory interleaved sortkey(c0, c1) AS SELECT * FROM
 base_inventory;
SELECT "table", sortkey1 FROM svv_table_info WHERE "table" = 'inventory';

ALTER materialized VIEW inventory ALTER sortkey(c0, c1);
SELECT "table", diststyle, sortkey_num FROM svv_table_info WHERE "table" = 'inventory';

ALTER materialized VIEW inventory ALTER sortkey NONE;
SELECT "table", diststyle, sortkey_num FROM svv_table_info WHERE "table" = 'inventory';

ALTER materialized VIEW inventory ALTER sortkey(c0);
```

```
SELECT "table", diststyle, sortkey_num FROM svv_table_info WHERE "table" = 'inventory';

DROP TABLE base_inventory CASCADE;
```

ALTER RLS POLICY

Altere uma política de segurança por linha em uma tabela.

Superusuários e usuários ou perfis que têm o perfil `sys:secadmin` podem alterar uma política.

Sintaxe

```
ALTER RLS POLICY
{ policy_name | database_name.policy_name }
USING ( using_predicate_exp );
```

Parâmetros

policy_name

O nome da política.

database_name

O nome do banco de dados no qual a política é criada. O banco de dados pode ser o conectado ou um banco de dados que comporte as permissões federadas do Amazon Redshift.

USING (using_predicate_exp)

Especifica um filtro que é aplicado à cláusula WHERE da consulta. O Amazon Redshift aplica um predicado de política antes dos predicados do usuário no nível da consulta. Por exemplo, **`current_user = 'joe' and price > 10`** limita Joe a ver apenas registros com o preço superior a US$ 10.

A expressão tem acesso às variáveis declaradas na cláusula WITH da instrução CREATE RLS POLICY que foi usada para criar a política com o nome policy_name.

Para o uso da ALTER RLS POLICY no Catálogo de Permissões Federadas do Amazon Redshift, consulte Gerenciar o controle de acesso com permissões federadas do Amazon Redshift.

Exemplos

O exemplo a seguir altera uma política de RLS.

```
-- First create an RLS policy that limits access to rows where catgroup is 'concerts'.
CREATE RLS POLICY policy_concerts
WITH (catgroup VARCHAR(10))
USING (catgroup = 'concerts');

-- Then, alter the RLS policy to only show rows where catgroup is 'piano concerts'.
ALTER RLS POLICY policy_concerts
USING (catgroup = 'piano concerts');
```

ALTER ROLE

Renomeia uma função ou altera o proprietário. Para conferir a lista de perfis definidos pelo sistema do Amazon Redshift, consulte the section called “Funções definidas pelo sistema do Amazon Redshift”.

Permissões obrigatórias

A seguir estão as permissões necessárias para ALTER ROLE:

- Superusuário
- Usuários com as permissões de ALTER ROLE

Sintaxe

```
ALTER ROLE role [ WITH ]
  { { RENAME TO role } | { OWNER TO user_name } }[, ...]
  [ EXTERNALID TO external_id ]
```

Parâmetros

Perfil do

O nome da função a ser alterada.

RENAME TO

Um novo nome para a função.

OWNER TO user_name

Um novo proprietário para a função.

EXTERNALID TO external_id

Um novo ID externo para a função, que está associado a um provedor de identidades. Para obter mais informações, consulte Federação do provedor de identidades (IdP) nativo para o Amazon Redshift.

Exemplos

O exemplo a seguir altera o nome de uma função de `sample_role1` para `sample_role2`.

```
ALTER ROLE sample_role1 RENAME TO sample_role2;
```

O exemplo a seguir altera o proprietário da função.

```
ALTER ROLE sample_role1 WITH OWNER TO user1
```

A sintaxe de ALTER ROLE é semelhante a ALTER PROCEDURE a seguir.

```
ALTER PROCEDURE first_quarter_revenue(bigint, numeric) RENAME TO quarterly_revenue;
```

O seguinte exemplo altera o proprietário de um procedimento para `etl_user`.

```
ALTER PROCEDURE quarterly_revenue(bigint, numeric) OWNER TO etl_user;
```

O exemplo a seguir atualiza uma função `sample_role1` com um novo ID externo associado a um provedor de identidades.

```
ALTER ROLE sample_role1 EXTERNALID TO "XYZ456";
```

ALTER PROCEDURE

Renomeia um procedimento ou altera o proprietário. São necessários o nome do procedimento e os tipos de dados, ou a assinatura. Somente o proprietário ou um usuário avançado pode renomear um procedimento. Somente um usuário avançado pode alterar o proprietário de um procedimento.

Sintaxe

```
ALTER PROCEDURE sp_name [ ( [ [ argname ] [ argmode ] argtype [, ...] ] ) ]
```

```
    RENAME TO new_name
```

```
ALTER PROCEDURE sp_name [ ( [ [ argname ] [ argmode ] argtype [, ...] ] ) ]
    OWNER TO { new_owner | CURRENT_USER | SESSION_USER }
```

Parâmetros

sp_name

O nome do procedimento a ser alterado. Especifique apenas o nome do procedimento no caminho de pesquisa atual ou use o formato `schema_name.sp_procedure_name` para adotar um esquema específico.

[argname] [argmode] argtype

Uma lista de nomes de argumentos, modos de argumentos e tipos de dados. Somente os tipos de dados de entrada são obrigatórios, usados para identificar o procedimento armazenado. Como alternativa, você pode fornecer a assinatura completa usada para criar o procedimento, incluindo os parâmetros de entrada e saída com seus modos.

new_name

Um novo nome para o procedimento armazenado.

new_owner | CURRENT_USER | SESSION_USER

Um novo proprietário para o procedimento armazenado.

Exemplos

O exemplo a seguir altera o nome de um procedimento de `first_quarter_revenue` para `quarterly_revenue`.

```
ALTER PROCEDURE first_quarter_revenue(volume INOUT bigint, at_price IN numeric,
 result OUT int) RENAME TO quarterly_revenue;
```

Este exemplo é equivalente ao seguinte:

```
ALTER PROCEDURE first_quarter_revenue(bigint, numeric) RENAME TO quarterly_revenue;
```

O seguinte exemplo altera o proprietário de um procedimento para `etl_user`.

```
ALTER PROCEDURE quarterly_revenue(bigint, numeric) OWNER TO etl_user;
```

ALTER SCHEMA

Altera a definição de um esquema existente. Use este comando para renomear um esquema ou alterar o proprietário de um esquema. Por exemplo, renomeie um esquema existente para preservar uma cópia de backup do esquema quando planejar criar uma nova versão do esquema em questão. Para obter mais informações sobre esquemas, consulte CREATE SCHEMA.

Para exibir as cotas de esquema configuradas, consulte SVV_SCHEMA_QUOTA_STATE.

Para exibir os registros em que as cotas de esquema foram excedidas, consulte STL_SCHEMA_QUOTA_VIOLATIONS.

Privilégios obrigatórios

Veja a seguir os privilégios obrigatórios para ALTER SCHEMA:

- Superusuário
- Usuário com o privilégio ALTER SCHEMA
- Proprietário do esquema

Ao alterar o nome de um esquema, observe que os objetos que usam o nome antigo, como procedimentos armazenados ou visões materializadas, devem ser atualizados para usar o novo nome.

Sintaxe

```
ALTER SCHEMA schema_name
{
RENAME TO new_name |
OWNER TO new_owner |
QUOTA { quota [MB | GB | TB] | UNLIMITED }
}
```

Parâmetros

schema_name

Nome do esquema de banco de dados a ser alterado.

RENAME TO

Cláusula que renomeia o esquema.

new_name

Novo nome do esquema. Para obter mais informações sobre nomes válidos, consulte Nomes e identificadores.

OWNER TO

Cláusula que altera o proprietário do esquema.

new_owner

Novo proprietário do esquema.

QUOTA

A quantidade máxima de espaço em disco que o esquema especificado pode usar. Esse espaço é o tamanho coletivo de todas as tabelas no esquema especificado. O Amazon Redshift converte o valor selecionado para megabytes. Gigabytes é a unidade de medida padrão quando um valor não é especificado.

Para obter mais informações sobre como configurar cotas de esquema, consulte CREATE SCHEMA.

Exemplos

O exemplo a seguir renomeia o esquema SALES para US_SALES.

```
alter schema sales
rename to us_sales;
```

O exemplo a seguir fornece a propriedade do esquema US_SALES ao usuário DWUSER.

```
alter schema us_sales
owner to dwuser;
```

O exemplo a seguir altera a cota para 300 MB e remove a cota.

```
alter schema us_sales QUOTA 300 GB;
```

```
alter schema us_sales QUOTA UNLIMITED;
```

ALTER SYSTEM

Altera uma opção de configuração no nível de sistema para o cluster do Amazon Redshift ou o grupo de trabalho do Redshift sem servidor.

Privilégios obrigatórios

Um dos seguintes tipos de usuário pode executar o comando ALTER SYSTEM:

- Superusuário
- Usuário Admin

Sintaxe

```
ALTER SYSTEM SET system-level-configuration = {true| t | on | false | f | off}
```

Parâmetros

system-level-configuration

Uma configuração no nível do sistema. Valor válido: `data_catalog_auto_mount` e `metadata_security`.

{true| t | on | false | f | off}

Um valor para ativar ou desativar a configuração no nível do sistema. Os valores `true`, `t` ou on indicam a ativação da configuração. Os valores `false`, `f` ou `off` indicam a desativação da configuração.

Observações de uso

Para um cluster provisionado, mudará para `data_catalog_auto_mount` na próxima reinicialização do cluster. Para obter mais informações, consulte Reinicialização de um cluster no Guia de gerenciamento do Amazon Redshift.

Para um grupo de trabalho sem servidor, as alterações feitas em `data_catalog_auto_mount` não entram em vigor imediatamente.

Exemplos

O exemplo a seguir ativa a montagem automática do AWS Glue Data Catalog.

```
ALTER SYSTEM SET data_catalog_auto_mount = true;
```

O exemplo a seguir ativa a segurança de metadados.

```
ALTER SYSTEM SET metadata_security = true;
```

Definir um namespace de identidade padrão

Esse exemplo é específico ao trabalho com um provedor de identidades. É possível integrar o Redshift ao Centro de Identidade do IAM e a um provedor de identidades para centralizar o gerenciamento de identidades do Redshift e de outros serviços da AWS.

O exemplo a seguir mostra como definir o namespace de identidade padrão para o sistema. Fazer isso posteriormente simplifica a execução das declarações GRANT e CREATE, porque não é necessário incluir o namespace como prefixo para cada identidade.

```
ALTER SYSTEM SET default_identity_namespace = 'MYCO';
```

Após a execução do comando, é possível executar declarações como as seguintes:

```
GRANT SELECT ON TABLE mytable TO alice;

GRANT UPDATE ON TABLE mytable TO salesrole;

CREATE USER bob password 'md50c983d1a624280812631c5389e60d48c';
```

O efeito de definir o namespace de identidade padrão é que cada identidade não o exige como prefixo. Neste exemplo, `alice` é substituído por `MYCO:alice`. Isso acontece com qualquer identidade incluída. Para ter mais informações sobre como usar um provedor de identidades com o Redshift, consulte Conectar o Redshift ao IAM Identity Center para proporcionar aos usuários uma experiência de logon único.

Para ter mais informações sobre definições relativas à configuração do Centro de Identidade do IAM, consulte SET e ALTER IDENTITY PROVIDER.

ALTER TABLE

Esse comando altera a definição de uma tabela do Amazon Redshift ou de uma tabela externa do Amazon Redshift Spectrum. Esse comando atualiza os valores e propriedades definidos por CRIAR TABELA ou CREATE EXTERNAL TABLE. É possível usar ALTER TABLE em uma visualização para segurança por linha (RLS).

Não é possível executar ALTER TABLE em uma tabela externa em um bloco de transação (BEGIN ... END). Para obter mais informações sobre transações, consulte Níveis de isolamento no Amazon Redshift.

ALTER TABLE bloqueia a tabela para operações de leitura e gravação até que a transação envolvendo a operação ALTER TABLE seja concluída, a menos que seja especificamente indicado na documentação que você pode consultar dados ou executar outras operações na tabela enquanto ela estiver sendo alterada.

Privilégios obrigatórios

O usuário que altera uma tabela precisa do privilégio adequado para que o comando seja bem-sucedido. Dependendo do comando ALTER TABLE, é necessário um dos privilégios a seguir.

- Superusuário
- Usuários com o privilégio ALTER TABLE
- Proprietário da tabela com o privilégio USAGE no esquema

Sintaxe

```
ALTER TABLE table_name
{
ADD table_constraint
| DROP CONSTRAINT constraint_name [ RESTRICT | CASCADE ]
| OWNER TO new_owner
| RENAME TO new_name
| RENAME COLUMN column_name TO new_name
| ALTER COLUMN column_name TYPE updated_varchar_data_type_size
| ALTER COLUMN column_name ENCODE new_encode_type
| ALTER COLUMN column_name ENCODE encode_type,
| ALTER COLUMN column_name ENCODE encode_type, .....;
| ALTER DISTKEY column_name
| ALTER DISTSTYLE ALL
```

```
| ALTER DISTSTYLE EVEN
| ALTER DISTSTYLE KEY DISTKEY column_name
| ALTER DISTSTYLE AUTO
| ALTER [COMPOUND] SORTKEY ( column_name [,...] )
| ALTER SORTKEY AUTO
| ALTER SORTKEY NONE
| ALTER ENCODE AUTO
| ADD [ COLUMN ] column_name column_type
  [ DEFAULT default_expr ]
  [ ENCODE encoding ]
  [ NOT NULL | NULL ]
  [ COLLATE { CASE_SENSITIVE | CS | CASE_INSENSITIVE | CI } ] |
| DROP [ COLUMN ] column_name [ RESTRICT | CASCADE ]
| ROW LEVEL SECURITY { ON | OFF } [ CONJUNCTION TYPE { AND | OR } ] [ FOR DATASHARES ]
| MASKING { ON | OFF } FOR DATASHARES }

where table_constraint is:

[ CONSTRAINT constraint_name ]
{ UNIQUE ( column_name [, ... ] )
| PRIMARY KEY ( column_name [, ... ] )
| FOREIGN KEY (column_name [, ... ] )
   REFERENCES  reftable [ ( refcolumn ) ]}

The following options apply only to external tables:

SET LOCATION { 's3://bucket/folder/' | 's3://bucket/manifest_file' }
| SET FILE FORMAT format |
| SET TABLE PROPERTIES ('property_name'='property_value')
| PARTITION ( partition_column=partition_value [, ...] )
  SET LOCATION { 's3://bucket/folder' |'s3://bucket/manifest_file' }
| ADD [IF NOT EXISTS]
    PARTITION ( partition_column=partition_value [, ...] ) LOCATION
 { 's3://bucket/folder' |'s3://bucket/manifest_file' }
    [, ... ]
| DROP PARTITION ( partition_column=partition_value [, ...] )
```

Para reduzir o tempo de execução do comando ALTER TABLE, você pode combinar algumas cláusulas do comando ALTER TABLE.

O Amazon Redshift oferece suporte às seguintes combinações de cláusulas de ALTER TABLE:

```
ALTER TABLE tablename ALTER SORTKEY (column_list), ALTER DISTKEY column_Id;
```

```
ALTER TABLE tablename ALTER DISTKEY column_Id, ALTER SORTKEY (column_list);
ALTER TABLE tablename ALTER SORTKEY (column_list), ALTER DISTSTYLE ALL;
ALTER TABLE tablename ALTER DISTSTYLE ALL, ALTER SORTKEY (column_list);
```

Parâmetros

table_name

Nome da tabela a ser alterada. Especifique somente o nome da tabela ou use o formato schema_name.table_name para usar um esquema específico. As tabelas externas devem ser qualificadas por um nome de esquema externo. Você também poderá especificar um nome de exibição se estiver usando a instrução ALTER TABLE para renomear ou alterar o proprietário de uma exibição. O tamanho máximo de um nome de tabela é 127 bytes; nomes mais longos são truncados para ter no máximo 127 bytes. É possível usar caracteres multibyte UFT-8 até um máximo de quatro bytes. Para obter mais informações sobre nomes válidos, consulte Nomes e identificadores.

ADD restrição_tabela

Cláusula que adiciona a restrição especificada à tabela. Para descrições de valores de table_constraint válidos, consulte CRIAR TABELA.

> **Note**
>
> Não é possível adicionar uma restrição de chave primária a uma coluna anulável. Se a coluna tiver sido criada originalmente com a restrição NOT NULL, é possível adicionar uma restrição de chave primária.

DROP CONSTRAINT nome_restrição

Cláusula que remove a restrição denominada da tabela. Para remover uma restrição, especifique o nome da restrição, não o tipo. Para visualizar nomes de restrições de tabelas, execute a consulta a seguir.

```
select constraint_name, constraint_type
from information_schema.table_constraints;
```

RESTRICT

Cláusula que remove somente a restrição especificada. RESTRICT é uma opção para o comando DROP CONSTRAINT. RESTRICT não pode ser usada com CASCADE.

CASCADE

Cláusula que remove a restrição especificada e qualquer dependência dessa restrição. CASCADE é uma opção para o comando DROP CONSTRAINT. CASCADE não pode ser usada com RESTRICT.

OWNER TO novo_proprietário

Cláusula que altera o proprietário da tabela (ou exibição) para o valor new_owner.

RENAME TO novo_nome

Cláusula que renomeia uma tabela (ou exibição) para o valor especificado em new_name. O tamanho máximo do nome da tabela é 127 bytes; nomes mais longos são truncados para ter no máximo 127 bytes.

Não é possível renomear uma tabela permanente para um nome que comece com '#'. '#' no início do nome é indicação de uma tabela temporária.

Você não pode renomear uma tabela externa.

ALTER COLUMN column_name TYPE updated_varchar_data_type_size

Uma cláusula que altera o tamanho de uma coluna definida como um tipo de dados VARCHAR. Essa cláusula só oferece suporte à alteração do tamanho de um tipo de dado VARCHAR. Considere as seguintes limitações:

- Não é possível alterar uma coluna com as codificações de compactação BYTEDICT, RUNLENGTH, TEXT255 ou TEXT32K.
- Você não reduzir o tamanho para menos que o tamanho máximo dos dados existentes.
- Não é possível alterar colunas com valores padrão
- Não é possível alterar colunas com UNIQUE, PRIMARY KEY ou FOREIGN KEY.
- Não é possível alterar colunas em um bloco de transação (BEGIN ... END). Para obter mais informações sobre transações, consulte Níveis de isolamento no Amazon Redshift.

ALTER COLUMN column_name ENCODE new_encode_type

Cláusula que altera a codificação de compactação de uma coluna. Se você especificar a codificação de compactação para uma coluna, a tabela não será mais definida como ENCODE

AUTO. Para obter informações sobre a codificação de compactação, consulte Compactação de colunas para reduzir o tamanho dos dados armazenados.

Quando você altera a codificação de compactação para uma coluna, a tabela permanece disponível para consulta.

Considere as seguintes limitações:

- Você não pode alterar uma coluna para a mesma codificação definida atualmente para a coluna.
- Não é possível alterar a codificação de uma coluna em uma tabela com uma chave de classificação intercalada.

ALTER COLUMN column_name ENCODE encode_type, ALTER COLUMN column_name ENCODE encode_type, ...;

Uma cláusula que altera a codificação de compactação de várias colunas em um único comando. Para obter informações sobre a codificação de compactação, consulte Compactação de colunas para reduzir o tamanho dos dados armazenados.

Quando você altera a codificação de compactação para uma coluna, a tabela permanece disponível para consulta.

Considere as seguintes limitações:

- Você não pode alterar uma coluna para o mesmo tipo de codificação ou uma diferente várias vezes em um único comando.
- Você não pode alterar uma coluna para a mesma codificação definida atualmente para a coluna.
- Não é possível alterar a codificação de uma coluna em uma tabela com uma chave de classificação intercalada.

ALTER DISTSTYLE ALL

Uma cláusula que altera o estilo de distribuição existente de uma tabela para ALL. Considere o seguinte:

- ALTER DISTSTYLE, ALTER SORTKEY e VACUUM não podem ser executados simultaneamente na mesma tabela.
  - Se VACUUM estiver sendo executado no momento, a execução ALTER DISTSTYLE ALL retornará um erro.

- Se ALTER DISTSTYLE ALL estiver em execução, a limpeza em segundo plano não será iniciada em uma tabela.

- O comando ALTER DISTSTYLE ALL não é compatível com tabelas com chaves de classificação intercaladas e tabelas temporárias.
- Se o estilo de distribuição foi definido anteriormente como AUTO, a tabela não é mais um candidato para otimização automática de tabela.

Para mais informações sobre DISTSTYLE ALL, consulte CRIAR TABELA.

ALTER DISTSTYLE EVEN

Uma cláusula que altera o estilo de distribuição existente de uma tabela para EVEN. Considere o seguinte:

- Um ALTER DISTSYTLE, ALTER SORTKEY e VACUUM não podem ser executados simultaneamente na mesma tabela.
  - Se VACUUM estiver sendo executado no momento, a execução ALTER DISTSTYLE EVEN retornará um erro.
  - Se ALTER DISTSTYLE EVEN estiver em execução, o vacuum em segundo plano não será iniciado em uma tabela.
- O comando ALTER DISTSTYLE EVEN não é compatível com tabelas com chaves de classificação intercaladas nem com tabelas temporárias.
- Se o estilo de distribuição foi definido anteriormente como AUTO, a tabela não é mais um candidato para otimização automática de tabela.

Para obter mais informações sobre DISTSTYLE EVEN, consulte CRIAR TABELA.

ALTER DISTKEY column_name ou ALTER DISTSTYLE KEY DISTKEY column_name

Uma cláusula que altera a coluna usada como a chave de distribuição da tabela. Considere o seguinte:

- VACUUM e ALTER DISTKEY não podem ser executados ao mesmo tempo na mesma tabela.
  - Se VACUUM já estiver sendo executado, então, ALTER DISTKEY retornará um erro.
  - Se ALTER DISTKEY estiver em execução, a limpeza em segundo plano não será iniciada na tabela.
  - Se ALTER DISTKEY estiver sendo executado, a limpeza em primeiro plano retornará um erro.

- Você só pode executar um comando ALTER DISTKEY por vez em uma tabela.
- O comando ALTER DISTKEY não é compatível com tabelas com chaves de classificação intercalada.
- Se o estilo de distribuição foi definido anteriormente como AUTO, a tabela não é mais um candidato para otimização automática de tabela.

Ao especificar DISTSTYLE KEY, os dados são distribuídos pelos valores na coluna DISTKEY. Para mais informações sobre DISTSTYLE, consulte CRIAR TABELA.

ALTER DISTSTYLE AUTO

Uma cláusula que altera o estilo de distribuição existente de uma tabela para AUTO.

Quando você altera um estilo de distribuição para AUTO, o estilo de distribuição da tabela é definido para o seguinte:

- Uma pequena tabela com DISTSTYLE ALL é convertida em AUTO(ALL).
- Uma pequena tabela com DISTSTYLE EVEN é convertida em AUTO(ALL).
- Uma pequena tabela com DISTSTYLE KEY é convertida em AUTO(ALL).
- Uma grande tabela com DISTSTYLE ALL é convertida em AUTO(EVEN).
- Uma grande tabela com DISTSTYLE EVEN é convertida em AUTO(EVEN).
- Uma tabela grande com DISTSTYLE KEY é convertida em AUTO(KEY) e a DISTKEY é mantida. Nesse caso, o Amazon Redshift não faz alterações na tabela.

Se o Amazon Redshift determinar que um novo estilo de distribuição ou chave melhorará a performance das consultas, o Amazon Redshift poderá alterar o estilo de distribuição ou a chave da sua tabela no futuro. Por exemplo, o Amazon Redshift pode converter uma tabela com um DISTSTYLE de AUTO(KEY) em AUTO(EVEN), ou vice-versa. Para obter mais informações sobre o comportamento quando as chaves de distribuição são alteradas, incluindo redistribuição de dados e bloqueios, consulte Recomendações do Amazon Redshift Advisor.

Para mais informações sobre DISTSTYLE AUTO, consulte CRIAR TABELA.

Para exibir o estilo de distribuição de uma tabela, consulte a visualização do catálogo do sistema SVV_TABLE_INFO. Para obter mais informações, consulte SVV_TABLE_INFO. Para exibir as recomendações do Amazon Redshift Advisor para tabelas, consulte a visualização do catálogo do sistema SVV_ALTER_TABLE_RECOMMENDATIONS. Para obter mais informações, consulte SVV_ALTER_TABLE_RECOMMENDATIONS. Para exibir as ações executadas pelo Amazon

Redshift, consulte a visualização do catálogo do sistema SVL_AUTO_WORKER_ACTION. Para obter mais informações, consulte SVL_AUTO_WORKER_ACTION.

ALTER [COMPOUND] SORTKEY (column_name [,...])

Uma cláusula que altera ou adiciona a chave de classificação usada para uma tabela. ALTER SORTKEY não é compatível com tabelas temporárias.

Quando você altera uma chave de classificação, a codificação de compactação de colunas na chave de classificação nova ou original pode ser alterada. Se nenhuma codificação for explicitamente definida para a tabela, o Amazon Redshift atribuirá automaticamente codificações de compactação da seguinte forma:

- Colunas que são definidas como chaves de classificação são designadas a compactação RAW.
- Colunas que são definidas como tipos de dados BOOLEAN, REAL ou DOUBLE PRECISION recebem a compactação RAW.
- As colunas definidas como SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIME, TIMETZ, TIMESTAMP ou TIMESTAMPTZ recebem a compactação AZ64.
- As colunas definidas como CHAR ou VARCHAR recebem a compactação LZO.

Considere o seguinte:

- Você pode definir um máximo de 400 colunas para uma chave de classificação por tabela.
- É possível alterar uma chave de classificação intercalada para uma chave de classificação composta ou nenhuma chave de classificação. Porém, não é possível alterar uma chave de classificação composta para uma chave de classificação intercalada.
- Se a chave de classificação tiver sido definida anteriormente como AUTO, a tabela não será mais candidata à otimização automática da tabela.
- O Amazon Redshift recomenda o uso de codificação RAW (sem compactação) para colunas definidas como chaves de classificação. Quando você altera uma coluna para escolhê-la como uma chave de classificação, a compactação da coluna é alterada para compactação RAW (sem compactação). Isso pode aumentar a quantidade de armazenamento necessária pela tabela. O quanto o tamanho da tabela aumenta depende da definição específica da tabela e do conteúdo da tabela. Para obter mais informações sobre compactação, consulte . Codificações de compactação

Quando os dados são carregados em uma tabela, os dados são carregados na ordem da chave de classificação. Quando você altera a chave de classificação, o Amazon Redshift reorganiza os dados. Para obter mais informações sobre SORTKEY, consulte CRIAR TABELA.

ALTER SORTKEY AUTO

Uma cláusula que altera ou adiciona a chave de classificação da tabela de destino como AUTO. ALTER SORTKEY AUTO não é compatível com tabelas temporárias.

Quando você altera uma chave de classificação para AUTO, o Amazon Redshift preserva a chave de classificação existente da tabela.

Se o Amazon Redshift determinar que uma nova chave de classificação melhorará a performance das consultas, o Amazon Redshift poderá alterar a chave de classificação ou a chave da sua tabela futuramente.

Para obter mais informações sobre SORTKEY AUTO, consulte CRIAR TABELA.

Para exibir a chave de classificação de uma tabela, consulte a visualização do catálogo do sistema SVV_TABLE_INFO. Para obter mais informações, consulte SVV_TABLE_INFO. Para exibir as recomendações do Amazon Redshift Advisor para tabelas, consulte a visualização do catálogo do sistema SVV_ALTER_TABLE_RECOMMENDATIONS. Para obter mais informações, consulte SVV_ALTER_TABLE_RECOMMENDATIONS. Para exibir as ações executadas pelo Amazon Redshift, consulte a visualização do catálogo do sistema SVL_AUTO_WORKER_ACTION. Para obter mais informações, consulte SVL_AUTO_WORKER_ACTION.

ALTER SORTKEY NONE

Cláusula que remove a chave de classificação da tabela de destino.

Se a chave de classificação tiver sido definida anteriormente como AUTO, a tabela não será mais candidata à otimização automática da tabela.

ALTER ENCODE AUTO

Uma cláusula que altera o tipo de codificação das colunas da tabela de destino para AUTO. Quando você altera a codificação para AUTO, o Amazon Redshift preserva o tipo de codificação existente das colunas na tabela. Em seguida, se o Amazon Redshift determinar que um novo tipo de codificação pode melhorar a performance da consulta, o Amazon Redshift poderá alterar o tipo de codificação das colunas da tabela.

Se você alterar uma ou mais colunas para especificar uma codificação, o Amazon Redshift não ajustará mais automaticamente a codificação para todas as colunas da tabela. As colunas retêm as configurações de codificação atuais.

As seguintes ações não afetam a configuração ENCODE AUTO para a tabela:

- Renomear a tabela.
- Alterar a configuração DISTSTYLE ou SORTKEY para a tabela.
- Adicionar ou eliminar uma coluna com uma configuração ENCODE.
- Usar a opção COMPUPDATE do comando COPY. Para obter mais informações, consulte Operações de carregamento de dados.

Para exibir a codificação de uma tabela, consulte a visualização do catálogo do sistema SVV_TABLE_INFO. Para obter mais informações, consulte SVV_TABLE_INFO.

RENAME COLUMN column_name TO new_name

Cláusula que renomeia uma coluna para o valor especificado em new_name. O tamanho máximo do nome da coluna é 127 bytes; nomes mais longos são truncados para ter no máximo 127 bytes. Para obter mais informações sobre nomes válidos, consulte Nomes e identificadores.

ADD [COLUMN] nome_coluna

Cláusula que adiciona uma coluna com o nome especificado à tabela. Só é possível adicionar uma coluna em cada instrução ALTER TABLE.

Não é possível adicionar uma coluna que é a chave de distribuição (DISTKEY) ou uma chave de classificação (SORTKEY) da tabela.

Não é possível usar o comando ALTER TABLE ADD COLUMN para modificar os atributos da tabela e da coluna a seguir:

- UNIQUE
- PRIMARY KEY
- REFERENCES (chave externa)
- IDENTITY ou GENERATED BY DEFAULT AS IDENTITY

O tamanho máximo do nome da coluna é 127 bytes; nomes mais longos são truncados para ter no máximo 127 bytes. O número máximo de colunas que você pode definir em uma única tabela é 1.600.

As seguintes restrições são aplicadas ao adicionar uma coluna a uma tabela externa:

- Você não pode adicionar uma coluna a uma tabela externa com a restrições de coluna DEFAULT, ENCODE, NOT NULL ou NULL.

- Você não pode adicionar colunas a uma tabela externa definida usando o formato de arquivo AVRO.
- Se pseudocolunas estiverem habilitadas, o número máximo de colunas que poderá ser definido em uma única tabela externa será 1.598. Se pseudocolunas não estiverem habilitadas, o número máximo de colunas que poderá ser definido em uma única tabela será 1.600.

Para obter mais informações, consulte CREATE EXTERNAL TABLE.

column_type

O tipo de dados da coluna que está sendo adicionada. Para as colunas CHAR e VARCHAR, é possível usar a palavra-chave MAX em vez de declarar o tamanho máximo. MAX define o tamanho máximo de CHAR para 4.096 bytes ou de VARCHAR para 65.535 bytes. O tamanho máximo de um objeto GEOMETRY é 1.048.447 bytes.

Para obter mais informações sobre os tipos de dados que o Amazon Redshift aceita, consulte Tipos de dados.

DEFAULT default_expr

Cláusula que atribui um valor de dados padrão à coluna. O tipo de dados expr_padrão deve ser compatível com o tipo de dados da coluna. O valor DEFAULT deve ser uma expressão sem variáveis. Não são permitidas subconsultas, referências cruzadas de outras colunas na tabela atual e funções definidas pelo usuário.

default_expr é usado em qualquer operação INSERT que não especifique um valor para a coluna. Se não houver um valor padrão especificado, o valor padrão da coluna é nulo.

Se uma operação COPY encontrar um campo nulo em uma coluna com um valor DEFAULT e uma restrição NOT NULL, o comando COPY vai inserir o valor da default_expr.

DEFAULT não é compatível com tabelas externas.

ENCODE encoding

Codificação de compactação de uma coluna. Por padrão, o Amazon Redshift gerencia automaticamente a codificação de compactação para todas as colunas de uma tabela se você não especificar a codificação de compactação para qualquer coluna da tabela ou se especificar a opção ENCODE AUTO para a tabela.

Se você especificar a codificação de compactação para qualquer coluna na tabela ou se não especificar a opção ENCODE AUTO para a tabela, o Amazon Redshift atribuirá automaticamente

a codificação de compactação às colunas para as quais você não especifica a codificação de compactação da seguinte maneira:

- Todas as colunas nas tabelas temporárias são atribuídas à compactação RAW como padrão.
- Colunas que são definidas como chaves de classificação são designadas a compactação RAW.
- Colunas que são definidas como tipos de dados BOOLEAN, REAL, DOUBLE PRECISION, GEOMETRY ou GEOGRAPHY recebem a compactação RAW.
- As colunas definidas como SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIME, TIMETZ, TIMESTAMP ou TIMESTAMPTZ recebem a compactação AZ64.
- As colunas definidas como CHAR, VARCHAR ou VARBYTE recebem a compactação LZO.

> **Note**
>
> Se você não quiser que uma coluna seja compactada, especifique explicitamente codificação RAW.

Os seguintes compression encodings (p. 81) são compatíveis:

- AZ64
- BYTEDICT
- DELTA
- DELTA32K
- LZO
- MOSTLY8
- MOSTLY16
- MOSTLY32
- RAW (sem compactação)
- RUNLENGTH
- TEXT255
- TEXT32K
- ZSTD

ENCODE não é compatível com tabelas externas.

NOT NULL | NULL

NOT NULL especifica que a coluna não deve conter valores nulos. NULL, o padrão, especifica que a coluna aceita valores nulos.

NOT NULL e NULL não são compatíveis com tabelas externas.

COLLATE { CASE_SENSITIVE | CS | CASE_INSENSITIVE | CI }

Uma cláusula que especifica se a pesquisa ou comparação de string na coluna distingue ou não maiúscula e minúsculas. O valor padrão é o mesmo da configuração atual de diferenciação de maiúsculas e minúsculas do banco de dados.

Para verificar o agrupamento atual de um banco de dados, use a função DB_COLLATION.

CASE_SENSITIVE e CS são intercambiáveis e geram os mesmos resultados. Da mesma forma, CASE_INSENSITIVE e CI são intercambiáveis e geram os mesmos resultados.

DROP [COLUMN] column_name

Nome da coluna a ser excluída da tabela.

Você não pode descartar a última coluna de uma tabela. Uma tabela deve ter pelo menos uma coluna.

Não é possível descartar uma coluna que seja a chave de distribuição (DISTKEY) ou uma chave de classificação (SORTKEY) da tabela. Comportamento padrão de DROP COLUMN será RESTRICT se a coluna tiver qualquer objeto dependente, como uma exibição, uma chave primária, uma chave externa ou uma restrição UNIQUE.

As seguintes restrições são aplicadas ao remover uma coluna de uma tabela externa:

- Você não poderá descartar uma coluna de uma tabela externa se a coluna for usada como uma partição.
- Você não pode descartar uma coluna de uma tabela externa definida usando o formato de arquivo AVRO.
- RESTRICT e CASCADE são ignorados para tabelas externas.
- Não é possível descartar as colunas da tabela de políticas referenciada dentro da definição de política, a menos que você descarte ou desanexe a política. Isso também se aplica quando a opção CASCADE é especificada. Você pode descartar outras colunas na tabela de políticas.

Para obter mais informações, consulte CREATE EXTERNAL TABLE.

RESTRICT

Quando usado com DROP COLUMN, RESTRICT significa que a coluna a ser descartada não será descartada, nestes casos:

- Se uma visualização definida fizer referência à coluna que está sendo descartada
- Se uma chave externa fizer referência à coluna
- Se a coluna fizer parte de uma chave multipart

RESTRICT não pode ser usada com CASCADE.

RESTRICT e CASCADE são ignorados para tabelas externas.

CASCADE

Quando usado com DROP COLUMN, remove a coluna especificada e qualquer objeto dependente dessa coluna. CASCADE não pode ser usada com RESTRICT.

RESTRICT e CASCADE são ignorados para tabelas externas.

As opções a seguir se aplicam somente a tabelas externas.

SET LOCATION { 's3://bucket/folder/' | 's3://bucket/manifest_file' }

O caminho para a pasta ou bucket do Amazon S3 que contém arquivos de dados ou um arquivo manifesto que contém uma lista de caminhos de objetos do Amazon S3. Os buckets devem estar na mesma região da AWS que o cluster do Amazon Redshift. Para obter uma lista de regiões da AWS compatíveis, consulte Limitações do Amazon Redshift Spectrum. Para obter mais informações sobre o uso de um arquivo manifesto, consulte LOCATION na referência CREATE EXTERNAL TABLE Parâmetros.

SET FILE FORMAT format

Formato para arquivos de dados externos.

Os formatos válidos são:

- AVRO
- PARQUET
- RCFILE

- SEQUENCEFILE
- TEXTFILE

SET TABLE PROPERTIES ('property_name'='property_value')

Uma cláusula que estabelece a definição da tabela de propriedades da tabela para uma tabela externa.

> **Note**
>
> As propriedades de tabela fazem distinção entre maiúsculas e minúsculas.

'numRows'='row_count'

Uma propriedade que define o valor de numRows para a definição da tabela. Para atualizar de maneira explícita as estatísticas de uma tabela externa, defina a propriedade numRows de maneira a indicar o tamanho da tabela. O Amazon Redshift não analisa as tabelas externas para gerar as estatísticas das tabelas que o otimizador de consultas utiliza para gerar um plano de consulta. Se as estatísticas da tabela não estiverem configuradas para uma tabela externa, o Amazon Redshift gerará um plano de execução de consulta. Tal plano é baseado em uma suposição de que as tabelas externas são as maiores e as tabelas locais são as menores.

'skip.header.line.count'='line_count'

Uma propriedade que define o número de linhas a serem ignoradas no início de cada arquivo de origem.

'format-version'='version'

Especifica a versão do formato da tabela do Iceberg. Valores possíveis: '2' (padrão), '3'. A versão 3 do formato é compatível com recursos adicionais, como valores padrão de colunas. No momento da criação, se não for especificado, o Amazon Redshift criará a tabela como uma tabela do Iceberg v2. Em uma tabela existente, definir essa propriedade como '3' atualiza a tabela de v2 para v3.

PARTITION (partition_column=partition_value [, ...] SET LOCATION { 's3://bucket/folder' | 's3://bucket/manifest_file' }

Cláusula que define um novo local para uma ou mais colunas de partição.

ADD [IF NOT EXISTS] PARTITION (partition_column=partition_value [, ...]) LOCATION { 's3://bucket/folder' | 's3://bucket/manifest_file' } [, ...]

Uma cláusula que adiciona uma ou mais partições. É possível especificar várias cláusulas PARTITION usando um único comando ALTER TABLE… ADD.

> **Note**
>
> Se usar o catálogo do AWS Glue, você poderá adicionar até 100 partições usando um único comando ALTER TABLE.

A cláusula IF NOT EXISTS indica que, se a partição especificada já existir, o comando não deverá fazer alterações. Também indica que o comando deverá retornar uma mensagem de que a partição existe, em vez de finalizar com um erro. Esta cláusula é útil para realizar scripting para que o script não falhe se o comando ALTER TABLE tentar adicionar uma partição que já existe.

DROP PARTITION (partition_column=partition_value [, ...])

Cláusula que remove a partição especificada. Remover uma partição altera somente os metadados da tabela externa. Os dados no Amazon S3 não são afetados.

ROW LEVEL SECURITY { ON | OFF } [CONJUNCTION TYPE { AND | OR }] [FOR DATASHARES]

Uma cláusula que ativa ou desativa a segurança no nível da linha para uma relação.

Quando a segurança no nível da linha é ativada para uma relação, você só pode ler as linhas às quais a política de segurança no nível da linha permite acesso. Quando não há política que conceda acesso à relação, você não consegue ver nenhuma linha da tabela. Somente superusuários e usuários ou perfis que tenham o perfil `sys:secadmin` podem definir a cláusula ROW LEVEL SECURITY. Para obter mais informações, consulte Segurança por linha. Essa declaração é aceita no banco de dados conectado ou em um banco de dados com permissões federadas do Amazon Redshift. A cláusula FOR DATASHARES não é aceita em um banco de dados com permissões federadas do Amazon Redshift.

- [CONJUNCTION TYPE { AND | OR }]

  Uma cláusula usada para escolher o tipo de conjunção da política de segurança por linha para uma relação. Quando várias políticas de segurança no nível da linha são anexadas a uma relação, você pode combinar as políticas com a cláusula AND ou OR. Por padrão, o Amazon Redshift combina políticas RLS com a cláusula AND. Superusuários, usuários ou funções que

tenham a função `sys:secadmin` podem usar essa cláusula para definir o tipo de conjunção da política de segurança no nível da linha para uma relação. Para obter mais informações, consulte Combinar várias políticas por usuário.

- PARA UNIDADES DE COMPARTILHAMENTO DE DADOS

  Uma cláusula que determina se uma relação protegida por RLS pode ser acessada por meio das unidades de compartilhamento de dados. Por padrão, uma relação protegida por RLS não pode ser acessada por meio de uma unidade de compartilhamento de dados. Um comando ALTER TABLE ROW LEVEL SECURITY executado com essa cláusula só afeta a propriedade de acessibilidade da unidade de compartilhamento de dados da relação. A propriedade ROW LEVEL SECURITY não foi alterada.

  Se você tornar uma relação protegida por RLS acessível por meio de unidades de compartilhamentos de dados, a relação não terá segurança no nível da linha no banco de dados compartilhado no lado do consumidor. A relação mantém a propriedade RLS do lado do produtor.

MASKING { ON | OFF } FOR DATASHARES

Uma cláusula que determina se uma relação protegida por DDM pode ser acessada por meio de unidades de compartilhamento de dados. Por padrão, uma relação protegida por DDM não pode ser acessada por meio de uma unidade de compartilhamento de dados. Se você tornar uma relação protegida por DDM acessível por meio de unidades de compartilhamentos de dados, a relação não terá proteção por mascaramento no banco de dados compartilhado no lado do consumidor. A relação mantém a propriedade de mascaramento no lado do produtor. Somente superusuários e usuários ou perfis que tenham a função `sys:secadmin` podem definir a cláusula MASKING FOR DATASHARES. Para obter mais informações, consulte Mascaramento dinâmico de dados.

Exemplos

Para exemplos que mostram como usar o comando ALTER TABLE, confira os tópicos a seguir.

- Exemplos de ALTER TABLE
- Exemplos de ALTER EXTERNAL TABLE
- Exemplos de ALTER TABLE ADD e DROP COLUMN

Exemplos de ALTER TABLE

Os exemplos a seguir demonstram o uso básico de comando ALTER TABLE.

Renomear uma tabela ou visão

O comando a seguir renomeia a tabela de USERS para USERS_BKUP:

```
alter table users
rename to users_bkup;
```

Você também pode usar esse tipo de comando para renomear uma exibição.

Alterar o proprietário de uma tabela ou visualização

O comando a seguir altera o proprietário da tabela VENUE para o usuário DWUSER:

```
alter table venue
owner to dwuser;
```

Os comandos a seguir criam uma exibição, depois alteram seu proprietário:

```
create view vdate as select * from date;
alter table vdate owner to vuser;
```

Renomeação de uma coluna

O comando a seguir renomeia a coluna VENUESEATS na tabela VENUE para VENUESIZE:

```
alter table venue
rename column venueseats to venuesize;
```

Remover uma restrição de tabela

Para remover uma restrição de tabela, como uma chave primária, chave externa ou restrição exclusiva, primeiro encontre o nome interno da restrição. Depois, especifique o nome da restrição no comando ALTER TABLE. O exemplo a seguir encontra as restrições da tabela CATEGORY, depois remove a chave primária com o nome `category_pkey`.

```
select constraint_name, constraint_type
from information_schema.table_constraints
```

```
where constraint_schema ='public'
and table_name = 'category';

constraint_name | constraint_type
----------------+----------------
category_pkey   | PRIMARY KEY

alter table category
drop constraint category_pkey;
```

Alterar uma coluna VARCHAR

Para poupar armazenamento, você pode definir uma tabela inicialmente com colunas VARCHAR com o tamanho mínimo necessário para os requisitos dos dados atuais. Posteriormente, para acomodar strings mais longas, você poderá alterar a tabela para aumentar o tamanho da coluna.

O exemplo a seguir aumenta o tamanho da coluna EVENTNAME para VARCHAR(300).

```
alter table event alter column eventname type varchar(300);
```

Alterar uma coluna VARBYTE

Para poupar armazenamento, é possível definir uma tabela inicialmente com colunas VARBYTE com o tamanho mínimo necessário para os requisitos dos dados atuais. Posteriormente, para acomodar strings mais longas, você poderá alterar a tabela para aumentar o tamanho da coluna.

O exemplo a seguir aumenta o tamanho da coluna EVENTNAME para VARBYTE(300).

```
alter table event alter column eventname type varbyte(300);
```

Alterar a codificação de compactação para uma coluna

É possível alterar a codificação de compactação de uma coluna. Abaixo, é possível encontrar um conjunto de exemplos demonstrando essa abordagem. A seguir, a definição da tabela para esses exemplos.

```
create table t1(c0 int encode lzo, c1 bigint encode zstd, c2 varchar(16) encode lzo, c3
 varchar(32) encode zstd);
```

A instrução a seguir altera a codificação de compactação para a coluna c0 de codificação LZO para codificação AZ64.

```
alter table t1 alter column c0 encode az64;
```

A instrução a seguir altera a codificação de compactação para a coluna c1 de codificação Zstandard para codificação AZ64.

```
alter table t1 alter column c1 encode az64;
```

A instrução a seguir altera a codificação de compactação para a coluna c2 de codificação LZO para codificação Byte-dictionary.

```
alter table t1 alter column c2 encode bytedict;
```

A instrução a seguir altera a codificação de compactação para a coluna c3 de codificação Zstandard para codificação Runlength.

```
alter table t1 alter column c3 encode runlength;
```

Alterar a coluna DISTSTYLE KEY DISTKEY

Os exemplos a seguir mostram como alterar o DISTSTYLE e DISTKEY de uma tabela.

Crie uma tabela com o estilo de distribuição EVEN. A visualização SVV_TABLE_INFO mostra que o DISTSTYLE é EVEN.

```
create table inventory(
  inv_date_sk int4 not null ,
  inv_item_sk int4 not null ,
  inv_warehouse_sk int4 not null ,
  inv_quantity_on_hand int4
) diststyle even;

Insert into inventory values(1,1,1,1);

select "table", "diststyle" from svv_table_info;

   table   |   diststyle
-----------+----------------
 inventory |     EVEN
```

Altere a tabela DISTKEY para `inv_warehouse_sk`. A visualização SVV_TABLE_INFO mostra a coluna `inv_warehouse_sk` como a chave de distribuição resultante.

```
alter table inventory alter diststyle key distkey inv_warehouse_sk;

select "table", "diststyle" from svv_table_info;

   table   |       diststyle
-----------+----------------------
 inventory | KEY(inv_warehouse_sk)
```

Altere a tabela DISTKEY para `inv_item_sk`. A visualização SVV_TABLE_INFO mostra a coluna `inv_item_sk` como a chave de distribuição resultante.

```
alter table inventory alter distkey inv_item_sk;

select "table", "diststyle" from svv_table_info;

   table   |       diststyle
-----------+----------------------
 inventory | KEY(inv_item_sk)
```

Alterar uma tabela para DISTSTYLE ALL

Os exemplos a seguir mostram como alterar uma tabela para DISTSTYLE ALL.

Crie uma tabela com o estilo de distribuição EVEN. A visualização SVV_TABLE_INFO mostra que o DISTSTYLE é EVEN.

```
create table inventory(
  inv_date_sk int4 not null ,
  inv_item_sk int4 not null ,
  inv_warehouse_sk int4 not null ,
  inv_quantity_on_hand int4
) diststyle even;

Insert into inventory values(1,1,1,1);

select "table", "diststyle" from svv_table_info;

   table   |   diststyle
-----------+---------------
```

```
 inventory |     EVEN
```

Altere a tabela DISTSTYLE para ALL. A visualização SVV_TABLE_INFO mostra a DISTSYTLE alterada.

```
alter table inventory alter diststyle all;

select "table", "diststyle" from svv_table_info;

   table   |   diststyle
-----------+----------------
 inventory |     ALL
```

Alterar uma tabela SORTKEY

É possível alterar uma tabela para ter uma chave de classificação composta ou nenhuma chave de classificação.

Na definição de tabela a seguir, a tabela `t1` é definida com uma chave de classificação intercalada.

```
create table t1 (c0 int, c1 int) interleaved sortkey(c0, c1);
```

O comando a seguir altera a tabela de uma chave de classificação intercalada para uma chave de classificação composta.

```
alter table t1 alter sortkey(c0, c1);
```

O comando a seguir altera a tabela para remover a chave de classificação intercalada.

```
alter table t1 alter sortkey none;
```

Na definição de tabela a seguir, a tabela `t1` é definida com coluna `c0` como chave de classificação intercalada.

```
create table t1 (c0 int, c1 int) sortkey(c0);
```

O comando a seguir altera a tabela `t1` para uma chave de classificação composta.

```
alter table t1 alter sortkey(c0, c1);
```

Alterar uma tabela para ENCODE AUTO

O exemplo a seguir mostra como alterar uma tabela para ENCODE AUTO.

A seguir, a definição da tabela para esse exemplo. A coluna `c0` é definida com o tipo de codificação AZ64 e coluna `c1` é definida com o tipo de codificação LZO.

```
create table t1(c0 int encode AZ64, c1 varchar encode LZO);
```

Para esta tabela, a instrução a seguir altera a codificação para AUTO.

```
alter table t1 alter encode auto;
```

O exemplo a seguir mostra como alterar uma tabela para remover a configuração ENCODE AUTO.

A seguir, a definição da tabela para esse exemplo. As colunas da tabela são definidas sem codificação. Nesse caso, a codificação usa como padrão ENCODE AUTO.

```
create table t2(c0 int, c1 varchar);
```

Para esta tabela, a instrução a seguir altera a codificação da coluna c0 para LZO. A codificação da tabela não está mais definida como ENCODE AUTO.

```
alter table t2 alter column c0 encode lzo;;
```

Alterar o controle de segurança por linha

O comando a seguir desativa o RLS para a tabela:

```
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY OFF;
```

O comando a seguir ativa o RLS para a tabela:

```
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY ON;
```

O seguinte comando ativa RLS para a tabela e a torna acessível por meio das unidades de compartilhamento de dados:

```
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY ON;
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY FOR DATASHARES OFF;
```

O seguinte comando ativa RLS para a tabela e a torna inacessível por meio das unidades de compartilhamento de dados:

```
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY ON;
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY FOR DATASHARES ON;
```

O seguinte comando ativa RLS e define o tipo de conjunção RLS como OR para a tabela:

```
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY ON CONJUNCTION TYPE OR;
```

O seguinte comando ativa RLS e define o tipo de conjunção RLS como AND para a tabela:

```
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY ON CONJUNCTION TYPE AND;
```

Exemplos de ALTER EXTERNAL TABLE

Os exemplos a seguir usam um bucket do Amazon S3 localizado na Região da AWSLeste dos EUA (Norte da Virgínia) (`us-east-1`) e as tabelas de exemplo criadas em Exemplos para CREATE TABLE. Para ter mais informações sobre como usar partições com tabelas externas, consulte Dividir as tabelas externas do Redshift Spectrum.

O exemplo a seguir define a propriedade de tabela numRows da tabela externa SPECTRUM.SALES para 170.000 linhas.

```
alter table spectrum.sales
set table properties ('numRows'='170000');
```

O exemplo a seguir altera o local da tabela externa SPECTRUM.SALES.

```
alter table spectrum.sales
set location 's3://redshift-downloads/tickit/spectrum/sales/';
```

O exemplo a seguir altera o formato da tabela externa SPECTRUM.SALES para Parquet.

```
alter table spectrum.sales
set file format parquet;
```

O exemplo a seguir adiciona uma partição à tabela SPECTRUM.SALES_PART.

```
alter table spectrum.sales_part
```

```
add if not exists partition(saledate='2008-01-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-01/';
```

O exemplo a seguir adiciona três partições à tabela SPECTRUM.SALES_PART.

```
alter table spectrum.sales_part add if not exists
partition(saledate='2008-01-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-01/'
partition(saledate='2008-02-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-02/'
partition(saledate='2008-03-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-03/';
```

O exemplo a seguir altera SPECTRUM.SALES_PART para remover a partição com `saledate='2008-01-01''`.

```
alter table spectrum.sales_part
drop partition(saledate='2008-01-01');
```

O exemplo a seguir define um novo caminho do Amazon S3 para a partição com `saledate='2008-01-01'`.

```
alter table spectrum.sales_part
partition(saledate='2008-01-01')
set location 's3://redshift-downloads/tickit/spectrum/sales_partition/
saledate=2008-01-01/';
```

O exemplo a seguir altera o nome de `sales_date` para `transaction_date`.

```
alter table spectrum.sales rename column sales_date to transaction_date;
```

O exemplo a seguir define o mapeamento de coluna para a posição de mapeamento para uma tabela externa que usa o formato de coluna de linha otimizada (ORC).

```
alter table spectrum.orc_example
set table properties('orc.schema.resolution'='position');
```

O exemplo a seguir define o mapeamento de coluna para o mapeamento de nome para uma tabela externa que usa o formato ORC.

```
alter table spectrum.orc_example
set table properties('orc.schema.resolution'='name');
```

Exemplos de ALTER TABLE ADD e DROP COLUMN

Os exemplos a seguir demonstram como usar o comando ALTER TABLE para adicionar e depois remover colunas básicas de tabela, e também como remover uma coluna com um objeto dependente.

Usar os comandos ADD e depois DROP em uma coluna básica

O exemplo a seguir adiciona uma coluna FEEDBACK_SCORE independente à tabela USERS. Esta coluna simplesmente contém um inteiro, e o valor padrão dessa coluna é NULL (sem pontuação de comentário).

Primeiro, consulte a tabela de catálogo PG_TABLE_DEF para exibir o esquema da tabela USERS:

```
column        | type                   | encoding | distkey | sortkey
--------------+------------------------+----------+---------+--------
userid        | integer                | delta    | true    |       1
username      | character(8)           | lzo      | false   |       0
firstname     | character varying(30)  | text32k  | false   |       0
lastname      | character varying(30)  | text32k  | false   |       0
city          | character varying(30)  | text32k  | false   |       0
state         | character(2)           | bytedict | false   |       0
email         | character varying(100) | lzo      | false   |       0
phone         | character(14)          | lzo      | false   |       0
likesports    | boolean                | none     | false   |       0
liketheatre   | boolean                | none     | false   |       0
likeconcerts  | boolean                | none     | false   |       0
likejazz      | boolean                | none     | false   |       0
likeclassical | boolean                | none     | false   |       0
likeopera     | boolean                | none     | false   |       0
likerock      | boolean                | none     | false   |       0
likevegas     | boolean                | none     | false   |       0
likebroadway  | boolean                | none     | false   |       0
likemusicals  | boolean                | none     | false   |       0
```

Depois adicione a coluna feedback_score:

```
alter table users
add column feedback_score int
```

```
default NULL;
```

Selecione a coluna FEEDBACK_SCORE em USERS para confirmar se ela foi adicionada:

```
select feedback_score from users limit 5;

feedback_score
----------------
NULL
NULL
NULL
NULL
NULL
```

Remova a coluna para restabelecer o DDL original:

```
alter table users drop column feedback_score;
```

Descartar uma coluna com um objeto dependente

O exemplo a seguir descarta uma coluna que possui um objeto dependente. Como resultado, o objeto dependente também é removido.

Para começar, adicione a coluna FEEDBACK_SCORE de novo à tabela USERS:

```
alter table users
add column feedback_score int
default NULL;
```

Em seguida, crie uma exibição chamada USERS_VIEW na tabela USERS:

```
create view users_view as select * from users;
```

Depois, tente remover a coluna FEEDBACK_SCORE da tabela USERS. A instrução DROP usa o comportamento padrão (RESTRICT):

```
alter table users drop column feedback_score;
```

O Amazon Redshift exibe uma mensagem de erro indicando que a coluna não pode ser descartada porque outro objeto depende dela.

Tente remover a coluna FEEDBACK_SCORE novamente, desta vez especificando em CASCADE para eliminar todos os objetos dependentes:

```
alter table users
drop column feedback_score cascade;
```

ALTER TABLE APPEND

Move dados de uma tabela de origem existente para acrescentar linhas a uma tabela de destino. Os dados da tabela de origem são movidos para as colunas correspondentes na tabela de destino. A ordem das colunas não importa. Depois que os dados forem acrescentados com sucesso na tabela de destino, a tabela de origem fica vazia. ALTER TABLE APPEND geralmente é muito mais rápido do que as operações CREATE TABLE AS ou INSERT semelhantes, pois os dados são movidos, não duplicados.

> **Note**
>
> ALTER TABLE APPEND move blocos de dados entre a tabela de origem e uma tabela de destino. Para melhorar a performance, ALTER TABLE APPEND não compacta o armazenamento como parte da operação de append. Como resultado, o uso do armazenamento aumenta temporariamente. Para recuperar o espaço, execute uma operação VACUUM.

Colunas com o mesmo nome também devem ter atributos de coluna idênticos. Se a tabela de origem ou a tabela de destino tiver colunas que não existem na outra tabela, use os parâmetros IGNOREEXTRA ou FILLTARGET para especificar como as colunas extras devem ser gerenciadas.

Não é possível acrescentar uma coluna de identidade. Se ambas as tabelas incluírem uma coluna de identidade, o comando falhará. Se somente uma tabela tiver uma coluna de identidade, inclua o parâmetro FILLTARGET ou IGNOREEXTRA. Para obter mais informações, consulte Observações de uso de ALTER TABLE APPEND.

É possível associar uma coluna GENERATED BY DEFAULT AS IDENTITY. Você pode atualizar as colunas definidas como GENERATED BY DEFAULT AS IDENTITY com os valores fornecidos. Para obter mais informações, consulte Observações de uso de ALTER TABLE APPEND.

A tabela de destino deve ser uma tabela permanente. No entanto, a origem pode ser uma tabela permanente ou uma visão materializada configurada para ingestão de streaming. Os objetos devem

usar o mesmo estilo de distribuição e a mesma chave de distribuição, caso definidos. Se os objetos estiverem classificados, ambos devem usar o mesmo estilo de classificação e definir as mesmas colunas como chaves de classificação.

Um comando ALTER TABLE APPEND é confirmado automaticamente logo depois da conclusão da operação. O comando não pode ser revertido. Não é possível executar ALTER TABLE APPEND em um bloco de transações (BEGIN ... END). Para obter mais informações sobre transações, consulte Níveis de isolamento no Amazon Redshift.

Privilégios obrigatórios

Dependendo do comando ALTER TABLE APPEND, é necessário um dos seguintes privilégios:

- Superusuário
- Usuários com o privilégio de sistema ALTER TABLE
- Usuários com os privilégios DELETE e SELECT na tabela de origem e com o privilégio INSERT na tabela de destino

Sintaxe

```
ALTER TABLE target_table_name APPEND FROM [ source_table_name
 | source_materialized_view_name ]
[ IGNOREEXTRA | FILLTARGET ]
```

A anexação com base em uma visão materializada funciona somente no caso em que a visão materializada está configurada para Ingestão de streaming para uma visão materializada.

Parâmetros

nome_tabela_destino

Nome da tabela à qual as linhas são acrescentadas. Especifique somente o nome da tabela ou use o formato schema_name.table_name para adotar um esquema específico. A tabela de destino deve ser uma tabela permanente existente.

FROM nome_tabela_origem

Nome da tabela que fornece as linhas que serão acrescentadas. Especifique somente o nome da tabela ou use o formato schema_name.table_name para adotar um esquema específico. A tabela de origem deve ser uma tabela permanente existente.

FROM source_materialized_view_name

O nome da visão materializada que fornece as linhas que serão anexadas. A anexação com base em uma visão materializada funciona somente no caso em que a visão materializada está configurada para Ingestão de streaming para uma visão materializada. A visão materializada de origem já deve existir.

IGNOREEXTRA

Palavra-chave que especifica se a tabela de origem inclui colunas que não constam na tabela de destino. Os dados nas colunas extras devem ser descartados. Não é possível usar IGNOREEXTRA com FILLTARGET.

FILLTARGET

Palavra-chave que especifica se uma tabela de destino inclui colunas que não constam na tabela de origem. As colunas devem ser preenchidas com o valor DEFAULT de coluna, caso haja um valor definido, ou com NULL. Não é possível usar IGNOREEXTRA com FILLTARGET.

Observações de uso de ALTER TABLE APPEND

- ALTER TABLE APPEND move somente colunas idênticas da tabela de origem para a tabela de destino. A ordem das colunas não importa.
- Se a tabela de origem ou de destino tiver colunas extras, use FILLTARGET ou IGNOREEXTRA de acordo com as seguintes regras:
  - Se a tabela de origem tiver colunas inexistentes na tabela de destino, inclua IGNOREEXTRA. O comando ignora as colunas extras na tabela de origem.
  - Se a tabela de destino tiver colunas inexistentes na tabela de origem, inclua FILLTARGET. O parâmetro preenche as colunas extras na tabela de destino com valores padrão de coluna ou com o valor IDENTITY, caso haja um valor definido, ou com NULL.
  - Se as tabelas de origem e de destino tiverem colunas extras, o comando falhará. Não é possível usar FILLTARGET e IGNOREEXTRA.
- Se uma coluna com o mesmo nome, porém atributos diferentes, for encontrada em ambas as tabelas, o comando falhará. Colunas de mesmo nome devem ter os atributos a seguir em comum:
  - Tipo de dados
  - Tamanho da coluna
  - Codificação de compactação
  - Não nulo

  - Estilo de classificação
  - Colunas de chave de classificação
  - Estilo de distribuição
  - Colunas de chave de distribuição
- Não é possível acrescentar uma coluna de identidade. Se as tabelas de origem e de destino tiverem colunas de identidade, o comando falhará. Se somente a tabela de origem tiver uma coluna de identidade, inclua o parâmetro IGNOREEXTRA para ignorar a coluna de identidade. Se somente a tabela de destino tiver uma coluna de identidade, inclua o parâmetro FILLTARGET para que a coluna de identidade seja preenchida de acordo com a cláusula IDENTITY definida para a tabela. Para obter mais informações, consulte DEFAULT.
- É possível associar uma coluna de identidade padrão com a instrução ALTER TABLE APPEND. Para obter mais informações, consulte CRIAR TABELA.
- As operações ALTER TABLE APPEND mantêm bloqueios exclusivos quando executadas em visões materializadas de streaming do Amazon Redshift conectadas a qualquer um dos seguintes:
  - Um Amazon Kinesis data stream
  - Um tópico Amazon Managed Streaming for Apache Kafka
  - Um stream externo compatível, como um tópico do Confluent Cloud Kafka

  Para obter mais informações, consulte Ingestão de streaming para uma visão materializada.

Exemplos de ALTER TABLE APPEND

Digamos que sua organização mantém uma tabela, SALES_MONTHLY, para capturar as transações de vendas atuais. Você deseja mover dados da tabela de transações para a tabela SALES todos os meses.

É possível usar os comandos INSERT INTO e TRUNCATE a seguir para realizar a tarefa.

```
insert into sales (select * from sales_monthly);
truncate sales_monthly;
```

No entanto, você pode realizar a mesma operação de maneira muito mais eficiente usando o comando ALTER TABLE APPEND.

Em primeiro lugar, consulte a PG_TABLE_DEF tabela de catálogo do sistema para verificar se ambas as tabelas têm as mesmas colunas com atributos de coluna idênticos.

```
select trim(tablename) as table, "column", trim(type) as type,
encoding, distkey, sortkey, "notnull"
from pg_table_def where tablename like 'sales%';

table      | column     | type                          | encoding | distkey | sortkey |
 notnull
-----------+------------+-------------------------------+----------+---------+---------
+--------
sales      | salesid    | integer                       | lzo      | false   |       0 |
 true
sales      | listid     | integer                       | none     | true    |       1 |
 true
sales      | sellerid   | integer                       | none     | false   |       2 |
 true
sales      | buyerid    | integer                       | lzo      | false   |       0 |
 true
sales      | eventid    | integer                       | mostly16 | false   |       0 |
 true
sales      | dateid     | smallint                      | lzo      | false   |       0 |
 true
sales      | qtysold    | smallint                      | mostly8  | false   |       0 |
 true
sales      | pricepaid  | numeric(8,2)                  | delta32k | false   |       0 |
 false
sales      | commission | numeric(8,2)                  | delta32k | false   |       0 |
 false
sales      | saletime   | timestamp without time zone | lzo      | false   |       0 |
 false
salesmonth | salesid    | integer                       | lzo      | false   |       0 |
 true
salesmonth | listid     | integer                       | none     | true    |       1 |
 true
salesmonth | sellerid   | integer                       | none     | false   |       2 |
 true
salesmonth | buyerid    | integer                       | lzo      | false   |       0 |
 true
salesmonth | eventid    | integer                       | mostly16 | false   |       0 |
 true
salesmonth | dateid     | smallint                      | lzo      | false   |       0 |
 true
salesmonth | qtysold    | smallint                      | mostly8  | false   |       0 |
 true
```

```
salesmonth | pricepaid  | numeric(8,2)                | delta32k | false   |       0 |
 false
salesmonth | commission | numeric(8,2)                | delta32k | false   |       0 |
 false
salesmonth | saletime   | timestamp without time zone | lzo      | false   |       0 |
 false
```

Em seguida, verifique o tamanho de cada tabela.

```
select count(*) from sales_monthly;
 count
-------
  2000
(1 row)

select count(*) from sales;
 count
-------
 412,214
(1 row)
```

Agora execute o comando ALTER TABLE APPEND.

```
alter table sales append from sales_monthly;
```

Verifique o tamanho de cada tabela novamente. A tabela SALES_MONTHLY agora contém 0 linha, e a tabela SALES tem 2.000 linhas a mais.

```
select count(*) from sales_monthly;
 count
-------
     0
(1 row)

select count(*) from sales;
 count
-------
 414214
(1 row)
```

Se a tabela de origem tiver mais colunas que a tabela de destino, especifique o parâmetro IGNOREEXTRA. O exemplo a seguir usa o parâmetro IGNOREEXTRA para ignorar colunas extras na tabela SALES_LISTING ao acrescentar dados na tabela SALES.

```
alter table sales append from sales_listing ignoreextra;
```

Se a tabela de destino tiver mais colunas que a tabela de origem, especifique o parâmetro FILLTARGET. O exemplo a seguir usa o parâmetro FILLTARGET para preencher colunas que não existem na tabela SALES_MONTH na tabela SALES_REPORT.

```
alter table sales_report append from sales_month filltarget;
```

O exemplo a seguir mostra como usar ALTER TABLE APPEND com uma visão materializada como origem.

```
ALTER TABLE target_tbl APPEND FROM my_streaming_materialized_view;
```

Neste exemplo, os nomes da tabela e da visão materializada são fictícios. A anexação com base em uma visão materializada funciona somente no caso em que a visão materializada está configurada para Ingestão de streaming para uma visão materializada. Ela move todos os registros da visão materializada de origem para uma tabela de destino com o mesmo esquema da visão materializada e deixa a visão materializada intacta. Esse comportamento é o mesmo de quando a origem dos dados é uma tabela.

ALTER TEMPLATE

Altera a definição de um modelo existente. Use esse comando para renomear um modelo, alterar o proprietário de um modelo, adicionar ou remover parâmetros da definição do modelo ou definir valores de parâmetro.

Privilégios obrigatórios

Para alterar um modelo, é necessário ter uma das seguintes opções:

- Privilégios de superusuário.
- O privilégio ALTER TEMPLATE e o privilégio USAGE no esquema que contém o modelo.

Sintaxe

```
ALTER TEMPLATE [database_name.][schema_name.]template_name
{
RENAME TO new_name
| OWNER TO new_owner
| ADD  parameter [AS] [value]
| DROP parameter
| SET parameter TO value1 [, parameter2 TO value2 , ...]
};
```

Parâmetros

database_name

(Opcional) O nome do banco de dados no qual o modelo é criado. Se não especificado, será usado o banco de dados atual.

schema_name

(Opcional) O nome do esquema no qual o modelo é criado. Se não for especificado, o modelo será pesquisado no caminho de pesquisa atual.

template_name

O nome do modelo a ser alterado.

RENAME TO

Uma cláusula que renomeia o modelo.

new_name

O novo nome do modelo. Para obter mais informações sobre nomes válidos, consulte Nomes e identificadores.

OWNER TO

Cláusula que altera o proprietário do modelo.

new_owner

O novo proprietário do modelo.

ADD parameter [AS] [value]

Adiciona um novo parâmetro ao modelo.

- Para parâmetros somente de palavras-chave (como CSV ou GZIP), especifique apenas o nome do parâmetro.
- Para parâmetros que exigem valores, especifique o nome do parâmetro seguido do valor. Opcionalmente, é possível incluir AS entre o parâmetro e o valor.

DROP parameter

Remove o parâmetro especificado do modelo. Não é possível eliminar vários parâmetros com um único comando DROP.

SET parameter TO value1 [, parameter2 TO value2 , ...]

Atualiza os valores de parâmetro de modelo existentes. Use somente para parâmetros que já tenham valores. Vários parâmetros podem ser atualizados em um único comando.

Exemplos

O exemplo a seguir muda o nome do modelo test_template para demo_template.

```
ALTER TEMPLATE test_template
RENAME TO demo_template;
```

O exemplo a seguir atribui a propriedade do esquema demo_template ao usuário bob.

```
ALTER TEMPLATE demo_template
OWNER TO bob;
```

O exemplo a seguir adiciona o parâmetro CSV ao modelo demo_template.

```
ALTER TEMPLATE demo_template
ADD CSV;
```

O exemplo a seguir adiciona o parâmetro `TIMEFORMAT 'auto'` ao modelo demo_template.

```
ALTER TEMPLATE demo_template
ADD TIMEFORMAT 'auto';
```

O exemplo a seguir remove o parâmetro ENCRYPTED do modelo demo_template.

```
ALTER TEMPLATE demo_template
```

```
DROP ENCRYPTED;
```

O seguinte exemplo define o DELIMITER parâmetro como '|' e o parâmetro TIMEFORMAT como 'epochsecs':

```
ALTER TEMPLATE demo_template
SET DELIMITER TO '|', TIMEFORMAT TO 'epochsecs';
```

ALTER USER

Muda um usuário do banco de dados.

Privilégios obrigatórios

A seguir estão os privilégios obrigatórios para ALTER USER:

- Superusuário
- Usuários com o privilégio ALTER USER
- Usuário atual que deseje alterar a própria senha

Sintaxe

```
ALTER USER username [ WITH ] option [, ... ]

where option is

CREATEDB | NOCREATEDB
| CREATEUSER | NOCREATEUSER
| SYSLOG ACCESS { RESTRICTED | UNRESTRICTED }
| PASSWORD { 'password' | 'md5hash' | 'sha256hash' | DISABLE }
[ VALID UNTIL 'expiration_date' ]
| RENAME TO new_name |
| CONNECTION LIMIT { limit | UNLIMITED }
| SESSION TIMEOUT limit | RESET SESSION TIMEOUT
| SET parameter { TO | = } { value | DEFAULT }
| RESET parameter
| EXTERNALID external_id
| NOLOGIN
| LOGIN PASSWORD { 'password' | 'md5hash' | 'sha256hash' }
```

Parâmetros

Nome de usuário do

Nome do usuário.

WITH

Palavra-chave opcional.

CREATEDB | NOCREATEDB

A opção CREATEDB permite que o usuário crie bancos de dados novos. NOCREATEDB é o valor padrão.

CREATEUSER | NOCREATEUSER

A opção CREATEUSER cria um superusuário com todos os privilégios de banco de dados, incluindo CREATE USER. NOCREATEUSER é o valor padrão. Para obter mais informações, consulte superuser.

SYSLOG ACCESS { RESTRICTED | UNRESTRICTED }

Uma cláusula que especifica o nível de acesso do usuário para as tabelas e as exibições de sistema do Amazon Redshift.

Usuários regulares que têm a permissão SYSLOG ACCESS RESTRICTED podem ver somente as linhas geradas por esse usuário nas tabelas e visualizações do sistema visíveis ao usuário. O padrão é RESTRICTED.

Usuários regulares que têm a permissão UNRESTRICTED podem ver todas as linhas nas tabelas e visualizações de sistema visíveis ao usuário, incluindo as linhas geradas por outro usuário. UNRESTRICTED não fornece acesso para os usuários regulares às tabelas visíveis para superusuários. Somente superusuários podem visualizar tabelas visíveis para superusuários.

> **Note**
>
> Fornecer acesso ilimitado para um usuário às tabelas de sistema é o mesmo que dar ao usuário visibilidade para os dados gerados por outros usuários. Por exemplo, STL_QUERY e STL_QUERYTEXT contêm texto completo de instruções INSERT, UPDATE e DELETE, e podem conter dados confidenciais gerados pelos usuários.

Todas as linhas em SVV_TRANSACTIONS são visíveis a todos os usuários.

Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

PASSWORD { 'password' | 'md5hash' | 'sha256hash' | DISABLE }

Define a senha do usuário.

Por padrão, os usuários podem alterar suas próprias senhas, a menos que a senha esteja desabilitada. Para desabilitar a senha de um usuário, especifique DISABLE. Quando a senha de um usuário for desabilitada, ela será excluída do sistema e o usuário poderá fazer logon apenas usando as credenciais temporárias do usuário do AWS Identity and Access Management (IAM). Para obter mais informações, consulte Uso da autenticação do IAM para gerar credenciais do usuário do banco de dados. Apenas um superusuário pode habilitar ou desabilitar senhas. Não é possível desabilitar a senha de um superusuário. Para habilitar uma senha, execute ALTER USER e especifique uma senha.

Para obter detalhes sobre como usar o parâmetro PASSWORD, consulte CREATE USER.

VALID UNTIL 'data_expiração'

Especifica que a senha tem uma data de expiração. Use o valor `'infinity'` para evitar ter uma data de expiração definida. O tipo de dados válido para o parâmetro é timestamp.

Somente superusuários podem usar esse parâmetro.

RENAME TO

Renomeia o usuário.

new_name

Novo nome de usuário. Para obter mais informações sobre nomes válidos, consulte Nomes e identificadores.

> **Important**
>
> Ao renomear um usuário, você também deve redefinir a senha do usuário. A senha redefinida não precisa ser diferente da senha anterior. O nome de usuário é usado como parte da criptografia da senha. Portanto, quando um usuário é renomeado, a senha é eliminada. O usuário só poderá fazer logon depois que a senha for redefinida. Por exemplo:

```
alter user newuser password 'EXAMPLENewPassword11';
```

CONNECTION LIMIT { limite | UNLIMITED }

Número máximo de conexões de banco de dados que o usuário pode abrir simultaneamente. Não há aplicação de limite para superusuários. Use a palavra-chave UNLIMITED para permitir o número máximo de conexões simultâneas. Um limite no número de conexões para cada banco de dados pode ser aplicável. Para obter mais informações, consulte CREATE DATABASE. O valor padrão é UNLIMITED. Para visualizar as conexões atuais, consulte a exibição STV_SESSIONS do sistema.

> **Note**
>
> Se limites de usuário e de conexão de banco de dados forem aplicáveis, um slot de conexão não utilizado que esteja dentro de ambos os limites deve estar disponível quando um usuário tenta se conectar.

SESSION TIMEOUT limit | RESET SESSION TIMEOUT

O tempo máximo em segundos em que uma sessão permanece inativa ou ociosa. O intervalo é de 60 segundos (um minuto) a 1.728.000 segundos (20 dias). Se nenhum tempo limite de sessão estiver definido para o usuário, a configuração de cluster será aplicada. Para obter mais informações, consulte “Cotas e limites no Amazon Redshift” no Guia de gerenciamento de clusters do Amazon Redshift.

Quando você define o tempo limite da sessão, ele é aplicado somente a novas sessões.

Para exibir informações sobre sessões de usuário ativas, incluindo a hora de início, o nome de usuário e o tempo limite da sessão, consulte a visualização de sistema STV_SESSIONS. Para exibir informações sobre o histórico de sessões de usuário, consulte a visualização STL_SESSIONS. Para recuperar informações sobre usuários do banco de dados, incluindo valores de tempo limite de sessão, consulte a visualização SVL_USER_INFO.

SET

Define um parâmetro de configuração para um novo valor padrão em todas as sessões executadas pelo usuário especificado.

RESET

Redefine um parâmetro de configuração para o valor padrão original para o usuário especificado.

Parâmetro

Nome do parâmetro a ser definido ou redefinido.

valor

Novo valor para o parâmetro.

DEFAULT

Define o parâmetro de configuração para o valor padrão em todas as sessões executadas pelo usuário especificado.

EXTERNALID external_id

O identificador para o usuário, que está associado a um provedor de identidades. O usuário deve ter a senha desabilitada. Para obter mais informações, consulte Federação do provedor de identidades (IdP) nativo para o Amazon Redshift.

NOLOGIN

Bloqueia o usuário especificado para que ele não possa se conectar ao banco de dados. O Amazon Redshift rejeita o login de usuários bloqueados. Quando você bloqueia um usuário, o Amazon Redshift não encerra as sessões existentes desse usuário. Para desbloquear o usuário, use LOGIN PASSWORD.

Especifique NOLOGIN sozinho, sem PASSWORD. Se você incluir PASSWORD com NOLOGIN, o Amazon Redshift vai alterar a senha do usuário, mas ainda bloqueará o usuário, o que deixa o usuário bloqueado com uma nova senha. Para bloquear um usuário sem alterar a senha, use apenas NOLOGIN.

O bloqueio de usuário se aplica somente a usuários de banco de dados baseados em senha. Usuários federados não podem ser bloqueados. Isso inclui usuários que se autenticam por meio do AWS Identity and Access Management (IAM) ou Centro de Identidade do AWS IAM. Você também não pode bloquear o usuário administrador. Para bloquear um usuário, é necessário ser um superusuário ou um usuário com o privilégio ALTER USER. Essa opção só estará disponível quando a segurança de bloqueio de usuários estiver habilitada para o cluster. Para obter mais informações, consulte SHOW USER LOCKOUT e max_failed_login_attempts.

LOGIN PASSWORD { 'password' | 'md5hash' | 'sha256hash' }

Desbloqueia o usuário especificado e define uma nova senha. Você deve especificar PASSWORD ao usar LOGIN. Quando você desbloqueia um usuário, o Amazon Redshift zera o contador de logins com falha desse usuário.

Para desbloquear um usuário, é necessário ser um superusuário ou um usuário com o privilégio ALTER USER. Assim como acontece com NOLOGIN, isso se aplica apenas a usuários baseados em senha. Essa opção só estará disponível quando a segurança de bloqueio de usuários estiver habilitada para o cluster.

Observações de uso

- Tentativa de alterar o rdsdb: não é possível alterar o usuário chamado `rdsdb`.
- Criação de uma senha desconhecida: ao usar a autenticação do AWS Identity and Access Management (IAM) para criar credenciais de usuário de banco de dados, convém criar um superusuário que possa fazer login usando apenas credenciais temporárias. Você não pode desabilitar a senha de um superusuário, mas pode criar uma senha desconhecida usando uma string de hash MD5 gerada aleatoriamente.

```
alter user iam_superuser password 'md512345678901234567801234567890l2';
```

- Definição de search_path: quando você define o parâmetro search_path com o comando ALTER USER, a modificação entra em vigor no próximo login do usuário especificado. Se quiser alterar o valor do search_path para o usuário e a sessão atuais, use o comando SET.
- Definição do fuso horário: quando você usa SET TIMEZONE com o comando ALTER USER, a modificação entra em vigor no próximo login do usuário especificado.
- Trabalho com políticas de segurança no nível da linha e de mascaramento dinâmico de dados: quando o cluster provisionado ou namespace sem servidor tem alguma política de segurança no nível da linha ou de mascaramento dinâmico de dados, os seguintes comandos são bloqueados para usuários comuns:

```
ALTER <current_user> SET enable_case_sensitive_super_attribute/
enable_case_sensitive_identifier/downcase_delimited_identifier
```

Somente superusuários e usuários com o privilégio ALTER USER podem definir essas opções de configuração. Para obter informações sobre a segurança por linha, consulte Segurança por

linha. Para obter informações sobre o mascaramento dinâmico de dados, consulte Mascaramento dinâmico de dados.

Exemplos

O exemplo a seguir permite que o usuário ADMIN tenha privilégios para criar bancos de dados:

```
alter user admin createdb;
```

O exemplo a seguir define a senha do usuário ADMIN como `adminPass9`, além de definir a data e o horário de validade da senha:

```
alter user admin password 'adminPass9'
valid until '2017-12-31 23:59';
```

O exemplo a seguir renomeia o usuário de ADMIN para SYSADMIN:

```
alter user admin rename to sysadmin;
```

O exemplo a seguir atualiza o tempo limite de sessão ociosa para um usuário para 300 segundos.

```
ALTER USER dbuser SESSION TIMEOUT 300;
```

Redefine o tempo limite da sessão ociosa do usuário. Quando você redefinir, a configuração de cluster se aplica. Você deve ser um superusuário do banco de dados para executar este comando. Para obter mais informações, consulte “Cotas e limites no Amazon Redshift” no Guia de gerenciamento de clusters do Amazon Redshift.

```
ALTER USER dbuser RESET SESSION TIMEOUT;
```

O exemplo a seguir atualiza o ID externo de um usuário chamado bob. Esse namespace é `myco_aad`. Se o namespace não estiver associado a um provedor de identidades inscrito, isso resultará em um erro.

```
ALTER USER myco_aad:bob EXTERNALID "ABC123" PASSWORD DISABLE;
```

O exemplo a seguir define o fuso horário para todas as sessões executadas por um usuário de banco de dados específico. Ele altera o fuso horário de sessões subsequentes, mas não da sessão atual.

```
ALTER USER odie SET TIMEZONE TO 'Europe/Zurich';
```

O exemplo a seguir bloqueia o usuário `data_analyst` para que ele não possa se conectar ao banco de dados:

```
ALTER USER data_analyst NOLOGIN;
```

O exemplo a seguir desbloqueia o usuário `data_analyst` e define uma nova senha. O desbloqueio do usuário zera o contador de logins com falha:

```
ALTER USER data_analyst LOGIN PASSWORD 'NewStr0ngP@ssword';
```

O exemplo a seguir define o número máximo de conexões de banco de dados que o usuário bob pode abrir.

```
ALTER USER bob CONNECTION LIMIT 10;
```

ANALYZE

Atualiza as estatísticas da tabela para uso pelo planejador de consulta.

Privilégios obrigatórios

A seguir estão os privilégios obrigatórios para ANALYZE:

- Superusuário
- Usuários com o privilégio ANALYZE
- Proprietário da relação
- Proprietário do banco de dados com o qual a tabela é compartilhada

Sintaxe

```
ANALYZE [ VERBOSE ]
[ [ table_name [ ( column_name [, ...] ) ] ]
```

```
[ PREDICATE COLUMNS | ALL  COLUMNS ]
```

Parâmetros

VERBOSE

Cláusula que retorna mensagens de informações de andamento sobre a operação ANALYZE. Esta opção é útil quando uma tabela não é especificada.

table_name

É possível analisar tabelas específicas, incluindo tabelas temporárias. É possível qualificar a tabela com o nome do esquema. Como opção, é possível especificar nome_tabela para analisar uma tabela exclusiva. Não é possível especificar mais de um nome_tabela com uma única instrução ANALYZE nome_tabela. Se você não especificar um valor para o table_name, todas as tabelas no banco de dados conectado atualmente são analisadas, incluindo as tabelas persistentes no catálogo do sistema. O Amazon Redshift ignora a análise de uma tabela se a porcentagem de linhas alteradas desde a última ANALYZE for inferior ao limite de análise. Para obter mais informações, consulte Limite de análise.

Não é necessário analisar tabelas de sistema do Amazon Redshift (tabelas STL e STV).

column_name

Se especificar um table_name, você também pode especificar uma ou mais colunas na tabela (como uma lista separada por colunas entre parênteses). Se uma lista de colunas for especificada, somente as colunas listadas serão analisadas.

PREDICATE COLUMNS | ALL COLUMNS

Cláusulas que indicam se ANALYZE deve incluir somente colunas de predicado. Especifique o parâmetro PREDICATE COLUMNS para analisar somente as colunas usadas como predicados em consultas anteriores ou que podem vir a ser usadas como predicados. Especifique ALL COLUMNS para analisar todas as colunas. ALL COLUMNS é o valor padrão.

Uma coluna é incluída no conjunto de colunas de predicado se qualquer das seguintes afirmações for verdadeira:

- A coluna foi usada em uma consulta como parte de um filtro, condição de junção ou agrupamento por cláusula.
- A coluna é uma chave de distribuição.
- A coluna faz parte de uma chave de classificação.

Se nenhuma coluna for marcada como coluna de predicado, por exemplo porque a tabela ainda não foi consultada, todas as colunas serão analisadas mesmo se o parâmetro PREDICATE COLUMNS for especificado. Quando isso acontece, o Amazon Redshift pode responder com uma mensagem, como Nenhuma coluna de predicado encontrada para "*table-name*". Analisar todas as colunas. Para obter mais informações sobre colunas de predicado, consulte Análise de tabelas.

Observações de uso

O Amazon Redshift executa automaticamente ANALYZE em tabelas criadas com os seguintes comandos:

- CREATE TABLE AS
- CREATE TEMP TABLE AS
- SELECT INTO

Não é possível analisar uma tabela externa.

Não é necessário executar o comando ANALYZE nessas tabelas quando elas são criadas. Se você as modificar, você deve analisá-las da mesma forma que outras tabelas.

Limite de análise

Para reduzir o tempo de processamento e aprimorar a performance geral do sistema, o Amazon Redshift ignorará o comando ANALYZE para uma tabela se a porcentagem de linhas alteradas desde a última vez em que o comando ANALYZE foi executado for inferior ao limite de análise especificado pelo parâmetro analyze_threshold_percent. Por padrão, `analyze_threshold_percent` é 10. Para alterar a `analyze_threshold_percent` para a sessão atual, execute o comando SET. O exemplo a seguir altera a `analyze_threshold_percent` para 20%.

```
set analyze_threshold_percent to 20;
```

Para analisar tabelas quando apenas uma quantidade pequena de linhas tiver sido alterada, defina `analyze_threshold_percent` como um número arbitrariamente pequeno. Por exemplo, se você definir `analyze_threshold_percent` como 0,01, a tabela com 100.000.000 linhas não será ignorada se pelo menos 10.000 linhas tiverem sido alteradas.

```
set analyze_threshold_percent to 0.01;
```

Se o comando ANALYZE ignorar uma tabela porque ela não atende ao limite, o Amazon Redshift retornará a mensagem a seguir.

```
ANALYZE SKIP
```

Para analisar todas as tabelas, mesmo se nenhuma linha tiver sido alterada, defina `analyze_threshold_percent` como 0.

Para visualizar os resultados das operações ANALYZE, consulte a tabela do sistema STL_ANALYZE.

Para obter mais informações sobre a análise de tabelas, consulte Análise de tabelas.

Exemplos

Analisa todas as tabelas no banco de dados TICKIT e retorna informações de andamento.

```
analyze verbose;
```

Analise somente a tabela LISTING.

```
analyze listing;
```

Analise as colunas VENUEID e VENUENAME na tabela VENUE.

```
analyze venue(venueid, venuename);
```

Analise somente colunas de predicado na tabela VENUE.

```
analyze venue predicate columns;
```

ANALYZE COMPRESSION

Executa análise de compactação e produz um relatório com a codificação sugerida de compactação para as tabelas analisadas. Para cada coluna, o relatório inclui uma estimativa da possível redução no espaço em disco para a codificação RAW.

Sintaxe

```
ANALYZE COMPRESSION
[ [ table_name ]
[ ( column_name [, ...] ) ] ]
[COMPROWS numrows]
```

Parâmetros

table_name

É possível analisar a compactação para tabelas específicas, incluindo tabelas temporárias. É possível qualificar a tabela com o nome do esquema. Como opção, é possível especificar nome_tabela para analisar uma tabela exclusiva. Se você não especificar um table_name, todas as tabelas conectadas no banco de dados no momento serão analisadas. Não é possível especificar mais de um table_name com uma única instrução ANALYZE COMPRESSION.

column_name

Se especificar um table_name, você também pode especificar uma ou mais colunas na tabela (como uma lista separada por colunas entre parênteses).

COMPROWS

Número de linhas a serem usadas como o tamanho de exemplo para análise de compactação. A análise é executada em linhas de cada fatia de dados. Por exemplo, se você especificar COMPROWS 1000000 (1.000.000) e o sistema tiver 4 fatias no total, não mais do que 250.000 linhas por fatia serão lidas e analisadas. Se COMPROWS não for especificado, o padrão do tamanho de exemplo será 100.000 por fatia. Os valores de COMPROWS menores que o padrão de 100.000 linhas por fatia são atualizados automaticamente para o valor padrão. Porém, a análise de compactação não produzirá recomendações se o valor de dados na tabela for insuficiente para produzir um exemplo significativo. Se o número de COMPROWS for maior que o número de linhas na tabela, o comando ANALYZE COMPRESSION continuará e executará a análise de compactação para todas as linhas disponíveis. O uso de COMPROWS gerará um erro se uma tabela não for especificada.

numrows

Número de linhas a serem usadas como o tamanho de exemplo para análise de compactação. O intervalo aceito para numrows é um número entre 1000 e 1000000000 (1.000.000.000).

Observações de uso

ANALYZE COMPRESSION adquire um bloqueio exclusivo de tabela que impede leituras e gravações simultâneas na tabela. Execute o comando ANALYZE COMPRESSION somente quando a tabela estiver ociosa.

Execute ANALYZE COMPRESSION para obter recomendações de esquemas de codificação de coluna com base em uma amostra do conteúdo da tabela. ANALYZE COMPRESSION é uma ferramenta de consulta e não modifica as codificações de coluna da tabela. A codificação sugerida pode ser aplicada recriando a tabela ou criando uma tabela com o mesmo esquema. Recriar uma tabela descompactada com esquemas de codificação apropriados pode reduzir significativamente o espaço ocupado no disco. Essa abordagem economiza espaço em disco e melhora a performance da consulta para workloads limitados a E/S.

ANALYZE COMPRESSION ignora a fase de análise real e retorna diretamente o tipo de codificação original em qualquer coluna designada como SORTKEY. Ele faz isso porque verificações restritas por intervalo podem ter uma má performance quando as colunas SORTKEY são muito mais compactadas que outras colunas.

Exemplos

O exemplo a seguir mostra a codificação e a redução percentual estimada somente para as colunas na tabela LISTING:

```
analyze compression listing;

  Table  |     Column     | Encoding | Est_reduction_pct
---------+----------------+----------+-------------------
 listing | listid         | az64     | 40.96
 listing | sellerid       | az64     | 46.92
 listing | eventid        | az64     | 53.37
 listing | dateid         | raw      | 0.00
 listing | numtickets     | az64     | 65.66
 listing | priceperticket | az64     | 72.94
 listing | totalprice     | az64     | 68.05
 listing | listtime       | az64     | 49.74
```

O exemplo a seguir analisa as colunas QTYSOLD, COMMISSION e SALETIME na tabela SALES.

```
analyze compression sales(qtysold, commission, saletime);
```

```
 Table |   Column   | Encoding | Est_reduction_pct
-------+------------+----------+-------------------
 sales | salesid    | N/A      | 0.00
 sales | listid     | N/A      | 0.00
 sales | sellerid   | N/A      | 0.00
 sales | buyerid    | N/A      | 0.00
 sales | eventid    | N/A      | 0.00
 sales | dateid     | N/A      | 0.00
 sales | qtysold    | az64     | 83.06
 sales | pricepaid  | N/A      | 0.00
 sales | commission | az64     | 71.85
 sales | saletime   | az64     | 49.63
```

ATTACH MASKING POLICY

Anexa uma política de mascaramento dinâmico de dados existente a uma coluna. Para obter mais informações sobre mascaramento dinâmico de dados, consulte Mascaramento dinâmico de dados.

Superusuários e usuários ou perfis que têm a função sys:secadmin podem anexar uma política de mascaramento.

Sintaxe

```
ATTACH MASKING POLICY
{
  policy_name ON relation_name
  | database_name.policy_name ON database_name.schema_name.relation_name
}
( { output_column_names | output_path } )
[ USING ( { input_column_names | input_path } ) ]
TO { user_name | ROLE role_name | PUBLIC }
[ PRIORITY priority ];
```

Parâmetros

policy_name

O nome da política de mascaramento a ser anexada.

database_name

O nome do banco de dados no qual a política e a relação são criadas. A política e a relação precisam estar no mesmo banco de dados. O banco de dados pode ser o conectado ou um banco de dados que comporte as permissões federadas do Amazon Redshift.

schema_name

O nome do esquema ao qual pertence a relação.

relation_name

O nome da relação à qual a política de mascaramento deve ser anexada.

output_column_names

Os nomes das colunas às quais a política de mascaramento se aplicará.

output_paths

O caminho completo do objeto SUPER ao qual a política de mascaramento será aplicada, inclusive o nome da coluna. Por exemplo, para uma relação com uma coluna do tipo SUPER chamada `person`, output_path pode ser `person.name.first_name`.

input_column_names

Os nomes das colunas que a política de mascaramento vai receber como entrada. Esse parâmetro é opcional. Se não for especificado, a política de mascaramento usará output_column_names como entradas.

input_paths

O caminho completo do objeto SUPER que a política de mascaramento vai utilizar como entrada. Esse parâmetro é opcional. Se não for especificado, a política de mascaramento usará output_path como entradas.

user_name

O nome do usuário ao qual a política de mascaramento será anexada. Você não pode anexar duas políticas à mesma combinação de usuário e coluna ou função e coluna. Você pode anexar uma política a um usuário e outra política ao perfil do usuário. Nesse caso, a política com maior prioridade se aplicará.

Só é possível definir uma opção entre user_name, role_name e PUBLIC em um comando ATTACH MASKING POLICY.

role_name

O nome do perfil ao qual a política de mascaramento será anexada. Não é possível anexar duas políticas ao mesmo par de coluna/perfil. Você pode anexar uma política a um usuário e outra política ao perfil do usuário. Nesse caso, a política com maior prioridade se aplicará.

Só é possível definir uma opção entre user_name, role_name e PUBLIC em um comando ATTACH MASKING POLICY.

PUBLIC

Anexa a política de mascaramento a todos os usuários que acessam a tabela. Você deve dar a outras políticas de mascaramento associadas a pares específicos de coluna/usuário ou coluna/perfil uma prioridade maior do que a política PUBLIC para que elas sejam aplicadas.

Só é possível definir uma opção entre user_name, role_name e PUBLIC em um comando ATTACH MASKING POLICY.

priority

A prioridade da política de mascaramento. Quando várias políticas de mascaramento se aplicam à consulta de um determinado usuário, a política de maior prioridade se aplica.

Você não pode anexar duas políticas diferentes à mesma coluna com prioridade igual, mesmo se as duas políticas estiverem anexadas a usuários ou funções diferentes. Você pode anexar a mesma política várias vezes ao mesmo conjunto de parâmetros de tabela, coluna de saída, coluna de entrada e prioridade, desde que o usuário ou a função à qual a política esteja anexada seja sempre diferente.

Você não pode aplicar uma política a uma coluna com a mesma prioridade de outra política anexada a essa coluna, mesmo que ela seja para perfis diferentes. Esse campo é opcional. Se você não especificar uma prioridade, a política de mascaramento será anexada com prioridade 0 por padrão.

Para o uso da ATTACH MASKING POLICY no Catálogo de Permissões Federadas do Amazon Redshift, consulte Gerenciar o controle de acesso com permissões federadas do Amazon Redshift.

ATTACH RLS POLICY

Anexar política de segurança no nível da linha de uma tabela a um ou mais usuários ou funções.

Superusuários e usuários ou funções que têm a função `sys:secadmin` podem anexar uma política.

Sintaxe

```
ATTACH RLS POLICY
{
  policy_name ON [TABLE] table_name [, ...]
  | database_name.policy_name ON [TABLE] database_name.schema_name.table_name [, ...]
}
TO { user_name | ROLE role_name | PUBLIC } [, ...]
```

Parâmetros

policy_name

O nome da política.

database_name

O nome do banco de dados no qual a política e a relação são criadas. A política e a relação precisam estar no mesmo banco de dados. O banco de dados pode ser o conectado ou um banco de dados que comporte as permissões federadas do Amazon Redshift.

schema_name

O nome do esquema ao qual pertence a relação.

table_name

A relação à qual a política de segurança no nível da linha está anexada.

TO { user_name | ROLE role_name | PUBLIC} [, ...]

Especifica se a política está anexada a um ou mais usuários ou funções especificados.

Para o uso da ATTACH RLS POLICY no Catálogo de Permissões Federadas do Amazon Redshift, consulte Gerenciar o controle de acesso com permissões federadas do Amazon Redshift.

Observações de uso

Ao trabalhar com a instrução ATTACH RLS POLICY, observe o seguinte:

- A tabela que está sendo anexada deve ter todas as colunas listadas na cláusula WITH da instrução de criação da política.
- A segurança por linha (RLS) do Amazon Redshift permite a anexação de políticas de RLS aos seguintes objetos:

  - Tabelas
  - Visualizações
  - Visualizações de vinculação tardia
  - Visualizações materializadas
- O Amazon Redshift RLS não dá suporte à anexação de políticas RLS para os seguintes objetos:
  - Tabelas de catálogo
  - Relações entre bancos de dados
  - Tabelas externas
  - Tabelas temporárias
  - Tabelas de pesquisa de políticas
  - Tabelas base de visão materializada
- Políticas de RLS anexadas a superusuários ou a usuários com a permissão `sys:secadmin` são ignoradas.

Exemplos

O exemplo a seguir anexa uma política de RLS às combinações especificadas de tabela e perfil. A política de RLS se aplica a qualquer usuário com o perfil de `analyst` ou `dbadmin` quando ele acessa a tabela tickit_category_redshift.

```
ATTACH RLS POLICY policy_concerts ON tickit_category_redshift TO ROLE analyst, ROLE
 dbadmin;
```

BEGIN

Inicia uma transação. Sinônimo de START TRANSACTION.

Quer consista em um ou mais comandos, uma transação é uma unidade de trabalho única e lógica. Geralmente, todos os comandos de uma transação são executados em um snapshot do banco de dados, cuja a hora de início é determinada pelo valor definido para o parâmetro de configuração do sistema `transaction_snapshot_begin`.

Por padrão, as operações individuais do Amazon Redshift (consultas, instruções de DDL, cargas) são confirmadas automaticamente no banco de dados. Se quiser suspender a confirmação de uma operação até que o trabalho subsequente seja concluído, você precisará abrir uma transação com a instrução BEGIN, executar os comandos e depois fechar a transação com a instrução COMMIT ou

END. Se necessário, use uma instrução ROLLBACK para abortar uma transação em andamento. Uma exceção a esse comportamento é o comando TRUNCATE que confirma a transação na qual é executado e não pode ser revertido.

Sintaxe

```
BEGIN [ WORK | TRANSACTION ] [ ISOLATION LEVEL option ] [ READ WRITE | READ ONLY ]

START TRANSACTION [ ISOLATION LEVEL option ] [ READ WRITE | READ ONLY ]

Where option is

SERIALIZABLE
| READ UNCOMMITTED
| READ COMMITTED
| REPEATABLE READ

Note: READ UNCOMMITTED, READ COMMITTED, and REPEATABLE READ have no
operational impact and map to SERIALIZABLE in Amazon Redshift. You can see database
 isolation levels on your cluster
by querying the stv_db_isolation_level table.
```

Parâmetros

WORK

Palavra-chave opcional.

TRANSACTION

Palavra-chave opcional; WORK e TRANSACTION são sinônimos.

ISOLATION LEVEL SERIALIZABLE

O isolamento serializável é compatível por padrão. Portanto, o comportamento da transação é o mesmo, esteja ou não essa sintaxe incluída na instrução. Para obter mais informações, consulte Gerenciamento de operações de gravação simultâneas. Nenhum outro nível de isolamento é compatível.

> **Note**
>
> O padrão SQL define quatro níveis de isolamento de transação para impedir dirty reads (leitura contaminada em que uma transação lê dados gravados por outra transação

simultânea não confirmada), nonrepeatable reads (leitura não repetível em que uma transação relê dados lidos anteriormente e descobre que os dados foram alterados por outra transação confirmada após a leitura inicial) e phantom reads (leitura fantasma em que uma transação executa uma consulta novamente, retorna um conjunto de linhas que satisfaz uma condição de pesquisa e descobre que o conjunto de linhas mudou por causa de outra transação confirmada recentemente):

- Leitura não confirmada: É possível que ocorram leituras contaminadas, não repetíveis e fantasmas.
- Leitura confirmada: É possível que ocorram leituras não repetíveis e fantasmas.
- Leitura repetível: É possível que ocorram leituras fantasma.
- Serializável: Impede que ocorram leituras contaminadas, não repetíveis e fantasma.

Embora seja possível usar qualquer dos quatro níveis de isolamento de transação, o Amazon Redshift processa todos os níveis de isolamento como serializáveis.

READ WRITE

Oferece permissões de leitura e gravação à transação.

READ ONLY

Oferece permissões somente de leitura à transação.

Exemplos

Os exemplos a seguir iniciam um bloco de transação serializável:

```
begin;
```

Os exemplos a seguir iniciam o bloco de transação com um nível de isolamento serializável e permissões de leitura e gravação:

```
begin read write;
```

CALL

Executa um procedimento armazenado. O comando CALL deve incluir o nome do procedimento e os valores do argumento de entrada. É obrigatório chamar um procedimento armazenado usando a instrução CALL.

> **Note**
>
> CALL não pode fazer parte de qualquer consulta regular.

Sintaxe

```
CALL sp_name ( [ argument ] [, ...] )
```

Parâmetros

sp_name

O nome do procedimento a ser executado.

argument

O valor do argumento de entrada. Esse parâmetro também pode ser um nome de função, por exemplo, `pg_last_query_id()`. Não é possível usar consultas como argumentos para CALL.

Observações de uso

Os procedimentos armazenados do Amazon Redshift oferecem suporte a chamadas aninhadas e recursivas, conforme descrito a seguir. Além disso, verifique se o seu suporte ao driver está atualizado, também descrito a seguir.

Tópicos

- Chamadas aninhadas
- Suporte a drivers

Chamadas aninhadas

Os procedimentos armazenados do Amazon Redshift oferecem suporte a chamadas aninhadas e recursivas. O número máximo de níveis de aninhamento permitido é 16. Chamadas aninhadas podem encapsular lógica de negócios em procedimentos menores, que podem ser compartilhados por vários chamadores.

Se você chamar um procedimento aninhado que tem parâmetros de saída, o procedimento interno deverá definir argumentos INOUT. Nesse caso, o procedimento interno será enviado em uma variável não constante. Argumentos OUT não são permitidos. Esse comportamento ocorre pois uma variável é necessária para conter a saída da chamada interna.

A relação entre os procedimentos interno e externo é registrada em log na coluna `from_sp_call` de SVL_STORED_PROC_CALL.

O exemplo a seguir mostra as variáveis enviadas a uma chamada de procedimento aninhado por meio de argumentos INOUT.

```
CREATE OR REPLACE PROCEDURE inner_proc(INOUT a int, b int, INOUT c int) LANGUAGE
 plpgsql
AS $$
BEGIN
  a := b * a;
  c := b * c;
END;
$$;

CREATE OR REPLACE PROCEDURE outer_proc(multiplier int) LANGUAGE plpgsql
AS $$
DECLARE
  x int := 3;
  y int := 4;
BEGIN
  DROP TABLE IF EXISTS test_tbl;
  CREATE TEMP TABLE test_tbl(a int, b varchar(256));
  CALL inner_proc(x, multiplier, y);
  insert into test_tbl values (x, y::varchar);
END;
$$;

CALL outer_proc(5);
```

```
SELECT * from test_tbl;
 a  | b
----+----
 15 | 20
(1 row)
```

Suporte a drivers

Recomendamos atualizar seus drivers de Java Database Connectivity (JDBC) e Open Database Connectivity (ODBC) para a versão mais recente com suporte para procedimentos armazenados do Amazon Redshift.

Você poderá usar o driver existente se a sua ferramenta de cliente usar operações da API de driver que transmitem a instrução CALL para o servidor. Parâmetros de saída, se houver, são retornados como um conjunto de resultados de uma linha.

As versões mais recentes dos drivers JDBC e ODBC do Amazon Redshift têm suporte a metadados para descoberta de procedimentos armazenados. Elas também têm suporte a `CallableStatement` para aplicativos Java personalizados. Para obter mais informações, consulte "Conectar-se a um cluster Amazon Redshift usando ferramentas de cliente SQL" no Guia de gerenciamento de clusters do Amazon Redshift.

Os exemplos a seguir mostram como usar diferentes operações da API do driver de JDBC para chamadas de procedimentos armazenados.

```
void statement_example(Connection conn) throws SQLException {
  statement.execute("CALL sp_statement_example(1)");
}

void prepared_statement_example(Connection conn) throws SQLException {
  String sql = "CALL sp_prepared_statement_example(42, 84)";
  PreparedStatement pstmt = conn.prepareStatement(sql);
  pstmt.execute();
}

void callable_statement_example(Connection conn) throws SQLException {
  CallableStatement cstmt = conn.prepareCall("CALL sp_create_out_in(?,?)");
  cstmt.registerOutParameter(1, java.sql.Types.INTEGER);
  cstmt.setInt(2, 42);
  cstmt.executeQuery();
  Integer out_value = cstmt.getInt(1);
```

```
}
```

Exemplos

O exemplo a seguir chama o nome de procedimento `test_spl`.

```
call test_sp1(3,'book');
INFO:  Table "tmp_tbl" does not exist and will be skipped
INFO:  min_val = 3, f2 = book
```

O exemplo a seguir chama o nome de procedimento `test_spl2`.

```
call test_sp2(2,'2019');

         f2          | column2
---------------------+---------
 2019+2019+2019+2019 | 2
(1 row)
```

CANCEL

Cancela uma consulta de banco de dados que está sendo executada no momento.

O comando CANCEL requer o ID do processo ou o ID da sessão da consulta que está sendo executada e exibe uma mensagem de confirmação para confirmar que a consulta foi cancelada.

Privilégios obrigatórios

A seguir estão os privilégios obrigatórios para CANCEL:

- Superusuário que cancele sua própria consulta
- Superusuário que cancele a consulta de um usuário
- Usuários com o privilégio CANCEL que cancelem a consulta de um usuário
- Usuário que cancele sua própria consulta

Sintaxe

```
CANCEL process_id [ 'message' ]
```

Parâmetros

process_id

Para cancelar uma consulta em execução em um cluster do Amazon Redshift, use o `pid` (ID do processo) de STV_RECENTS que corresponde à consulta a ser cancelada.

Para cancelar uma consulta em execução em um grupo de trabalho do Amazon Redshift sem servidor, use o `session_id` de SYS_QUERY_HISTORY que corresponde à consulta a ser cancelada.

'mensagem'

Mensagem de confirmação opcional exibida quando o cancelamento da consulta é concluído. Se você não especificar uma mensagem, o Amazon Redshift exibirá a mensagem padrão como verificação. Você deve colocar a mensagem entre aspas simples.

Observações de uso

Não é possível cancelar uma consulta especificando um ID de consulta; é necessário especificar o ID de processo (PID) ou o ID da sessão da consulta. Somente é possível cancelar consultas atualmente em execução pelo seu usuário. Superusuários podem cancelar todas as consultas.

Se as consultas em várias sessões mantiverem bloqueios na mesma tabela, você poderá usar a função PG_TERMINATE_BACKEND para encerrar uma das sessões. Isso força todas as transações em execução na sessão encerrada a liberar todos os bloqueios e reverter a transação. Consulte a tabela de sistema STV_LOCKS para exibir os bloqueios atuais.

Depois de alguns eventos internos, o Amazon Redshift pode reiniciar uma sessão ativa e atribuir um novo PID. Se o PID foi alterado, a seguinte mensagem de erro pode aparecer.

```
Session <PID> does not exist. The session PID might have changed. Check the
 stl_restarted_sessions system table for details.
```

Para encontrar o PID, consulte a tabela de sistema STL_RESTARTED_SESSIONS e o filtro na coluna `oldpid`.

```
select oldpid, newpid from stl_restarted_sessions where oldpid = 1234;
```

Exemplos

Para cancelar uma consulta em execução em um cluster do Amazon Redshift, recupere primeiro o ID de processo para a consulta a ser cancelada. Para determinar os IDs de processo para todas as consultas atualmente em execução, digite o seguinte comando:

```
select pid, starttime, duration,
trim(user_name) as user,
trim (query) as querytxt
from stv_recents
where status = 'Running';

pid |         starttime          | duration |   user   |    querytxt
-----+----------------------------+----------+----------+-----------------
802 | 2008-10-14 09:19:03.550885 |      132 | dwuser | select
venuename from venue where venuestate='FL', where venuecity not in
('Miami' , 'Orlando');
834 | 2008-10-14 08:33:49.473585 |  1250414 | dwuser | select *
from listing;
964 | 2008-10-14 08:30:43.290527 |   326179 | dwuser | select
sellerid from sales where qtysold in (8, 10);
```

Verifique o texto da consulta para determinar qual ID de processo (PID) corresponde à consulta que você deseja cancelar.

Digite o seguinte comando a ser usado com o PID 802 para cancelar a consulta:

```
cancel 802;
```

A sessão em que a consulta estava sendo executada exibirá a seguinte mensagem:

```
ERROR:  Query (168) cancelled on user's request
```

onde 168 é o ID de consulta (não o ID de processo usado para cancelar a consulta).

Como alternativa, você pode especificar uma mensagem de confirmação personalizada a ser exibida no lugar da mensagem padrão. Para especificar uma mensagem personalizada, inclua sua mensagem entre aspas simples no final do comando CANCEL:

```
cancel 802 'Long-running query';
```

A sessão em que a consulta estava sendo executada exibirá a seguinte mensagem:

```
ERROR:  Long-running query
```

CLOSE

(Opcional) Fecha todos os recursos livres associados com um cursor aberto. COMMIT, END e ROLLBACK fecham automaticamente o cursor, para que não seja necessário usar o comando CLOSE para fechá-lo explicitamente.

Para obter mais informações, consulte DECLAREFETCH.

Sintaxe

```
CLOSE cursor
```

Parâmetros

cursor

Nome do cursor a ser fechado.

Exemplo de CLOSE

Os comandos a seguir fecham o cursor e realizam uma confirmação, que finaliza a transação:

```
close movie_cursor;
commit;
```

COMMENT

Cria ou altera um comentário sobre um objeto do banco de dados.

Sintaxe

```
COMMENT ON
{
TABLE object_name |
COLUMN object_name.column_name |
CONSTRAINT constraint_name ON table_name |
```

```
DATABASE object_name |
VIEW object_name
}
IS 'text' | NULL
```

Parâmetros

nome_objeto

Nome do objeto do banco de dados que está sendo comentado. Você pode adicionar um comentário aos seguintes objetos:

- TABLE
- COLUMN (também utiliza um nome_coluna).
- CONSTRAINT (também utiliza um nome_restrição e um nome_tabela).
- DATABASE
- VIEW
- SCHEMA

IS 'text' | NULL

O texto do comentário que você deseja adicionar ou substituir pelo objeto especificado. A string text é o tipo de dados TEXT. Coloque o comentário entre aspas simples. Defina o valor como NULL para remover o texto do comentário.

column_name

Nome da coluna que está sendo comentada. Parâmetro de COLUMN. Acompanha uma tabela específica em `object_name`.

nome_restrição

Nome da restrição que está sendo comentada. Parâmetro de CONSTRAINT.

table_name

Nome de uma tabela que contém a restrição. Parâmetro de CONSTRAINT.

Observações de uso

É necessário ser um superusuário ou o proprietário de um objeto de banco de dados para adicionar ou atualizar um comentário.

Comentários sobre bancos de dados podem ser aplicados somente ao banco de dados atual. Uma mensagem de advertência é exibida se você tentar fazer comentários sobre um banco de dados diferente. A mesma mensagem é exibida para comentários sobre bancos de dados que não existem.

Comentários em tabelas externas, colunas externas e colunas de visões de vinculação tardia não são compatíveis.

Exemplos

O exemplo a seguir adiciona um comentário à tabela SALES.

```
COMMENT ON TABLE sales IS 'This table stores tickets sales data';
```

O exemplo a seguir exibe o comentário na tabela SALES.

```
select obj_description('public.sales'::regclass);

obj_description
------------------------------------
This table stores tickets sales data
```

O exemplo a seguir remove um comentário da tabela SALES.

```
COMMENT ON TABLE sales IS NULL;
```

O exemplo a seguir adiciona um comentário à coluna EVENTID da tabela SALES.

```
COMMENT ON COLUMN sales.eventid IS 'Foreign-key reference to the EVENT table.';
```

O exemplo a seguir exibe um comentário na coluna EVENTID (coluna número 5) da tabela SALES.

```
select col_description( 'public.sales'::regclass, 5::integer );

col_description
---------------------------------------
Foreign-key reference to the EVENT table.
```

O exemplo a seguir adiciona um comentário descritivo à tabela EVENT.

```
comment on table event is 'Contains listings of individual events.';
```

Para visualizar os comentários, faça uma consulta no catálogo do sistema PG_DESCRIPTION. O exemplo a seguir retorna a descrição da tabela EVENT.

```
select * from pg_catalog.pg_description
where objoid =
(select oid from pg_class where relname = 'event'
and relnamespace =
(select oid from pg_catalog.pg_namespace where nspname = 'public') );

objoid | classoid | objsubid | description
-------+----------+----------+---------------------------------------
116658 |     1259 |        0 | Contains listings of individual events.
```

COMMIT

Confirma a transação atual no banco de dados. Este comando realiza as atualizações de bancos de dados a partir da transação permanente.

Sintaxe

```
COMMIT [ WORK | TRANSACTION ]
```

Parâmetros

WORK

Palavra-chave opcional. Essa palavra-chave não é permitida em um procedimento armazenado.

TRANSACTION

Palavra-chave opcional. WORK e TRANSACTION são sinônimos. Nenhuma delas é permitida em um procedimento armazenado.

Para obter informações sobre como usar COMMIT em um procedimento armazenado, consulte Gerenciamento de transações.

Exemplos

Cada um dos exemplos a seguir confirma a transação atual no banco de dados:

```
commit;
```

```
commit work;
```

```
commit transaction;
```

COPY

A criptografia do lado do cliente para os comandos COPY e UNLOAD não estará mais disponível para novos clientes a partir de 30 de abril de 2025. Se você usou a criptografia do lado do cliente com os comandos COPY e UNLOAD nos 12 meses anteriores a 30 de abril de 2025, poderá continuar usando a criptografia do lado do cliente com os comandos COPY ou UNLOAD até 30 de abril de 2026. Após 30 de abril de 2026, você não poderá usar a criptografia do lado do cliente para COPY e UNLOAD. Recomendamos que você passe a usar a criptografia do lado do servidor para COPY e UNLOAD o mais rápido possível. Se você já usa a criptografia do lado do servidor para COPY e UNLOAD, não há nenhuma alteração a fazer e você pode continuar a usá-la sem alterar suas consultas. Para ter mais informações sobre criptografia para COPY e UNLOAD, consulte o parâmetro ENCRYPTED abaixo.

Carrega dados em uma tabela de arquivos de dados ou de uma tabela do Amazon DynamoDB. Os arquivos podem estar localizados em um bucket do Amazon Simple Storage Service (Amazon S3), em um cluster do Amazon EMR ou em um host remoto acessado com o uso de uma conexão SSH (Secure Shell).

> **Note**
>
> As tabelas externas do Amazon Redshift Spectrum são de somente leitura. Não é possível usar o comando COPY em uma tabela externa.

O comando COPY acrescenta os dados de entrada na forma de linhas adicionais na tabela.

O tamanho máximo de uma única linha de entrada de qualquer origem é 4 MB.

Tópicos

- Permissões obrigatórias
- Sintaxe de COPY
- Parâmetros necessários
- Parâmetros opcionais
- Observações sobre uso e recursos adicionais para o comando COPY
- Exemplos de comando COPY
- COPY JOB
- COPIAR com MODELO
- Referência de parâmetro de COPY
- Observações de uso
- Exemplos de COPY

Permissões obrigatórias

Para usar o comando COPY, você deve ter o privilégio INSERT para a tabela do Amazon Redshift.

Sintaxe de COPY

```
COPY table-name
[ column-list ]
FROM data_source
authorization
[ [ FORMAT ] [ AS ] data_format ]
[ parameter [ argument ] [, ... ] ]
```

Você pode realizar uma operação COPY com apenas três parâmetros: um nome de tabela, uma fonte de dados e uma autorização para acessar os dados.

O Amazon Redshift estende a funcionalidade do comando COPY para que seja possível carregar dados em vários formatos de dados de várias fontes de dados, controlar o acesso aos dados a serem carregados, gerenciar transformações de dados e gerenciar a operação de carregamento.

As seções a seguir apresentam os parâmetros do comando COPY obrigatório. agrupando os parâmetros opcionais por função. Elas também descrevem cada parâmetro e explicam como

diversas opções funcionam juntas. Você pode ir diretamente à descrição de um parâmetro usando a lista de parâmetros alfabética.

Parâmetros necessários

O comando COPY exige três elementos:

- Table Name
- Data Source
- Authorization

O comando COPY mais simples usa o formato a seguir.

```
COPY table-name
FROM data-source
authorization;
```

O exemplo a seguir criar uma tabela chamada CATDEMO e carrega a tabela com dados de exemplo de um arquivo de dados no Amazon S3 chamado `category_pipe.txt`.

```
create table catdemo(catid smallint, catgroup varchar(10), catname varchar(10), catdesc
 varchar(50));
```

No exemplo a seguir, a origem dos dados do comando COPY é um arquivo de dados chamado `category_pipe.txt` na pasta `tickit` de um bucket do Amazon S3 chamado `redshift-downloads`. O comando COPY tem autorização para acessar o bucket do Amazon S3 por meio de uma função do AWS Identity and Access Management (IAM). Se o cluster tiver uma função do IAM existente com permissão para acessar o Amazon S3 anexado, você poderá substituir o nome do recurso da Amazon (ARN) no comando COPY a seguir e executá-lo.

```
copy catdemo
from 's3://redshift-downloads/tickit/category_pipe.txt'
iam_role 'arn:aws:iam::<aws-account-id>:role/<role-name>'
region 'us-east-1';
```

Consulte instruções completas sobre como usar comandos COPY para carregar dados de exemplo, incluindo instruções para carregar dados de outras regiões da AWS em Carregar dados de exemplo do Amazon S3 no Guia de conceitos básicos do Amazon Redshift.

table-name

O nome da tabela de destino do comando COPY. A tabela já deve existir no banco de dados. A tabela pode ser temporária ou persistente. O comando COPY acrescenta os novos dados de entrada a todas as linhas existentes na tabela.

FROM data-source

O local dos dados de origem a serem carregados na tabela de destino. É possível especificar um arquivo de manifesto com algumas fontes de dados.

O repositório de dados mais usado é um bucket do Amazon S3. Você também pode carregar os arquivos de dados localizados em um cluster do Amazon EMR, uma instância do Amazon EC2 ou um host remoto que o cluster pode acessar usando uma conexão SSH, ou ainda pode carregar diretamente de uma tabela do DynamoDB.

- COPY do Amazon S3
- COPY do Amazon EMR
- COPY de host remoto (SSH)
- COPY do Amazon DynamoDB

Autorização

Uma cláusula que indica o método que o seu cluster usa na autenticação e autorização para acessar outros recursos da AWS. O comando COPY precisa de autorização para acessar dados em outro recurso da AWS, inclusive em Amazon S3, Amazon EMR, Amazon DynamoDB e Amazon EC2. Essa autorização é fornecida por meio de referência ao perfil do IAM que é associada ao cluster.

- Parâmetros de autorização
- Controle de acesso com base em função

Parâmetros opcionais

Opcionalmente, você pode especificar como o COPY mapeia os dados de campo para colunas na tabela de destino, definir atributos de dados de origem para permitir que o comando COPY leia e analise corretamente os dados de origem e gerenciar quais operações o comando COPY executa durante o processo de carregamento.

- Opções de mapeamento da coluna
- Parâmetros de formato de dados

- Parâmetros da conversão de dados
- Operações de carregamento de dados

Mapeamento de colunas

Por padrão, COPY insere valores de campo nas colunas da tabela de destino na mesma ordem dos campos ocorridos nos arquivos de dados. Se a ordem de coluna padrão não funcionar, você poderá especificar uma lista de colunas ou usar expressões JSONPath para mapear campos de dados de origem para as colunas de destino.

- Column List
- JSONPaths File

Parâmetros de formato de dados

Você pode carregar dados de arquivos de texto em formato de largura fixa, delimitado por caractere, CSV (Comma-Separated Values) ou JSON, ou de arquivos Avro.

Por padrão, o comando COPY espera que os dados de origem estejam em arquivos de texto UTF-8 delimitados por caractere. O delimitador padrão é um caractere de barra (|). Se os dados de origem estiverem em outro formato, use os parâmetros a seguir para especificar o formato de dados.

- FORMAT
- CSV
- DELIMITER
- FIXEDWIDTH
- SHAPEFILE
- AVRO
- JSON format for COPY
- ENCRYPTED
- BZIP2
- GZIP
- LZOP
- PARQUET
- ORC

- ZSTD

Parâmetros da conversão de dados

À medida que carrega a tabela, COPY tenta converter implicitamente as strings nos dados de origem no tipo de dados da coluna de destino. Se precisar especificar uma conversão diferente do comportamento padrão, ou se a conversão padrão resultar em erros, você poderá gerenciar conversões de dados especificando os parâmetros a seguir.

- ACCEPTANYDATE
- ACCEPTINVCHARS
- BLANKSASNULL
- DATEFORMAT
- EMPTYASNULL
- ENCODING
- ESCAPE
- EXPLICIT_IDS
- FILLRECORD
- IGNOREBLANKLINES
- IGNOREHEADER
- NULL AS
- REMOVEQUOTES
- ROUNDEC
- TIMEFORMAT
- TRIMBLANKS
- TRUNCATECOLUMNS

Operações de carregamento de dados

Gerencie o comportamento padrão da operação de carregamento para solucionar problemas ou reduzir o tempo de carregamento especificando os parâmetros a seguir.

- COMPROWS
- COMPUPDATE

- IGNOREALLERRORS
- MAXERROR
- NOLOAD
- STATUPDATE

Observações sobre uso e recursos adicionais para o comando COPY

Para obter mais informações sobre como usar o comando COPY, consulte os seguintes tópicos:

- Observações de uso
- Tutorial: Carregar dados do Amazon S3
- Práticas recomendadas do Amazon Redshift para carregamento de dados
- Carregar tabelas com o comando COPY
  - Carregar dados do Amazon S3
  - Carregar dados do Amazon EMR
  - Carregamento de dados de hosts remotos
  - Carregar dados de uma tabela do Amazon DynamoDB
- Solução de problemas de carregamento de dados

Exemplos de comando COPY

Para obter mais exemplos que mostram como COPIAR de várias fontes, em formatos diferentes e com diferentes opções de CÓPIA, consulte Exemplos de COPY.

COPY JOB

Para obter mais informações sobre esse comando, consulte Criar uma integração de eventos do S3 para copiar automaticamente arquivos de buckets do Amazon S3.

Gerencia comandos COPY que carregam dados em uma tabela. O comando COPY JOB é uma extensão do comando COPY e automatiza o carregamento de dados dos buckets do Amazon S3. Quando você cria um trabalho COPY, o Amazon Redshift detecta quando são criados arquivos do Amazon S3 em um caminho especificado e os carrega automaticamente sem sua intervenção. Os mesmos parâmetros usados no comando COPY original são usados ao carregar os dados. O Amazon Redshift monitora os arquivos carregados (com base no nome do arquivo) para verificar se eles são carregados apenas uma vez.

> **Note**
>
> Para obter informações sobre o comando COPY, incluindo uso, parâmetros e permissões, consulteCOPY.

Permissão obrigatória

Para usar o comando COPY JOB, você deve ter uma das seguintes permissões, além de todas as permissões necessárias para usar COPY:

- Superusuário
- Todas as seguintes:
  - A permissão relevante CREATE, ALTER ou DROP com escopo para COPY JOBS no banco de dados para o qual você deseja COPIAR.
  - Permissão USAGE referente ao esquema para o qual você deseja COPIAR ou permissão USAGE com escopo para esquemas no banco de dados para o qual você deseja COPIAR.
  - Permissão INSERT para a tabela para a qual você deseja COPIAR ou a permissão INSERT com escopo para tabelas no esquema ou banco de dados para o qual você deseja COPIAR.

O perfil do IAM especificado com o comando COPY deve ter permissão para acessar os dados a serem carregados. Para obter mais informações, consulte Permissões do IAM para COPY, UNLOAD e CREATE LIBRARY.

Sintaxe

Crie um trabalho de cópia. Os parâmetros do comando COPY são salvos com o trabalho de cópia.

Não é possível executar COPY JOB CREATE dentro do escopo de um bloco de transação.

```
COPY copy-command JOB CREATE job-name
[AUTO ON | OFF]
```

Altere a configuração de um trabalho de cópia.

```
COPY JOB ALTER job-name
[AUTO ON | OFF]
```

Execute um trabalho de cópia. Os parâmetros do comando COPY armazenados são usados.

```
COPY JOB RUN job-name
```

Liste todos os trabalhos de cópia.

```
COPY JOB LIST
```

Mostre os detalhes de um trabalho de cópia.

```
COPY JOB SHOW job-name
```

Exclua um trabalho de cópia.

Não é possível executar COPY JOB DROP dentro do escopo de um bloco de transação.

```
COPY JOB DROP job-name
```

Parâmetros

copy-command

Um comando COPY que carrega dados do Amazon S3 para o Amazon Redshift. A cláusula contém parâmetros de COPY que definem o bucket do Amazon S3, a tabela de destino, o perfil do IAM e outros parâmetros usados ao carregar dados. Todos os parâmetros do comando COPY para um carregamento de dados do Amazon S3 são compatíveis, exceto:

- COPY JOB não ingere arquivos preexistentes na pasta apontada pelo comando COPY. Somente arquivos criados após o carimbo de data e hora da criação de COPY JOB são ingeridos.
- Não é possível especificar um comando COPY com as opções MAXERROR ou IGNOREALLERRORS.
- Você não pode especificar um arquivo de manifesto. COPY JOB exige um local designado no Amazon S3 para monitorar a criação de arquivos.
- Você não pode especificar um comando COPY com tipos de autorização como chaves de acesso e secretas. Somente comandos COPY que usam o parâmetro `IAM_ROLE` para autorização são compatíveis. Para obter mais informações, consulte Parâmetros de autorização.

- O COPY JOB não oferece suporte ao perfil do IAM padrão associado com o cluster. Você deve especificar o `IAM_ROLE` no comando COPY.

Para obter mais informações, consulte COPY do Amazon S3.

job-name

O nome do trabalho usado para fazer referência ao trabalho COPY. O job-name não pode conter um hífen (-).

[AUTO ON | OFF]

Cláusula que indica se os dados do Amazon S3 são carregados automaticamente nas tabelas do Amazon Redshift.

- Quando `ON`, o Amazon Redshift monitora o caminho de origem do Amazon S3 para arquivos recém-criados e, se encontrado, um comando COPY é executado com os parâmetros de COPY na definição do trabalho. Esse é o padrão.
- Quando `OFF`, o Amazon Redshift não executa COPY JOB automaticamente.

Observações de uso

As opções do comando COPY não são validadas até o tempo de execução. Por exemplo, um `IAM_ROLE` inválido ou uma fonte de dados do Amazon S3 resulta em erros de tempo de execução quando COPY JOB é iniciado.

Se o cluster estiver pausado, COPY JOBS não serão executados.

Para consultar arquivos de comando COPY carregados e erros de carregamento, consulte STL_LOAD_COMMITS, STL_LOAD_ERRORS e STL_LOADERROR_DETAIL. Para obter mais informações, consulte Como verificar se os dados foram carregados corretamente.

Não é possível usar COPY JOBS em bancos de dados com ETL zero, pois eles operam no modo somente leitura.

Exemplos

O exemplo a seguir mostra a criação de um COPY JOB para carregar dados de um bucket do Amazon S3.

```
COPY public.target_table
```

```
FROM 's3://amzn-s3-demo-bucket/staging-folder'
IAM_ROLE 'arn:aws:iam::123456789012:role/MyLoadRoleName'
JOB CREATE my_copy_job_name
AUTO ON;
```

COPIAR com MODELO

É possível usar modelos do Redshift com comandos COPY para simplificar a sintaxe dos comandos e manter a consistência nas operações de carregamento de dados. Em vez de especificar os mesmos parâmetros de formatação repetidamente, defina-os uma vez em um modelo e faça referência ao modelo em seus comandos COPY. Quando você usa um modelo, o comando COPY associa os parâmetros do modelo a quaisquer parâmetros especificados diretamente no comando. Se o mesmo parâmetro aparecer no modelo e no comando, o parâmetro do comando terá precedência. Para obter mais informações, consulte CREATE TEMPLATE.

Os modelos para o comando COPY podem ser criados com:

- Parâmetros de formato de dados
- Parâmetros de compactação de arquivo
- Parâmetros da conversão de dados
- Operações de carregamento de dados

Para ver uma lista completa de parâmetros compatíveis, consulte o comando COPY.

Permissão obrigatória

Para usar um modelo em um comando COPY, você deve ter:

- Todas as permissões necessárias para executar o comando COPY (consulte Permissões obrigatórias).
- Uma das seguintes permissões de modelo:
  - Privilégios de superusuário.
  - Privilégio USAGE no modelo e privilégio USAGE no esquema que contém o modelo.

Sintaxe

```
COPY target_table FROM 's3://...'
```

```
authorization
[ option, ...]
USING TEMPLATE [database_name.][schema_name.]template_name;
```

Parâmetros

database_name

(Opcional) O nome do banco de dados no qual o modelo existe. Se não especificado, será usado o banco de dados atual.

schema_name

(Opcional) O nome do esquema no qual o modelo existe. Se não for especificado, o modelo será pesquisado no caminho de pesquisa atual.

template_name

O nome do modelo a ser usado em COPY.

Observações de uso

- Os parâmetros específicos do comando (origem, destino e autorização) ainda devem ser especificados no comando COPY.
- Os modelos não podem conter especificações de arquivo de manifesto para comandos COPY.

Exemplos

Os seguintes exemplos mostram como criar um modelo e usá-lo em comandos COPY:

```
CREATE TEMPLATE public.test_template FOR COPY AS
CSV DELIMITER '|' IGNOREHEADER 1 MAXERROR 100;

COPY public.target_table
FROM 's3://amzn-s3-demo-bucket/staging-folder'
IAM_ROLE 'arn:aws:iam::123456789012:role/MyLoadRoleName'
USING TEMPLATE public.test_template;
```

Quando existe um parâmetro no modelo e no comando, o parâmetro do comando tem precedência. Neste exemplo, se o modelo `public.test_template` contiver `DELIMITER '|'`, mas o comando

COPY especificar `DELIMITER ','`, o delimitador de vírgula (`,`) do comando será usado em vez do delimitador de barra vertical (`|`) do modelo.

```
COPY public.target_table
FROM 's3://amzn-s3-demo-bucket/staging-folder'
IAM_ROLE 'arn:aws:iam::123456789012:role/MyLoadRoleName'
DELIMITER ','
USING TEMPLATE public.test_template;
```

Referência de parâmetro de COPY

COPY tem vários parâmetros que podem ser usados em muitas situações. No entanto, nem todos os parâmetros são compatíveis com cada situação. Por exemplo, para carregar arquivos ORC ou PARQUET, há um número limitado de parâmetros compatíveis. Para obter mais informações, consulte COPY de formatos de dados colunar.

Tópicos

- Fontes de dados
- Parâmetros de autorização
- Opções de mapeamento da coluna
- Parâmetros de formato de dados
- Parâmetros de compactação de arquivo
- Parâmetros da conversão de dados
- Operações de carregamento de dados
- Lista de parâmetros alfabética

Fontes de dados

Você pode carregar dados de arquivos de texto em um bucket do Amazon S3, em um cluster do Amazon EMR ou em um host remoto que o cluster pode acessar usando uma conexão SSH. Você também pode carregar dados diretamente de uma tabela do DynamoDB.

O tamanho máximo de uma única linha de entrada de qualquer origem é 4 MB.

Para exportar dados de uma tabela para um conjunto de arquivos em um Amazon S3, use o comando UNLOAD.

Tópicos

- COPY do Amazon S3
- COPY do Amazon EMR
- COPY de host remoto (SSH)
- COPY do Amazon DynamoDB

COPY do Amazon S3

Para carregar dados de arquivos localizados em um ou mais buckets do S3, use a cláusula FROM para indicar como o COPY localiza os arquivos no Amazon S3. Você pode fornecer o caminho do objeto para os arquivos de dados como parte da cláusula FROM ou a localização de um arquivo manifesto que contenha uma lista de caminhos de objeto do Amazon S3. COPY de Amazon S3 usa uma conexão HTTPS. Verifique se os intervalos de IP do S3 foram adicionados à sua lista de permissões. Para saber mais sobre os intervalos de IP do S3 necessários, consulte Isolamento de rede.

> **Important**
>
> Se os buckets do Amazon S3 que mantêm os arquivos de dados não residirem na mesma região da AWS que o cluster, use o parâmetro REGION para especificar a região na qual os dados estão localizados.

Tópicos

- Sintaxe
- Exemplos
- Parâmetros opcionais
- Parâmetros incompatíveis

Sintaxe

```
FROM { 's3://objectpath' | 's3://manifest_file' }
authorization
| MANIFEST
| ENCRYPTED
```

```
| REGION [AS] 'aws-region'
| optional-parameters
```

Exemplos

O exemplo a seguir usa um caminho de objeto para carregar dados do Amazon S3.

```
copy customer
from 's3://amzn-s3-demo-bucket/customer'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

O exemplo a seguir usa um arquivo manifesto para carregar dados do Amazon S3.

```
copy customer
from 's3://amzn-s3-demo-bucket/cust.manifest'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
manifest;
```

Parâmetros

FROM

A origem dos dados a serem carregados. Para obter mais informações sobre a codificação do arquivo do Amazon S3, consulte Parâmetros da conversão de dados.

's3://copy_from_s3_objectpath'

Especifica o caminho para os objetos do Amazon S3 que contêm os dados por exemplo, `'s3://amzn-s3-demo-bucket/custdata.txt'`. O parâmetro s3://copy_from_s3_objectpath pode referenciar um único arquivo ou um conjunto de objetos ou pastas que tenham o mesmo prefixo de chaves. Por exemplo, o nome `custdata.txt` é um prefixo de chaves que referencia um número de arquivos físicos: `custdata.txt`,`custdata.txt.1`, `custdata.txt.2`, `custdata.txt.bak` e assim por diante. O prefixo de chaves também pode referenciar um número de pastas. Por exemplo, `'s3://amzn-s3-demo-bucket/custfolder'` se refere às pastas `custfolder`, `custfolder_1`, `custfolder_2` e assim por diante. Se um prefixo de chaves referencia várias pastas, todos os arquivos nas pastas são carregados. Se um prefixo de chaves corresponder a um arquivo, bem como a uma pasta, como `custfolder.log`, COPY também tentará carregar o arquivo. Se um prefixo de chaves puder resultar na tentativa de COPY de carregar arquivos indesejados, use um arquivo manifesto. Para obter mais informações, consulte copy_from_s3_manifest_file, a seguir.

> **Important**
>
> Se o bucket do S3 que mantém os arquivos de dados não residir na mesma região da AWS que o cluster, use o parâmetro REGION para especificar a região na qual os dados estão localizados.

Para obter mais informações, consulte Carregar dados do Amazon S3.

's3://copy_from_s3_manifest_file'

Especifica a chave de objeto do Amazon S3 para um arquivo manifesto que lista os arquivos de dados a serem carregados. O argumento 's3://copy_from_s3_manifest_file' deve referenciar explicitamente um único arquivo, por exemplo, `'s3://amzn-s3-demo-bucket/manifest.txt'`. Ele não pode fazer referência a um prefixo de chaves.

O manifesto é um arquivo de texto em formato JSON que lista o URL de cada arquivo que deve ser carregado do Amazon S3. O URL inclui o nome do bucket e o caminho de objeto completo do arquivo. Os arquivos especificados no manifesto podem estar em buckets diferentes, mas todos os buckets devem estar na mesma região da AWS que o cluster do Amazon Redshift. Se for listado duas vezes, o arquivo será carregado duas vezes. O exemplo a seguir mostra o JSON de um manifesto que carrega três arquivos.

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket1/custdata.1","mandatory":true},
    {"url":"s3://amzn-s3-demo-bucket1/custdata.2","mandatory":true},
    {"url":"s3://amzn-s3-demo-bucket2/custdata.1","mandatory":false}
  ]
}
```

As aspas duplas são obrigatórias e devem ser simples (0x22), e não inclinadas ou "inteligentes". Cada entrada no manifesto também pode incluir um sinalizador `mandatory`. Se `mandatory` estiver definido como `true`, COPY será encerrado se não encontrar o arquivo dessa entrada; do contrário, COPY continuará. O valor padrão para `mandatory` é `false`.

Ao carregar arquivos de dados em ORC ou em formato Parquet, um campo `meta` é necessário, conforme exibido no seguinte exemplo.

```
{
```

```
   "entries":[
      {
         "url":"s3://amzn-s3-demo-bucket1/orc/2013-10-04-custdata",
         "mandatory":true,
         "meta":{
            "content_length":99
         }
      },
      {
         "url":"s3://amzn-s3-demo-bucket2/orc/2013-10-05-custdata",
         "mandatory":true,
         "meta":{
            "content_length":99
         }
      }
   ]
}
```

O arquivo manifesto não deverá ser criptografado ou compactado, mesmo se as opções ENCRYPTED, GZIP, LZOP, BZIP2 ou ZSTD forem especificadas. COPY retornará um erro se o arquivo de manifesto especificado não for encontrado ou se ele não estiver formado corretamente.

Se um arquivo manifesto for usado, o parâmetro MANIFEST deverá ser especificado com o comando COPY. Se o parâmetro MANIFEST não for especificado, COPY vai pressupor que o arquivo especificado com FROM seja um arquivo de dados.

Para obter mais informações, consulte Carregar dados do Amazon S3.

authorization

O comando COPY precisa de autorização para acessar dados em outro recurso da AWS, inclusive em Amazon S3, Amazon EMR, Amazon DynamoDB e Amazon EC2. É possível fornecer essa autorização por meio de referência à função do AWS Identity and Access Management (IAM) que é associada ao cluster (controle de acesso com base da função). Para dados do Amazon S3, as identidades federadas pelo IAM podem usar credenciais de sessão especificando `IAM_ROLE 'SESSION'`. Para obter mais informações, consulte Parâmetros de autorização.

MANIFEST

Especifica que um manifesto é usado para identificar os arquivos de dados a serem carregados do Amazon S3. Se o parâmetro MANIFEST for usado, COPY carregará dados dos arquivos listados no manifesto referenciado por 's3://copy_from_s3_manifest_file'. Se o arquivo de

manifesto não for encontrado ou não estiver formado corretamente, ocorrerá uma falha no COPY. Para obter mais informações, consulte Uso de um manifesto para especificar arquivos de dados.

ENCRYPTED

Uma cláusula que especifica que os arquivos de entrada no Amazon S3 são criptografados com criptografia do lado do cliente com chaves gerenciadas pelo cliente. Para obter mais informações, consulte Carregar arquivos de dados criptografados do Amazon S3. Não especifique ENCRYPTED se arquivos de entrada forem criptografados usando-se criptografia no lado do servidor do Amazon S3 (SSE-KMS ou SSE-S3). COPY lê arquivos criptografados no lado do servidor automaticamente.

Se você especificar o parâmetro ENCRYPTED, também deverá especificar o parâmetro MASTER_SYMMETRIC_KEY.

Se os arquivos criptografados estiverem em formato compactado, adicione o parâmetro GZIP, LZOP, BZIP2 ou ZSTD.

Arquivos manifesto e JSONPaths não devem ser criptografados, mesmo se a opção ENCRYPTED for especificada.

MASTER_SYMMETRIC_KEY 'root_key'

A chave simétrica raiz que foi usada para criptografar arquivos de dados no Amazon S3. Se MASTER_SYMMETRIC_KEY for especificado, o parâmetro ENCRYPTED também deverá ser especificado. Para obter mais informações, consulte Carregar arquivos de dados criptografados do Amazon S3.

Se os arquivos criptografados estiverem em formato compactado, adicione o parâmetro GZIP, LZOP, BZIP2 ou ZSTD.

REGION [AS] 'aws-region'

Especifica a região da AWS onde os dados de origem estão localizados. REGION é necessário para COPY em um bucket do Amazon S3 ou uma tabela do DynamoDB quando o recurso da AWS que contém os dados não está na mesma região que o cluster do Amazon Redshift.

O valor de aws_region deve combinar com uma região listada na tabela regiões e endpoints do Amazon Redshift.

Se o parâmetro REGION for especificado, todos os recursos, inclusive um arquivo manifesto ou vários buckets do Amazon S3, devem estar localizados na região especificada.

> **Note**
>
> A transferência de dados entre regiões incorre em cobranças adicionais em relação ao bucket do Amazon S3 ou à tabela do DynamoDB que contém dados. Para ter mais informações sobre preço, consulte as transferências de dados de saída do Amazon S3 para outra região da AWS na página Preços do Amazon S3 e as transferências de dados de saída na página Preço do Amazon DynamoDB.

Por padrão, COPY pressupõe que os dados estejam localizados na mesma região do cluster do Amazon Redshift.

Parâmetros opcionais

Você também pode especificar os seguintes parâmetros com COPY do Amazon S3:

- Opções de mapeamento da coluna
- Parâmetros de formato de dados
- Parâmetros da conversão de dados
- Operações de carregamento de dados

Parâmetros incompatíveis

Você não pode usar os seguintes parâmetros com COPY do Amazon S3:

- SSH
- READRATIO

COPY do Amazon EMR

Você pode usar o comando COPY para carregar dados em paralelo de um cluster do Amazon EMR configurado para gravar arquivos de texto no Hadoop Distributed File System (HDFS) do cluster na forma de arquivos de largura fixa, delimitados por caractere, CSV, formatados em JSON ou Avro.

Tópicos

- Sintaxe
- Exemplo

- Parâmetros
- Parâmetros compatíveis
- Parâmetros incompatíveis

Sintaxe

```
FROM 'emr://emr_cluster_id/hdfs_filepath'
authorization
[ optional_parameters ]
```

Exemplo

O exemplo a seguir carrega dados como um cluster do Amazon EMR.

```
copy sales
from 'emr://j-SAMPLE2B500FC/myoutput/part-*'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Parâmetros

FROM

A origem dos dados a serem carregados.

'emr://emr_cluster_id/hdfs_file_path'

O identificador exclusivo do cluster do Amazon EMR e o caminho de arquivo HDFS que referencia os arquivos de dados para o comando COPY. Os nomes de arquivos de dados HDFS não devem conter o asterisco de caracteres curinga (*) e o ponto de interrogação (?).

> **Note**
>
> O cluster do Amazon EMR deve continuar em execução enquanto a operação COPY é concluída. Se alguns dos arquivos de dados HDFS forem alterados ou excluídos antes da operação COPY ser concluída, você poderá ter resultados inesperados, ou a operação COPY poderá falhar.

Você pode usar os caracteres curinga asterisco (*) e ponto de interrogação (?) como parte do argumento hdfs_file_path para especificar o carregamento de vários arquivos. Por exemplo,

`'emr://j-SAMPLE2B500FC/myoutput/part*'` identifica os arquivos `part-0000`, `part-0001` e assim por diante. Se não contiver caracteres curinga, o caminho do arquivo será tratado como uma string literal. Se você especificar somente um nome de pasta, COPY tentará carregar todos os arquivos na pasta.

> **Important**
>
> Se você usar caracteres curinga ou somente o nome da pasta, verifique se nenhum arquivo indesejado será cobrado. Por exemplo, alguns processos podem gravar um arquivo de log na pasta de saída.

Para obter mais informações, consulte Carregar dados do Amazon EMR.

authorization

O comando COPY precisa de autorização para acessar dados em outro recurso da AWS, inclusive em Amazon S3, Amazon EMR, Amazon DynamoDB e Amazon EC2. Essa autorização é fornecida por meio de referência ao perfil do AWS Identity and Access Management (IAM) que é associado ao cluster (controle de acesso com base no perfil). Para obter mais informações, consulte Parâmetros de autorização.

Parâmetros compatíveis

Você também pode especificar os seguintes parâmetros com COPY do Amazon EMR:

- Opções de mapeamento da coluna
- Parâmetros de formato de dados
- Parâmetros da conversão de dados
- Operações de carregamento de dados

Parâmetros incompatíveis

Você não pode usar os seguintes parâmetros com COPY do Amazon EMR:

- ENCRYPTED
- MANIFEST
- REGION

- READRATIO
- SSH

COPY de host remoto (SSH)

Você pode usar o comando COPY para carregar dados em paralelo de um ou mais hosts remotos, como instâncias do Amazon Elastic Compute Cloud (Amazon EC2) ou outros computadores. COPY se conecta aos hosts remotos usando o Secure Shell (SSH) e executa comandos nos hosts remotos para gerar a saída de texto. O host remoto pode ser uma instância do EC2 Linux ou outro computador Unix ou Linux configurado para aceitar conexões SSH. O Amazon Redshift pode conectar-se a vários hosts e abrir várias conexões SSH para cada host. O Amazon Redshift envia um comando exclusivo por meio de cada conexão para gerar saída de texto para a saída padrão do host, que o Amazon Redshift lê enquanto faz com um arquivo de texto.

Use a cláusula FROM para especificar a chave do objeto do Amazon S3 para o arquivo manifesto que fornece as informações que o COPY usa para abrir conexões SSH e executar os comandos remotos.

Tópicos

- Sintaxe
- Exemplos
- Parâmetros
- Parâmetros opcionais
- Parâmetros incompatíveis

> **Important**
>
> Se o bucket do S3 que mantém o arquivo manifesto não residir na mesma região da AWS que o cluster, use o parâmetro REGION para especificar a região na qual o bucket está localizado.

Sintaxe

```
FROM 's3://'ssh_manifest_file' }
```

```
authorization
SSH
| optional-parameters
```

Exemplos

O exemplo a seguir usa um arquivo manifesto para carregar dados de um host remoto usando o SSH.

```
copy sales
from 's3://amzn-s3-demo-bucket/ssh_manifest'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
ssh;
```

Parâmetros

FROM

A origem dos dados a serem carregados.

's3://copy_from_ssh_manifest_file'

O comando COPY pode conectar-se a vários hosts usando SSH e criar várias conexões SSH para cada host. O COPY executa um comando em cada conexão do host e carrega a saída dos comandos para a tabela em paralelo. O argumento s3://copy_from_ssh_manifest_file especifica a chave do objeto do Amazon S3 para o arquivo manifesto que fornece as informações que o COPY usa para abrir conexões SSH e executar os comandos remotos.

O argumento s3://copy_from_ssh_manifest_file deve referenciar explicitamente um único arquivo; ele não pode ser um prefixo das chaves. Por exemplo:

```
's3://amzn-s3-demo-bucket/ssh_manifest.txt'
```

O arquivo manifesto é um arquivo de texto no formato JSON que o Amazon Redshift usa para se conectar ao host. O arquivo manifesto especifica os endpoints do host SSH e os comandos que serão executados no host para retornar dados ao Amazon Redshift. Você também pode incluir a chave pública do host, o nome de usuário de login e um sinalizador obrigatório para cada entrada. O exemplo a seguir mostra um arquivo manifesto que cria duas conexões SSH:

```
{
```

```
    "entries": [
     {"endpoint":"<ssh_endpoint_or_IP>",
           "command": "<remote_command>",
           "mandatory":true,
           "publickey": "<public_key>",
           "username": "<host_user_name>"},
     {"endpoint":"<ssh_endpoint_or_IP>",
           "command": "<remote_command>",
           "mandatory":true,
           "publickey": "<public_key>",
           "username": "<host_user_name>"}
     ]
}
```

O arquivo manifesto contém um construto `"entries"` para cada conexão SSH. Você pode ter várias conexões para um único host ou várias conexões para vários hosts. As aspas duplas são necessárias, conforme mostrado, para os nomes de campo e valores. As aspas devem ser simples (0x22), e não inclinadas ou “inteligentes”. O único valor que não precisa de aspas duplas é o valor booliano `true` ou `false` para o campo `"mandatory"`.

A lista a seguir descreve os campos no arquivo manifesto.

endpoint

O endereço de URL ou o endereço IP do host. Por exemplo, `"ec2-111-222-333.compute-1.amazonaws.com"`, ou `"198.51.100.0"`.

command

O comando a ser executado pelo host para gerar a saída de texto ou a saída binária em formato gzip, lzop, bzip2 ou zstd. O comando pode ser qualquer um que o usuário "host_user_name" tenha permissão para executar. O comando pode ser tão simples quanto imprimir um arquivo, ou pode consultar um banco de dados ou iniciar um script. A saída (arquivo de texto, arquivo binário gzip, arquivo binário lzop ou arquivo binário bzip2) deve estar em uma forma que o comando COPY do Amazon Redshift possa ingerir. Para obter mais informações, consulte Preparação de dados de entrada.

publickey

(Opcional) A chave pública do host. Se fornecida, o Amazon Redshift usará a chave pública para identificar o host. Se a chave pública não for fornecida, o Amazon Redshift não tentará a identificação do host. Por exemplo, se a chave pública do host remoto for `ssh-rsa`

`AbcCbaxxx…Example root@amazon.com`, digite o seguinte texto no campo de chave pública: `"AbcCbaxxx…Example"`

mandatory

(Opcional) Uma cláusula que indica se o comando COPY deverá falhar se a tentativa de conexão falhar. O padrão é `false`. Se o Amazon Redshift não estabelecer pelo menos uma conexão, o comando COPY falhará.

username

(Opcional) O nome de usuário que será usado para fazer logon no sistema do host e executar o comando remoto. O nome de login do usuário deve ser o mesmo do login que foi usado para adicionar a chave pública do cluster do Amazon Redshift ao arquivo de chaves autorizadas do host. O nome de usuário padrão é `redshift`.

Para obter mais informações sobre como criar um arquivo manifesto, consulte Processo de carregamento de dados.

Para executar COPY de um host remoto, o parâmetro SSH deve estar especificado com o comando COPY. Se o parâmetro SSH não for especificado, COPY vai pressupor que o arquivo especificado com FROM seja um arquivo de dados e ocorrerá uma falha.

Se você usar a compactação automática, o comando COPY realizará duas operações de leitura de dados, o que significa que executará o comando remoto duas vezes. A primeira operação de leitura deve fornecer um exemplo de dados para análise da compactação, e a segunda efetivamente carrega os dados. Se executar o comando remoto duas vezes puder causar um problema, você deverá desabilitar a compactação automática. Para desabilitar a compactação automática, execute o comando COPY com o parâmetro COMPUPDATE definido como OFF. Para obter mais informações, consulte Carregamento de tabelas com compactação automática.

Para obter procedimentos detalhados sobre como usar COPY em SSH, consulte Carregamento de dados de hosts remotos.

authorization

O comando COPY precisa de autorização para acessar dados em outro recurso da AWS, inclusive em Amazon S3, Amazon EMR, Amazon DynamoDB e Amazon EC2. É possível fornecer essa autorização por meio de referência à função do AWS Identity and Access Management (IAM) que é associada ao cluster (controle de acesso com base da função). Para obter mais informações, consulte Parâmetros de autorização.

SSH

Uma cláusula que especifica que os dados devem ser carregados de um host remoto usando o protocolo SSH. Se especificar SSH, você também deverá fornecer um arquivo manifesto usando o argumento s3://copy_from_ssh_manifest_file.

> **Note**
>
> Se você estiver usando o SSH para copiar de um host usando um endereço IP privado em uma VPC remota, a VPC deverá ter o roteamento VPC aprimorado habilitado. Para obter mais informações sobre o Enhanced VPC Routing, consulte Enhanced VPC Routing do Amazon Redshift.

Parâmetros opcionais

Você também pode especificar os seguintes parâmetros com COPY do SSH:

- Opções de mapeamento da coluna
- Parâmetros de formato de dados
- Parâmetros da conversão de dados
- Operações de carregamento de dados

Parâmetros incompatíveis

Você não pode usar os seguintes parâmetros com COPY do SSH:

- ENCRYPTED
- MANIFEST
- READRATIO

COPY do Amazon DynamoDB

Para carregar dados de uma tabela do DynamoDB existente, use a cláusula FROM para especificar o nome da tabela do DynamoDB.

Tópicos

- Sintaxe

- Exemplos
- Parâmetros opcionais
- Parâmetros incompatíveis

> **Important**
>
> Se a tabela do DynamoDB não estiver localizada na mesma Região do seu cluster do Amazon Redshift, você deverá usar o parâmetro REGION para especificar a Região na qual os dados estão localizados.

Sintaxe

```
FROM 'dynamodb://table-name'
authorization
READRATIO ratio
| REGION [AS] 'aws_region'
| optional-parameters
```

Exemplos

O exemplo a seguir carrega dados de uma tabela do DynamoDB.

```
copy favoritemovies from 'dynamodb://ProductCatalog'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
readratio 50;
```

Parâmetros

FROM

A origem dos dados a serem carregados.

'dynamodb://table-name'

O nome da tabela do DynamoDB que contém os dados; por exemplo, `'dynamodb://ProductCatalog'`. Para obter detalhes sobre como os atributos do DynamoDB são mapeados para colunas do Amazon Redshift, consulte Carregar dados de uma tabela do Amazon DynamoDB.

O nome de uma tabela do DynamoDB é exclusivo de uma conta da AWS, identificada por credenciais de acesso da AWS.

authorization

O comando COPY precisa de autorização para acessar dados em outro recurso da AWS, inclusive em Amazon S3, Amazon EMR, Amazon DynamoDB e Amazon EC2. Essa autorização é fornecida por meio de referência ao perfil do AWS Identity and Access Management (IAM) que é associado ao cluster (controle de acesso com base no perfil). Para obter mais informações, consulte Parâmetros de autorização.

READRATIO [AS] ratio

A porcentagem do throughput provisionado da tabela do DynamoDB a ser usada no carregamento de dados. READRATIO é obrigatório para COPY do DynamoDB. Ele não pode ser usado com COPY do Amazon S3. É altamente recomendável definir a proporção como um valor menor que o throughput provisionado não usado médio. Os valores válidos são números inteiros de 1 a 200.

> **Important**
>
> A definição de READRATIO como 100 ou mais permite que o Amazon Redshift consuma todo o throughput provisionado da tabela do DynamoDB, o que afeta gravemente a performance de operações de leitura simultâneas na mesma tabela durante a sessão de COPY. O tráfego de gravação não é afetado. Os valores maiores que 100 têm permissão para solucionar problemas em cenários raros quando o Amazon Redshift deixar de atender ao throughput provisionado da tabela. Se você carregar dados do DynamoDB no Amazon Redshift continuamente, considere organizar as tabelas do DynamoDB como uma série temporal para separar o tráfego ao vivo da operação COPY.

Parâmetros opcionais

Você também pode especificar os seguintes parâmetros com COPY do Amazon DynamoDB:

- Opções de mapeamento da coluna
- Os seguintes parâmetros de conversão de dados são compatíveis:
  - ACCEPTANYDATE
  - BLANKSASNULL

- DATEFORMAT
- EMPTYASNULL
- ROUNDEC
- TIMEFORMAT
- TRIMBLANKS
- TRUNCATECOLUMNS

- Operações de carregamento de dados

Parâmetros incompatíveis

Você não pode usar os seguintes parâmetros com COPY do DynamoDB:

- Todos os parâmetros de formato de dados
- ESCAPE
- FILLRECORD
- IGNOREBLANKLINES
- IGNOREHEADER
- NULL
- REMOVEQUOTES
- ACCEPTINVCHARS
- MANIFEST
- ENCRYPTED

Parâmetros de autorização

O comando COPY precisa de autorização para acessar dados em outro recurso da AWS, inclusive em Amazon S3, Amazon EMR, Amazon DynamoDB e Amazon EC2. É possível fornecer essa autorização por meio de referência à função do AWS Identity and Access Management (IAM) que é associada ao cluster (controle de acesso com base da função). Você pode criptografar seus dados de carregamento no Amazon S3.

Os seguintes tópicos dão mais detalhes e exemplos de opções de autenticação:

- Permissões do IAM para COPY, UNLOAD e CREATE LIBRARY

- Controle de acesso com base em função

Use o seguinte para dar autorização para o comando COPY:

- Parâmetro the section called “Usar IAM_ROLE”

Usar o parâmetro IAM_ROLE

IAM_ROLE

Use a palavra-chave padrão para que o Amazon Redshift use a função do IAM definida como padrão e associada ao cluster quando o comando COPY for executado.

Use o nome do recurso da Amazon (ARN) de uma função do IAM que seu cluster usa para autenticação e autorização.

A seguir, a sintaxe do parâmetro IAM_ROLE.

```
IAM_ROLE { default | 'SESSION' | 'arn:aws:iam::<Conta da AWS-id>:role/<role-name>' }
```

Para obter mais informações, consulte Controle de acesso com base em função.

SESSION

Use a palavra-chave SESSION para carregar dados usando as credenciais da sua sessão atual federada do IAM. Essa opção está disponível somente quando você se conecta ao Amazon Redshift com uma identidade federada do IAM.

Com SESSION, o Amazon Redshift usa as mesmas permissões do Amazon S3 que sua identidade federada já tem. O Amazon Redshift não assume um perfil do IAM de cluster, então você não precisa anexar um ao cluster ou grupo de trabalho para esse acesso.

Ao especificar SESSION, você não pode combiná-lo com nenhum outro método de autorização.

Para carregar com SESSION, você deve ter as permissões listadas emPermissões do IAM para COPY, UNLOAD e CREATE LIBRARY.

> **Note**
>
> SESSION não é compatível com trabalhos de cópia automática. Um trabalho de cópia automática é executado de forma assíncrona e não pode acessar suas credenciais de

sessão federada, portanto, requer um perfil do IAM. Para obter mais informações, consulte COPY JOB.

Para ver um exemplo do uso de uma identidade federada, consulte Usar uma identidade federada para gerenciar o acesso do Amazon Redshift aos recursos locais e a tabelas externas do Amazon Redshift Spectrum.

Opções de mapeamento da coluna

Por padrão, COPY insere valores nas colunas da tabela de destino na mesma ordem dos campos ocorridos nos arquivos de dados. Se a ordem de coluna padrão não funcionar, você poderá especificar uma lista de colunas ou usar expressões JSONPath para mapear campos de dados de origem para as colunas de destino.

- Column List
- JSONPaths File

Lista de colunas

Você pode especificar uma lista separada por vírgulas de nomes de coluna para carregar campos de dados de origem em colunas de destino específicas. As colunas podem estar em qualquer ordem na instrução COPY, mas durante o carregamento de arquivos simples, como em um bucket do Amazon S3, a ordem deve corresponder à ordem dos dados de origem.

Durante o carregamento de uma tabela do Amazon DynamoDB, a ordem não importa. O comando COPY compara nomes de atributo nos itens recuperados da tabela do DynamoDB com nomes de coluna na tabela do Amazon Redshift. Para obter mais informações, consulte . Carregar dados de uma tabela do Amazon DynamoDB

A seguir, o formato para uma lista de colunas.

```
COPY tablename (column1 [,column2, ...])
```

Se uma coluna na tabela de destino for omitida da lista de colunas, COPY carregará a expressão DEFAULT da coluna.

Se a coluna de destino não tiver um padrão, COPY tentará carregar NULL.

Se COPY tentar atribuir NULL a uma coluna definida como NOT NULL, o comando COPY falhará.

Se uma coluna IDENTITY estiver incluída na lista de colunas, EXPLICIT_IDS também deverá ser especificado; se uma coluna IDENTITY for omitida, EXPLICIT_IDS não poderá ser especificado. Se nenhuma lista de colunas for especificada, o comando se comportará como se uma lista de colunas completa, na ordem, tivesse sido especificada, com colunas IDENTITY omitidas se EXPLICIT_IDS também não tiver sido especificado.

Se uma coluna estiver definida com GENERATED BY DEFAULT AS IDENTITY, será possível copiá-la. Os valores são gerados ou atualizados com os valores fornecidos. A opção EXPLICIT_IDS não é necessária. COPY não atualiza a marca d'água alta de identidade. Para obter mais informações, consulte GENERATED BY DEFAULT AS IDENTITY.

Arquivo JSONPaths

Ao carregar de arquivos de dados em formato JSON ou Avro, COPY mapeará automaticamente os elemento de dados nos dados JSON ou Avro para as colunas na tabela de destino. Ele faz isso comparando nomes de campo no esquema do Avro com nomes de coluna na tabela de destino ou na lista de colunas.

Em alguns casos, os nomes de coluna e os nomes de campo não correspondem, ou você precisa mapear para níveis mais profundos na hierarquia de dados. Para mapear explicitamente elemento de dados JSON ou Avro para colunas, você pode usar um arquivo JSONPaths.

Para obter mais informações, consulte Arquivo JSONPaths.

Parâmetros de formato de dados

Por padrão, o comando COPY espera que os dados de origem sejam texto UTF-8 delimitados por caractere. O delimitador padrão é um caractere de barra (|). Se os dados de origem estiverem em outro formato, use os parâmetros a seguir para especificar o formato de dados:

- FORMAT
- CSV
- DELIMITER
- FIXEDWIDTH
- SHAPEFILE
- AVRO

- JSON format for COPY
- PARQUET
- ORC

Além dos formatos de dados padrão, COPY é compatível com os seguintes formatos de dados colunares para COPY do Amazon S3:

- ORC
- PARQUET

COPY de formato colunar é compatível com determinada restrição. Para obter mais informações, consulte COPY de formatos de dados colunar.

Parâmetros de formato de dados

FORMAT [AS]

(Opcional) Identifica palavras-chave do formato de dados. Os argumentos FORMAT estão descritos a seguir.

CSV [QUOTE [AS] 'quote_character']

Permite o uso do formato CSV nos dados de entrada. Para delimitadores de escape automático, caracteres em nova linha e retornos de carro, deixe o campo no caractere especificado no parâmetro QUOTE. As aspas padrão são aspas duplas ("). Quando as aspas são usadas dentro de um campo, faça o escape do caractere com aspas adicionais. Por exemplo, se as aspas forem duplas, para inserir a string `A "quoted" word` o arquivo de entrada deve incluir a string `"A ""quoted"" word"`. Quando o parâmetro CSV for usado, o delimitador padrão será uma vírgula (,). Você pode especificar um delimitador diferente usando o parâmetro DELIMITER.

Quando um campo está entre aspas, o espaço em branco entre os delimitadores e as aspas é ignorado. Se o delimitador for um caractere de espaço em branco, como uma tabulação, o delimitador não será tratado como um espaço em branco.

CSV não pode ser usado com FIXEDWIDTH, REMOVEQUOTES ou ESCAPE.

QUOTE [AS] 'quote_character'

Opcional. Especifica o caractere a ser usado como as aspas durante o uso do parâmetro CSV. O padrão são aspas duplas ("). Se usar o parâmetro QUOTE para definir aspas

diferentes das aspas duplas, você não precisará escapar aspas duplas dentro do campo. O parâmetro QUOTE só pode ser usado com o parâmetro CSV. A palavra-chave AS é opcional.

DELIMITER [AS] ['delimiter_char']

Especifica os caracteres que são usados para separar campos no arquivo de entrada, como barra (`|`), vírgula (`,`) ou tabulação (`\t`), ou vários caracteres, como `|~|`. Caracteres não imprimíveis são aceitos. Os caracteres também podem ser representados em octal como suas unidades de código UTF-8. Para octal, use o formato "\ddd", em que "d" é um dígito octal (0 a 7). O delimitador padrão é um caractere de barra (`|`), a menos que o parâmetro CSV seja usado, caso em que o delimitador padrão é uma vírgula (`,`). A palavra-chave AS é opcional. DELIMITER não pode ser usado com FIXEDWIDTH.

FIXEDWIDTH 'fixedwidth_spec'

Carrega os dados de um arquivo em que a largura de cada coluna tem um tamanho fixo, em vez de colunas separadas por um delimitador. A fixedwidth_spec é uma string que especifica um rótulo de coluna definido pelo usuário e uma largura de coluna. O rótulo da coluna pode ser uma string de texto ou um inteiro, dependendo do que o usuário escolher. O rótulo da coluna não tem relação com o nome da coluna. A ordem dos pares de rótulo/largura deve corresponder exatamente à ordem das colunas da tabela. FIXEDWIDTH não pode ser usado com CSV ou DELIMITER. No Amazon Redshift, como o tamanho das colunas CHAR e VARCHAR é expressado em bytes, verifique se a largura de coluna especificada por você acomoda o tamanho binário de caracteres multibyte ao preparar o arquivo a ser carregado. Para obter mais informações, consulte Tipos de caracteres.

O formato de fixedwidth_spec é mostrado a seguir:

```
'colLabel1:colWidth1,colLabel:colWidth2, ...'
```

SHAPEFILE [SIMPLIFY [AUTO] ['tolerance']]

Permite o uso do formato SHAPEFILE nos dados de entrada. Por padrão, a primeira coluna do shapefile é uma coluna `GEOMETRY` ou `IDENTITY`. Todas as colunas subsequentes seguem a ordem especificada no shapefile.

É possível usar SHAPEFILE com FIXEDWIDTH, REMOVEQUOTES ou ESCAPE.

Para usar objetos `GEOGRAPHY` com `COPY FROM SHAPEFILE`, primeiro ingira em um coluna `GEOMETRY` e transmita os objetos para objetos `GEOGRAPHY`. .

SIMPLIFY [tolerance]

(Opcional) Simplifica todas as geometrias durante o processo de ingestão usando o algoritmo Ramer-Douglas-Peucker e a tolerância dada.

SIMPLIFY AUTO [tolerance]

(Opcional) Simplifica apenas geometrias maiores que o tamanho máximo da geometria. Essa simplificação usa o algoritmo Ramer-Douglas-Peucker e a tolerância calculada automaticamente se isso não exceder a tolerância especificada. Este algoritmo calcula o tamanho para armazenar objetos dentro da tolerância especificada. O valor tolerance é opcional.

Para obter exemplos de carregamento de shapefiles, consulte Carregar um shapefile no Amazon Redshift.

AVRO [AS] 'avro_option'

Especifica se os dados de origem estão em formato Avro.

O formato Avro é compatível com COPY nestes serviços e protocolos:

- Amazon S3
- Amazon EMR
- Hosts remotos (SSH)

Avro não é compatível com COPY no DynamoDB.

Avro é um protocolo de serialização de dados. Um arquivo de origem do Avro inclui um esquema que define a estrutura dos dados. O tipo de esquema do Avro deve ser `record`. COPY aceita a criação de arquivos do Avro usando o codec não compactado padrão, bem como os codecs de compactação `deflate` e `snappy`. Para obter mais informações sobre o Avro, vá até Apache Avro.

Os valores válidos para avro_option são os seguintes:

- `'auto'`
- `'auto ignorecase'`
- `'s3://jsonpaths_file'`

O padrão é `'auto'`.

COPY mapeará automaticamente os elemento de dados nos dados JSON ou Avro para as colunas na tabela de destino. Ele faz isso comparando nomes de campo no esquema do Avro com nomes de coluna na tabela de destino. A correspondência diferencia maiúsculas de minúsculas para `'auto'` e não diferencia maiúsculas de minúsculas para `'auto ignorecase'`.

Como os nomes de coluna em tabelas do Amazon Redshift estão sempre em minúsculas, quando você usa a opção `'auto'`, os nomes de campo do JSON correspondentes também devem estar em minúsculas. Se os nomes dos campos não estiverem todos em minúsculas, você poderá usar a opção `'auto ignorecase'`. Com o arguento padrão `'auto'`, COPY reconhece apenas o primeiro nível de campos, ou campos externos, na estrutura.

Para mapear explicitamente nomes de coluna para nomes de campo do Avro, você pode usar um Arquivo JSONPaths.

Por padrão, COPY tenta comparar todas as colunas na tabela de destino com os nomes de campo do Avro. Para carregar um subconjunto das colunas, você também pode especificar uma lista de colunas. Se uma coluna na tabela de destino for omitida da lista de colunas, COPY carregará a expressão DEFAULT da coluna. Se a coluna de destino não tiver um padrão, COPY tentará carregar NULL. Se uma coluna estiver incluída na lista de colunas e COPY não encontrar um campo correspondente nos dados do Avro, COPY tentará carregar NULL na coluna.

Se COPY tentar atribuir NULL a uma coluna definida como NOT NULL, o comando COPY falhará.

Esquema do Avro

Um arquivo de dados de origem do Avro inclui um esquema que define a estrutura dos dados. COPY lê o esquema que faz parte do arquivo de dados de origem do Avro a fim de mapear elementos de dados para colunas da tabela de destino. O exemplo a seguir mostra um esquema do Avro.

```
{
    "name": "person",
    "type": "record",
    "fields": [
        {"name": "id", "type": "int"},
        {"name": "guid", "type": "string"},
        {"name": "name", "type": "string"},
        {"name": "address", "type": "string"}]
```

```
}
```

O esquema do Avro é definido usando-se o formato JSON. O objeto JSON de nível mais alto contém três pares de nome/valor com os nomes, ou chaves, `"name"`, `"type"` e `"fields"`.

Os pares de chaves `"fields"` com uma matriz de objetos que definem o nome e o tipo de dados de cada campo na estrutura de dados. Por padrão, COPY compara automaticamente os nomes de campo com os nomes de coluna. Como nomes de coluna estão sempre em minúsculas, nomes de campo também devem estar em minúsculas, a não ser a opção `‘auto ignorecase’` seja especificada. Todos os nomes de campo que não corresponderem a um nome de coluna serão ignorados. A ordem das colunas não importa. No exemplo anterior, COPY é mapeado para os nomes de coluna `id`, `guid`, `name` e `address`.

Com o argumento `'auto'` padrão, COPY compara apenas os objetos no primeiro nível com as colunas. A fim de mapear para níveis mais profundos no esquema, ou se nomes de campo e de coluna não corresponderem, use um arquivo JSONPaths para definir o mapeamento. Para obter mais informações, consulte Arquivo JSONPaths.

Se o valor associado a uma chave for um tipo de dados do Avro complexo como byte, matriz, registro, mapa ou link, COPY carregará o valor como uma string, em que a string é a representação JSON dos dados. Aqui, a string é a representação JSON dos dados. COPY carrega tipos de dados enum do Avro como strings, em que o conteúdo é o nome do tipo. Para ver um exemplo, consulte COPY no formato JSON.

O tamanho máximo do cabeçalho de arquivo do Avro, o que inclui o esquema e os metadados do arquivo, é 1 MB.

O tamanho máximo de um único bloco de dados do Avro é 4 MB. Isso é diferente do tamanho máximo da linha. Se o tamanho máximo de um único bloco de dados do Avro for excedido, mesmo se o tamanho de linha resultante for menor que o limite do tamanho de linha de 4 MB, o comando COPY falhará.

Ao calcular o tamanho da linha, o Amazon Redshift contabiliza internamente os caracteres de barra vertical (|) duas vezes. Se os dados de entrada contiverem um número muito grande de caracteres de barra, será possível para o tamanho da linha exceder 4 MB, mesmo se o bloco de dados for menor que 4 MB.

JSON [AS] 'json_option'

Os dados de origem estão em formato JSON.

O formato JSON é compatível com COPY nestes serviços e protocolos:

- Amazon S3
- COPY do Amazon EMR
- COPY de SSH

JSON não é compatível com COPY no DynamoDB.

Os valores válidos para json_option são os seguintes:

- `'auto'`
- `'auto ignorecase'`
- `'s3://jsonpaths_file'`
- `'noshred'`

O padrão é `'auto'`. O Amazon Redshift não fragmenta os atributos de estruturas JSON em várias colunas ao carregar um documento JSON.

Por padrão, COPY tenta comparar todas as colunas na tabela de destino com as chaves de nome de campo do JSON. Para carregar um subconjunto das colunas, você também pode especificar uma lista de colunas. Se as chaves de nome de campo JSON não estiverem todas em minúsculas, você poderá usar a opção `'auto ignorecase'` ou um Arquivo JSONPaths a fim de mapear explicitamente nomes de coluna para chaves de nome de campo JSON.

Se uma coluna na tabela de destino for omitida da lista de colunas, COPY carregará a expressão DEFAULT da coluna. Se a coluna de destino não tiver um padrão, COPY tentará carregar NULL. Se uma coluna estiver incluída na lista de colunas e COPY não encontrar um campo correspondente nos dados JSON, COPY tentará carregar NULL na coluna.

Se COPY tentar atribuir NULL a uma coluna definida como NOT NULL, o comando COPY falhará.

COPY mapeará automaticamente os elemento de dados nos dados JSON ou Avro para as colunas na tabela de destino. Ele faz isso combinando chaves de objeto, ou nomes, nos pares de nome-valor de origem com os nomes de colunas na tabela de destino.

Consulte os seguintes detalhes sobre cada valor de json_option:

'auto'

Com essa opção, a correspondência diferencia maiúsculas de minúsculas. Como os nomes de coluna em tabelas do Amazon Redshift estão sempre em minúsculas, quando você usa

a opção `'auto'`, os nomes de campo do JSON correspondentes também devem estar em minúsculas.

'auto ignorecase'

Com essa opção, a correspondência não diferencia maiúsculas de minúsculas. Como os nomes de coluna nas tabelas do Amazon Redshift estão sempre em minúsculas, quando você usa a opção `'auto ignorecase'`, os nomes de campo JSON correspondentes podem ser minúsculas, maiúsculas ou mistas.

's3://jsonpaths_file'

COPY usa o arquivo chamado JSONPaths a fim de mapear os elementos de dados nos dados de origem do JSON para as colunas na tabela de destino. O *`s3://jsonpaths_file`* deve ser uma chave de objeto do Amazon S3 que referencia explicitamente um único arquivo. Um exemplo é `'s3://amzn-s3-demo-bucket/jsonpaths.txt'`. O argumento não pode ser um prefixo das chaves. Para obter mais informações sobre como usar um arquivo JSONPaths, consulte the section called “Arquivo JSONPaths”.

Em alguns casos, o arquivo especificado pelo `jsonpaths_file` tem o mesmo prefixo que o caminho especificado por `copy_from_s3_objectpath` para os arquivos de dados. Em caso afirmativo, COPY lê o arquivo JSONPaths como um arquivo de dados e retorna erros. Por exemplo, digamos que seus arquivos de dados usem o caminho do objeto `s3://amzn-s3-demo-bucket/my_data.json` e seu arquivo JSONPaths é `s3://amzn-s3-demo-bucket/my_data.jsonpaths`. Nesse caso, COPY tenta carregar `my_data.jsonpaths` como um arquivo de dados.

‘noshred'

Com esta opção, o Amazon Redshift não fragmenta os atributos de estruturas JSON em várias colunas ao carregar um documento JSON.

Arquivo de dados JSON

O arquivo de dados JSON contém um conjunto de objetos ou matrizes. COPY carrega cada objeto ou matriz JSON em uma linha na tabela de destino. Cada objeto ou matriz correspondente a uma linha deve ser uma estrutura no nível raiz, autônoma; ou seja, não deve ser um membro de outra estrutura JSON.

Um objeto JSON começa e termina com chaves ({ }) e contém uma coleção de pares de nome-valor desordenada. Cada nome e valor em par são separados por um dois-pontos, e os pares são

separados por vírgulas. Por padrão, a chave de objeto, ou o nome, nos pares de nome-valor deve corresponder ao nome da coluna correspondente na tabela. Os nomes de coluna nas tabelas do Amazon Redshift estão sempre em minúsculas, as chaves de nome de campo correspondentes do JSON também deverão estar em minúsculas. Se os nomes de coluna e as chaves do JSON não forem correspondentes, use the section called “Arquivo JSONPaths” a fim de mapear explicitamente as colunas para as chaves.

A ordem em um objeto JSON não importa. Todos os nomes que não corresponderem a um nome de coluna serão ignorados. A seguir, a estrutura de um objeto JSON simples.

```
{
  "column1": "value1",
  "column2": value2,
  "notacolumn" : "ignore this value"
}
```

Uma matriz JSON começa e termina com colchetes ([ ]), e contém uma coleção ordenada de valores separados por vírgulas. Se os arquivos de dados usarem matrizes, você deverá especificar um arquivo JSONPaths para corresponder os valores às colunas. A seguir, a estrutura de uma matriz JSON simples.

```
["value1", value2]
```

O JSON bastante deve ser bem formado. Por exemplo, os objetos ou as matrizes não podem ser separados por vírgulas ou por quaisquer outros caracteres, exceto o espaço em branco. As strings devem estar entre aspas duplas. As aspas devem ser simples (0x22), e não inclinadas ou "inteligentes".

O tamanho máximo de um objeto ou matriz JSON, inclusive colchetes ou chaves, é 4 MB. Isso é diferente do tamanho máximo da linha. Se o tamanho máximo de um único objeto ou matriz JSON for excedido, mesmo se o tamanho de linha resultante for menor que o limite do tamanho de linha de 4 MB, o comando COPY falhará.

Ao calcular o tamanho da linha, o Amazon Redshift contabiliza internamente os caracteres de barra vertical (|) duas vezes. Se os dados de entrada contiverem um número muito grande de caracteres de barra, será possível para o tamanho da linha exceder 4 MB, mesmo se o tamanho do objeto for menor que 4 MB.

COPY carrega `\n` como um caractere de nova linha e `\t` como um caractere de tabulação. Para carregar uma barra invertida, use uma barra invertida de escape (`\\`).

COPY pesquisa a origem JSON especificada em busca de um objeto JSON válido ou uma matriz bem formada. Se COPY encontrar algum caractere que não seja um espaço em branco antes de localizar uma estrutura JSON útil, ou entre objetos JSON ou matrizes válidos, COPY retornará um erro para cada instância. Esses erros são válidos para a contagem MAXERROR. Quando a contagem de erros for igual ou exceder MAXERROR, COPY falhará.

Para cada erro, o Amazon Redshift registra uma linha na tabela do sistema STL_LOAD_ERRORS. A coluna LINE_NUMBER registra a última linha do objeto JSON que causou o erro.

Se IGNOREHEADER for especificado, COPY ignorará o número especificado de linhas nos dados JSON. Os caracteres de nova linha nos dados JSON são sempre contados para cálculos de IGNOREHEADER.

COPY carrega strings vazias como campos vazios por padrão. Se EMPTYASNULL for especificado, COPY carregará strings vazias de campos CHAR e VARCHAR como NULL. As strings vazias de outros tipos de dados, como INT, são sempre carregadas com NULL.

As seguintes opções não são compatíveis com JSON:

- CSV
- DELIMITER
- ESCAPE
- FILLRECORD
- FIXEDWIDTH
- IGNOREBLANKLINES
- NULL AS
- READRATIO
- REMOVEQUOTES

Para obter mais informações, consulte COPY no formato JSON. Para obter mais informações sobre estruturas de dados JSON, vá até www.json.org.

Arquivo JSONPaths

Se você estiver carregando dados formatados em JSON ou de origem Avro, por padrão, COPY mapeará os elementos de dados no primeiro nível nos dados de origem para as colunas na tabela

de destino. Ele faz isso combinando chaves de objeto, ou nomes, nos pares de nome-valor com os nomes de colunas na tabela de destino.

Se os nomes de coluna e as chaves de objeto não forem correspondentes, ou a fim de mapear para níveis mais profundos na hierarquia de dados, você poderá usar um arquivo JSONPaths a fim de mapear explicitamente elemento de dados JSON ou Avro para colunas. O arquivo JSONPaths mapeia elementos de dados JSON para colunas comparando a ordem das colunas na tabela de destino ou na lista de colunas.

O arquivo JSONPaths deve conter somente um único objeto JSON (e não uma matriz). O objeto JSON é um par de nome-valor. A chave de objeto, que é o nome no par de nome-valor, deve ser `"jsonpaths"`. O valor no par de nome-valor é uma matriz de expressões JSONPath. Cada expressão JSONPath referencia um único elemento na hierarquia de dados JSON ou no esquema Avro, da mesma maneira como uma expressão XPath se refere a elementos em um documento XML. Para obter mais informações, consulte Expressões JSONPath.

Para usar um arquivo JSONPaths, adicione a palavra-chave JSON ou AVRO ao comando COPY. Especifique o nome do bucket do S3 e o caminho de objeto do arquivo JSONPaths usando o formato a seguir.

```
COPY tablename
FROM 'data_source'
IAM_ROLE 'arn:aws:iam::<aws-account-id>:role/<role-name>'
FORMAT AS { AVRO | JSON } 's3://jsonpaths_file';
```

O valor `s3://jsonpaths_file` deve ser uma chave de objeto do Amazon S3 que referencia explicitamente um único arquivo, como `'s3://amzn-s3-demo-bucket/jsonpaths.txt'`. Não pode ser um prefixo das chaves.

Em alguns casos, se você estiver carregando do Amazon S3, o arquivo especificado por `jsonpaths_file` tem o mesmo prefixo que o caminho especificado por `copy_from_s3_objectpath` para os arquivos de dados. Em caso afirmativo, COPY lê o arquivo JSONPaths como um arquivo de dados e retorna erros. Por exemplo, digamos que seus arquivos de dados usem o caminho do objeto `s3://amzn-s3-demo-bucket/my_data.json` e seu arquivo JSONPaths é `s3://amzn-s3-demo-bucket/my_data.jsonpaths`. Nesse caso, COPY tenta carregar `my_data.jsonpaths` como um arquivo de dados.

Se o nome de uma chave for qualquer string diferente de `"jsonpaths"`, o comando COPY não retornará um erro, mas ignorará jsonpaths_file e usará o argumento `'auto'` em seu lugar.

Se algum dos seguintes ocorrer, o comando COPY falhará:

- O JSON está malformado.
- Existe mais de um objeto JSON.
- Todos os caracteres, exceto o espaço branco, estão fora do objeto.
- Um elemento de matriz é uma string vazia ou não é uma string.

MAXERROR não se aplica ao arquivo JSONPaths.

O arquivo JSONPaths não deve ser criptografado, mesmo se a opção ENCRYPTED for especificada.

Para obter mais informações, consulte COPY no formato JSON.

Expressões JSONPath

O arquivo JSONPaths usa expressões JSONPath a fim de mapear campos de dados para colunas de destino. Cada expressão JSONPath corresponde a uma coluna na tabela de destino do Amazon Redshift. A ordem dos elementos de matriz JSONPath deverá corresponder à ordem das colunas na tabela de destino ou na lista de colunas, se uma lista de colunas for usada.

As aspas duplas são necessárias, conforme mostrado, para os nomes de campo e valores. As aspas devem ser simples (0x22), e não inclinadas ou “inteligentes”.

Se um elemento de objeto referenciado por uma expressão JSONPath não for encontrado nos dados JSON, COPY tentará carregar um valor NULL. Se o objeto referenciado for malformado, COPY retornará um erro de carregamento.

Se um elemento de matriz referenciado por uma expressão JSONPath não for encontrado nos dados JSON nem Avro, COPY falhará com o seguinte erro: `Invalid JSONPath format: Not an array or index out of range.` Remova todos os elementos de matriz de JSONPaths que não existirem nos dados de origem e verifique se as matrizes nos dados de origem estão bem formadas.

As expressões JSONPath podem usar a notação de colchetes ou de pontos, mas não pode misturar notações. O exemplo a seguir mostra expressões JSONPath usando notação de colchetes.

```
{
    "jsonpaths": [
        "$['venuename']",
```

```
        "$['venuecity']",
        "$['venuestate']",
        "$['venueseats']"
    ]
}
```

O exemplo a seguir mostra expressões JSONPath usando notação de pontos.

```
{
    "jsonpaths": [
        "$.venuename",
        "$.venuecity",
        "$.venuestate",
        "$.venueseats"
    ]
}
```

No contexto da sintaxe COPY do Amazon Redshift, uma expressão JSONPath deve especificar o caminho explícito para um único elemento de nome em uma estrutura de dados hierárquicos JSON ou Avro. O Amazon Redshift não dá suporte a elementos JSONPath, como caracteres curinga ou expressões de filtro, que podem ser resolvidos para um caminho ambíguo ou vários elementos de nome.

Para obter mais informações, consulte COPY no formato JSON.

Usar JSONPaths com dados Avro

O exemplo a seguir mostra um esquema do Avro com vários níveis.

```
{
    "name": "person",
    "type": "record",
    "fields": [
        {"name": "id", "type": "int"},
        {"name": "guid", "type": "string"},
        {"name": "isActive", "type": "boolean"},
        {"name": "age", "type": "int"},
        {"name": "name", "type": "string"},
        {"name": "address", "type": "string"},
        {"name": "latitude", "type": "double"},
        {"name": "longitude", "type": "double"},
        {
```

```
            "name": "tags",
            "type": {
                        "type" : "array",
                        "name" : "inner_tags",
                        "items" : "string"
                    }
        },
        {
            "name": "friends",
            "type": {
                        "type" : "array",
                        "name" : "inner_friends",
                        "items" : {
                                    "name" : "friends_record",
                                    "type" : "record",
                                    "fields" : [
                                                {"name" : "id", "type" : "int"},
                                                {"name" : "name", "type" : "string"}
                                               ]
                                   }
                    }
        },
        {"name": "randomArrayItem", "type": "string"}
    ]
}
```

O exemplo a seguir mostra um arquivo JSONPaths que usa expressões do AvroPath para referenciar o esquema anterior.

```
{
    "jsonpaths": [
        "$.id",
        "$.guid",
        "$.address",
        "$.friends[0].id"
    ]
}
```

O exemplo JSONPaths inclui os seguintes elementos:

jsonpaths

O nome do objeto JSON que contém as expressões AvroPath.

[…]

A matriz JSON está entre colchetes contendo os elementos do caminho.

$

O cifrão se refere ao elemento raiz no esquema do Avro, que é a matriz `"fields"`.

"$.id",

O destino da expressão AvroPath. Nesta instância, o destino é o elemento na matriz `"fields"` com o nome `"id"`. As expressões são separadas por vírgulas.

"$.friends[0].id"

Os colchetes indicam um índice de matriz. Como as expressões JSONPath usam indexação baseada em zero, essa expressão referencia o primeiro elemento na matriz `"friends"` com o nome `"id"`.

A sintaxe do esquema do Avro exige o uso de campos internos para definir a estrutura de tipos de dados de registro e matriz. Os campos internos são ignorados pelas expressões AvroPath. Por exemplo, o campo `"friends"` define uma matriz chamada `"inner_friends"`, que, por sua vez, define um registro chamado `"friends_record"`. A expressão AvroPath para referenciar o campo `"id"` pode ignorar os campos extras para referenciar diretamente o campo de destino. As expressões AvroPath a seguir referenciam os dois campos que pertencem à matriz `"friends"`.

```
"$.friends[0].id"
"$.friends[0].name"
```

Parâmetros de formato de dados colunar

Além dos formatos de dados padrão, COPY é compatível com os seguintes formatos de dados colunares para COPY do Amazon S3. COPY de formato colunar é compatível com determinadas restrições. Para obter mais informações, consulte COPY de formatos de dados colunar.

ORC

Carrega os dados de um arquivo que usa o formato de arquivo Colunar de linha otimizado (ORC).

PARQUET

Carrega os dados de um arquivo que usa o formato de arquivo Parquet.

Parâmetros de compactação de arquivo

Você pode carregar a partir de arquivos de dados compactados especificando os seguintes parâmetros.

Parâmetros de compactação de arquivo

BZIP2

Um valor que especifica que o arquivo ou os arquivos de entrada estão em formato bzip2 compactado (arquivos .bz2). A operação COPY lê cada arquivo compactado e descompacta os dados à medida que eles são carregados.

GZIP

Um valor que especifica que o arquivo ou os arquivos de entrada estão em formato gzip compactado (arquivos .gz). A operação COPY lê cada arquivo compactado e descompacta os dados à medida que eles são carregados.

LZOP

Um valor que especifica que o arquivo ou os arquivos de entrada estão em formato lzop compactado (arquivos .lzo). A operação COPY lê cada arquivo compactado e descompacta os dados à medida que eles são carregados.

> **Note**
>
> COPY não dá suporte a arquivos que foram compactados com a opção lzop --filter.

ZSTD

Um valor que especifica que o arquivo ou os arquivos de entrada estão em formato compactado Zstandard (arquivos .zst). A operação COPY lê cada arquivo compactado e descompacta os dados à medida que eles são carregados.

> **Note**
>
> ZSTD é compatível apenas com COPY do Amazon S3.

Parâmetros da conversão de dados

À medida que carrega a tabela, COPY tenta converter implicitamente as strings nos dados de origem no tipo de dados da coluna de destino. Se precisar especificar uma conversão diferente do comportamento padrão, ou se a conversão padrão resultar em erros, você poderá gerenciar conversões de dados especificando os parâmetros a seguir. Para obter mais informações sobre a sintaxe desses parâmetros, consulte Sintaxe de COPY.

- ACCEPTANYDATE
- ACCEPTINVCHARS
- BLANKSASNULL
- DATEFORMAT
- EMPTYASNULL
- ENCODING
- ESCAPE
- EXPLICIT_IDS
- FILLRECORD
- IGNOREBLANKLINES
- IGNOREHEADER
- NULL AS
- REMOVEQUOTES
- ROUNDEC
- TIMEFORMAT
- TRIMBLANKS
- TRUNCATECOLUMNS

Parâmetros da conversão de dados

ACCEPTANYDATE

Permite que qualquer formato de data, inclusive formatos inválidos, como `00/00/00 00:00:00`, seja carregado sem gerar um erro. Esse parâmetro se aplica somente a colunas TIMESTAMP e DATA. Use sempre ACCEPTANYDATE com o parâmetro DATEFORMAT. Se o formato de data dos dados não for correspondente à especificação DATEFORMAT, o Amazon Redshift inserirá um valor NULL nesse campo.

ACCEPTINVCHARS [AS] ['replacement_char']

Permite o carregamento de dados em colunas VARCHAR mesmo se os dados contiverem caracteres UTF-8 inválidos. Quando ACCEPTINVCHARS é especificado, COPY substitui todos os caracteres UTF-8 inválidos por uma string de mesmo tamanho que consiste no caractere especificado por replacement_char. Por exemplo, se o caractere substituto for '^', um caractere inválido de três bytes será substituído por '^^^'.

O caractere substituto pode ser qualquer caractere ASCII, exceto NULL. O padrão é um ponto de interrogação (?). Para obter informações sobre caracteres UTF-8 inválidos, consulte Erros no carregamento de caracteres multibyte.

COPY retorna o número de linhas que apresentavam caracteres UTF-8 inválidos e adiciona uma entrada à tabela do sistema STL_REPLACEMENTS para cada linha afetada, até 100 linhas, no máximo, para cada fatia de nó. Os caracteres UTF-8 inválidos adicionais também são substituídos, mas esses eventos substitutos não são registrados.

Se ACCEPTINVCHARS não for especificado, COPY retornará um erro sempre que encontrar um caractere UTF-8 inválido.

ACCEPTINVCHARS só é válido para colunas VARCHAR.

BLANKSASNULL

Carrega campos em branco, que consistem somente em caracteres de espaço em branco, como NULL. Essa opção se aplica somente a colunas CHAR e VARCHAR. Os campos em branco de outros tipos de dados, como INT, são sempre carregados com NULL. Por exemplo, uma string que contém três caracteres de espaço em sucessão (e qualquer outro caractere) é carregada como um NULL. O comportamento padrão, sem essa opção, é carregar os caracteres de espaço como estão.

DATEFORMAT [AS] {'dateformat_string' | 'auto' }

Se nenhum DATEFORMAT for especificado, o formato padrão será `'YYYY-MM-DD'`. Por exemplo, um formato válido alternativo é `'MM-DD-YYYY'`.

Se o comando COPY não reconhecer o formato dos valores de data ou hora, ou se os valores de data ou hora usarem formatos diferentes, use o argumento `'auto'` com o parâmetro DATEFORMAT ou TIMEFORMAT. O argumento `'auto'` reconhece diversos formatos não compatíveis ao usar uma string DATEFORMAT e TIMEFORMAT. A palavra-chave `'auto''` diferencia maiúsculas de minúsculas. Para obter mais informações, consulte Usar o reconhecimento automático com DATEFORMAT e TIMEFORMAT.

O formato de data pode incluir informações sobre horário (hora, minutos, segundos), mas essas informações são ignoradas. A palavra-chave AS é opcional. Para obter mais informações, consulte Strings DATEFORMAT e TIMEFORMAT.

EMPTYASNULL

Indica que o Amazon Redshift deve carregar campos CHAR e VARCHAR vazios como NULL. Os campos vazios de outros tipos de dados, como INT, são sempre carregados com NULL. Os campos vazios ocorrem quando os dados contêm dois delimitadores em sucessão sem caracteres entre os delimitadores. EMPTYASNULL e NULL AS '' (string vazia) produzem o mesmo comportamento.

ENCODING [AS] file_encoding

Especifica o tipo de codificação dos dados de carregamento. O comando COPY converte os dados da codificação especificada em UTF-8 durante o carregamento.

Os valores válidos para file_encoding são os seguintes:

- `UTF8`
- `UTF16`
- `UTF16LE`
- `UTF16BE`
- `ISO88591`

O padrão é `UTF8`.

Os nomes de arquivo de origem devem usar codificação UTF-8.

Os seguintes arquivos devem usar codificação UTF-8, mesmo se uma codificação diferente for especificada para os dados de carregamento:

- Arquivos de manifesto
- Arquivos JSONPaths

As string de argumento que acompanham os seguintes parâmetros devem usar UTF-8:

- FIXEDWIDTH 'fixedwidth_spec'
- ACCEPTINVCHARS 'replacement_char'
- DATEFORMAT 'dateformat_string'
- TIMEFORMAT 'timeformat_string'

- NULL AS 'null_string'

Os arquivos de dados de largura fixa devem usar codificação UTF-8. As larguras de campo se baseiam no número de caracteres, e não no número de bytes.

Todos os dados de carregamento devem usar a codificação especificada. Se encontrar uma codificação diferente, COPY ignorará o arquivo e retornará um erro.

Se você especificar `UTF16`, os dados deverão ter uma Byte Order Mark (BOM – Marca da ordem de bytes). Se souber se os dados UTF-16 são LE (Little-Endian) ou BE (Big-Endian), você poderá usar `UTF16LE` ou `UTF16BE`, independentemente da presença de uma BOM.

Para usar a codificação ISO-8859-1, especifique `ISO88591`. Para obter mais informações, consulte ISO/IEC 8859-1 na Wikipédia.

ESCAPE

Quando esse parâmetro é especificado, o caractere de barra invertida (\) em dados de entrada é tratado como um caractere de escape. O caractere logo depois do caractere de barra invertida será carregado na tabela como parte do valor de coluna atual, mesmo se for um caractere que normalmente atende a uma finalidade especial. Por exemplo, você poderá usar esse parâmetro para escapar o caractere delimitador, aspas, um caractere de nova linha integrado ou o próprio caractere de escape quando qualquer um desses caracteres for uma parte legítima de um valor de coluna.

Se especificar o parâmetro ESCAPE em combinação com o parâmetro REMOVEQUOTES, você poderá escapar e manter aspas (`'` ou `"`) que possam ser removidas de outra maneira. A string nula padrão, `\N`, funciona como está, mas também pode ser escapada nos dados de entrada como `\\N`. Desde que você não especifique uma string nula alternativa com o parâmetro NULL AS, `\N` e `\\N` produzem os mesmos resultados.

> **Note**
>
> O caractere de controle `0x00` (NUL) não pode ser escapado e deve ser removido dos dados de entrada ou convertido. Esse caractere é tratado como um marcador End Of Record (EOR – Fim do registro), truncando o restante do registro.

Você não pode usar o parâmetro ESCAPE para cargas FIXEDWIDTH, e você não pode especificar o caractere de escape propriamente dito; o caractere de escape é sempre o caractere

de barra invertida. Além disso, verifique se os dados de entrada contêm o caractere de escape nos locais apropriados.

Aqui estão alguns exemplos de dados de entrada e os dados carregados resultantes quando o parâmetro ESCAPE é especificado. O resultado da linha 4 pressupõe que o parâmetro REMOVEQUOTES também esteja especificado. Os dados de entrada consistem em dois campos delimitados por barra:

```
1|The quick brown fox\[newline]
jumped over the lazy dog.
2| A\\B\\C
3| A \| B \| C
4| 'A Midsummer Night\'s Dream'
```

Os dados carregados na coluna 2 têm esta aparência:

```
The quick brown fox
jumped over the lazy dog.
A\B\C
A|B|C
A Midsummer Night's Dream
```

> **Note**
>
> A aplicação do caractere de escape aos dados de entrada de uma carga é responsabilidade do usuário. Uma exceção a esse requisito é quando você recarrega dados que foram descarregados anteriormente com o parâmetro ESCAPE. Nesse caso, os dados já conterão os caracteres de escape necessários.

O parâmetro ESCAPE não interpreta octal, hex, Unicode nem outras notações da sequência de escape. Por exemplo, se os dados de origem contiverem o valor de alimentação de linha octal (`\012`) e você tentar carregar esses dados com o parâmetro ESCAPE, o Amazon Redshift carregará o valor `012` na tabela e não interpretará esse valor como uma alimentação de linha com escape.

Para escapar caracteres de nova linha em dados originados de plataformas do Microsoft Windows, você pode precisar usar dois caracteres de escape: um para o retorno de carro e um

para a alimentação de linha. Você também pode remover os retorno de carro antes de carregar o arquivo (por exemplo, usando o utilitário dos2unix).

EXPLICIT_IDS

Use EXPLICIT_IDS com tabelas que tenham colunas IDENTITY se você quiser substituir os valores gerados automaticamente por valores explícitos dos arquivos de dados de origem das tabelas. Se o comando incluir uma lista de colunas, essa lista deverá incluir as colunas IDENTITY para usar esse parâmetro. O formato de dados de valores EXPLICIT_IDS deve corresponder ao formato IDENTITY especificado pela definição de CREATE TABLE.

Depois de executar o comando COPY em relação a uma tabela com a opção EXPLICIT_IDS, o Amazon Redshift não conferirá a exclusividade de colunas IDENTITY na tabela.

Se uma coluna estiver definida com GENERATED BY DEFAULT AS IDENTITY, será possível copiá-la. Os valores são gerados ou atualizados com os valores fornecidos. A opção EXPLICIT_IDS não é necessária. COPY não atualiza a marca d'água alta de identidade.

Para obter um exemplo de um comando COPY usando EXPLICIT_IDS, consulte Carregar VENUE com valores explícitos para uma coluna IDENTITY.

FILLRECORD

Permite que arquivos de dados sejam carregados quando colunas contíguas não forem encontradas ao final de alguns dos registros. As colunas ausentes são carregadas como NULLs. Para formatos de texto e CSV, se a coluna ausente for uma coluna VARCHAR, as strings de comprimento zero serão carregadas em vez de NULLs. Para carregar NULLs em colunas VARCHAR a partir de texto e CSV, especifique a palavra-chave EMPTYASNULL. A substituição de NULL funcionará somente se a definição de coluna permitir NULLs.

Por exemplo, se a definição de tabela contiver quatro colunas CHAR anuláveis e um registro contiver os valores `apple, orange, banana, mango`, o comando COPY poderá carregar e preencher um registro que contenha somente os valores `apple, orange`. Os valores CHAR não encontrados seriam carregados como valores NULL.

IGNOREBLANKLINES

Ignora linhas em branco que contêm somente uma alimentação de linha em um arquivo de dados e não tenta carregá-las.

IGNOREHEADER [AS] number_rows

Trata as number_rows especificadas como um cabeçalho de arquivo e não as carrega. Use IGNOREHEADER para ignorar cabeçalhos de arquivo em todos os arquivos em uma carga paralela.

NULL AS 'null_string'

Carrega campos que correspondam a null_string como NULL, em que null_string pode ser qualquer string. Se os dados incluírem um terminador nulo, também conhecido como NUL (UTF-8 0000) ou zero binário (0x000), COPY o tratará como qualquer outro caractere. Por exemplo, um registro contendo '1' || NUL || '2' é copiado como string de comprimento de 3 bytes. Se um campo contiver somente NUL, você poderá usar NULL AS para substituir o terminador nulo por NULL, especificando `'\0'` ou `'\000'`. Por exemplo, `NULL AS '\0'` ou `NULL AS '\000'`. Se um campo contiver uma string que termina com NUL e NULL AS for especificado, a string será inserida com NUL ao final. Não use '\n' (nova linha) para o valor null_string. O Amazon Redshift reserva o uso de '\n' como um delimitador de linha. O null_string padrão é `'\N'`.

> **Note**
>
> Se você tentar carregar nulos em uma coluna definida como NOT NULL, o comando COPY falhará.

REMOVEQUOTES

Remove as aspas de strings nos dados recebidos. Todos os caracteres entre aspas, inclusive delimitadores, são mantidos. Se uma string tiver aspas simples ou duplas de abertura, mas nenhuma de fechamento correspondente, o comando COPY deixará de carregar essa linha e retornará um erro. A tabela a seguir mostra alguns exemplos simples de strings que contêm aspas e os valores carregados resultantes.

String de entrada	Valor carregado com opção REMOVEQUOTES
"O delimitador é um caractere de barra (|)"	O delimitador é um caractere de barra (|)
'Preto'	Preto
"Branco"	Branco

String de entrada	Valor carregado com opção REMOVEQUOTES
Azul'	Azul'
Azul'	Valor não carregado: condição de erro
"Azul	Valor não carregado: condição de erro
' 'Preto' ' '	' 'Preto' '
' '	<espaço em branco>

ROUNDEC

Arredonda para cima valores numéricos quando a escala do valor de entrada é maior que a escala da coluna. Por padrão, COPY trunca valores quando necessário para caber na escala da coluna. Por exemplo, se um valor de `20.259` for carregado em uma coluna DECIMAL(8,2), COPY truncará o valor em `20.25`, por padrão. Se ROUNDEC for especificado, COPY arredondará o valor para `20.26`. Como o comando INSERT sempre arredonda valores quando necessário de acordo com a escala da coluna, um comando COPY com o parâmetro ROUNDEC se comporta da mesma maneira que um comando INSERT.

TIMEFORMAT [AS] {'timeformat_string' | 'auto' | 'epochsecs' | 'epochmillisecs' }

Especifica o formato de hora. Se nenhum TIMEFORMAT for especificado, o formato padrão será `YYYY-MM-DD HH:MI:SS` para colunas TIMESTAMP ou `YYYY-MM-DD HH:MI:SSOF` para colunas TIMESTAMPTZ, em que `OF` é a diferença em relação ao Universal Coordinated Time (UTC – Tempo universal coordenado). Você não pode incluir um especificador de fuso horário no timeformat_string. Para carregar dados TIMESTAMPTZ em um formato diferente do formato padrão, especifique 'auto'; para obter mais informações, consulte Usar o reconhecimento automático com DATEFORMAT e TIMEFORMAT. Para obter mais informações sobre timeformat_string, consulte Strings DATEFORMAT e TIMEFORMAT.

O argumento `'auto'` reconhece diversos formatos não compatíveis ao usar uma string DATEFORMAT e TIMEFORMAT. Se o comando COPY não reconhecer o formato dos valores de data ou hora, ou se os valores de data e hora usarem formatos diferentes, use o argumento `'auto'` com o parâmetro DATEFORMAT ou TIMEFORMAT. Para obter mais informações, consulte Usar o reconhecimento automático com DATEFORMAT e TIMEFORMAT.

Se os dados de origem forem representados como hora epoch, ou seja o número de segundos ou milissegundos desde 1º de janeiro de 1970, 00:00:00 UTC, especifique `'epochsecs'` ou `'epochmillisecs'`.

As palavras-chave `'auto'`, `'epochsecs'` e `'epochmillisecs'` diferenciam maiúsculas de minúsculas.

A palavra-chave AS é opcional.

TRIMBLANKS

Remove os caracteres de espaço em branco à direita de uma string VARCHAR. Esse parâmetro se aplica somente a colunas com um tipo de dados VARCHAR.

TRUNCATECOLUMNS

Trunca dados em colunas no número apropriado de caracteres, de maneira que ele caiba na especificação da coluna. Aplica-se somente a colunas com um tipo de dados VARCHAR ou CHAR e linhas com 4 MB ou menos.

Operações de carregamento de dados

Gerencie o comportamento padrão da operação de carregamento para solucionar problemas ou reduzir o tempo de carregamento especificando os parâmetros a seguir.

- COMPROWS
- COMPUPDATE
- IGNOREALLERRORS
- MAXERROR
- NOLOAD
- STATUPDATE

Parâmetros

COMPROWS numrows

Especifica o número de linhas a serem usadas como o tamanho de exemplo para análise de compactação. A análise é executada em linhas de cada fatia de dados. Por exemplo, se você especificar `COMPROWS 1000000` (1.000.000) e o sistema contiver quatro fatias no total, não mais do que 250.000 linhas para cada fatia serão lidas e analisadas.

Se COMPROWS não for especificado, o padrão do tamanho de exemplo será 100.000 para cada fatia. Os valores de COMPROWS menores que o padrão de 100.000 linhas para cada fatia são atualizados automaticamente para o valor padrão. Porém, a compactação automática não acontecerá se o valor de dados carregados for insuficiente para produzir um exemplo significativo.

Se o número de COMPROWS for maior que o número de linhas no arquivo de entrada, o comando COPY continuará e executará a análise de compactação em todas as linhas disponíveis. O intervalo aceito para esse argumento é um número entre 1000 e 2147483647 (2,147,483,647).

COMPUPDATE [PRESET | { ON | TRUE } | { OFF | FALSE }],

Controla se as codificações de compactação são aplicadas automaticamente durante um comando COPY.

Quando COMPUPDATE é PRESET, o comando COPY escolhe a codificação de compactação para cada coluna se a tabela de destino estiver vazia; mesmo se as colunas já tiverem codificações diferentes de RAW. As codificações de coluna atualmente especificadas podem ser substituídas. A codificação para cada coluna é baseada no tipo de dados da coluna. Não é criada nenhuma amostra de dados. O Amazon Redshift atribui automaticamente a codificação de compactação da seguinte maneira:

- Colunas que são definidas como chaves de classificação são designadas a compactação RAW.
- Colunas que são definidas como tipos de dados BOOLEAN, REAL ou DOUBLE PRECISION recebem a compactação RAW.
- As colunas definidas como SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIMESTAMP ou TIMESTAMPTZ recebem a compactação AZ64.
- As colunas definidas como CHAR ou VARCHAR recebem a compactação LZO.

Quando COMPUPDATE for omitido, o comando COPY escolherá a codificação de compactação para cada coluna apenas se a tabela de destino estiver vazia e você não tiver especificado uma codificação (diferente de RAW) para nenhuma das colunas. A codificação de cada coluna é determinada pelo Amazon Redshift. Não é criada nenhuma amostra de dados.

Quando COMPUPDATE estiver definido como ON (ou TRUE) ou especificado sem uma opção, o comando COPY aplicará a compactação automática se a tabela estiver vazia, mesmo que as colunas da tabela já tenham codificações diferentes de RAW. As codificações de coluna

atualmente especificadas podem ser substituídas. A codificação de cada coluna se baseia em uma análise de dados de amostra. Para obter mais informações, consulte Carregamento de tabelas com compactação automática.

Quando COMPUPDATE é definido como OFF (ou FALSE), a compactação automática é desativada. As codificações de coluna não são alteradas.

Para obter informações sobre a tabela do sistema para analisar a compactação, consulte STL_ANALYZE_COMPRESSION.

IGNOREALLERRORS

Você pode especificar essa opção para ignorar todos os erros que ocorrem durante a operação de carregamento.

Você não pode especificar a opção IGNOREALLERRORS se especificar a opção MAXERROR. Você não pode especificar a opção IGNOREALLERRORS para formatos colunares, incluindo ORC e Parquet.

MAXERROR [AS] error_count

Se retornar o número de erros error_count ou mais, a carga falhará. Se retornar menos erros, a carga continuará e retornará uma mensagem INFO informando que não foi possível carregar o número de linhas. Use esse parâmetro para permitir que as cargas continuem quando determinadas linhas deixarem de ser carregadas na tabela por causa de erros de formatação ou outras inconsistências nos dados.

Defina esse valor como `0` ou `1`, se você quiser que a carga falhe assim que o primeiro erro ocorrer. A palavra-chave AS é opcional. O valor padrão MAXERROR é `0`, e o limite é `100000`.

O número real de erros relatados pode ser maior que o MAXERROR especificado por causa da natureza paralela do Amazon Redshift. Se o nó no cluster do Amazon Redshift detectar que MAXERROR foi excedido, cada nó relatará todos os nós encontrados.

NOLOAD

Verifica a validade do arquivo de dados sem efetivamente carregar os dados. Use o parâmetro NOLOAD para garantir que o arquivo de dados seja carregado sem erros antes de executar o carregamento de dados real Executar COPY com o parâmetro NOLOAD é muito mais rápido do que carregar os dados porque somente analisa os arquivos.

STATUPDATE [{ ON | TRUE } | { OFF | FALSE }]

Regula a computação automática e a atualização de estatísticas do otimizador ao final de um comando COPY bem-sucedido. Por padrão, se o parâmetro STATUPDATE não for usado, as estatísticas serão atualizadas automaticamente se a tabela estiver inicialmente vazia.

Sempre que o processamento de dados em uma tabela não vazia alterar significativamente o tamanho da tabela, recomendamos atualizar as estatísticas executando um comando ANALYZE ou usando o argumento STATUPDATE ON.

Com STATUPDATE ON (ou TRUE), as estatísticas são atualizadas automaticamente, independente da tabela estar inicialmente vazia. Se STATUPDATE for usado, o usuário atual não deverá ser o proprietário da tabela ou um superusuário. Se STATUPDATE não for especificado, somente a permissão INSERT será necessária.

Com STATUPDATE OFF (ou FALSE), as estatísticas jamais são atualizadas.

Para obter informações adicionais, consulte Análise de tabelas.

Lista de parâmetros alfabética

A lista a seguir fornece links para cada descrição de parâmetro do comando COPY, classificados em ordem alfabética.

- ACCEPTANYDATE
- ACCEPTINVCHARS
- AVRO
- BLANKSASNULL
- BZIP2
- COMPROWS
- COMPUPDATE
- CSV
- DATEFORMAT
- DELIMITER
- EMPTYASNULL

- ENCODING
- ENCRYPTED
- ESCAPE
- EXPLICIT_IDS
- FILLRECORD
- FIXEDWIDTH
- FORMAT
- FROM
- GZIP
- the section called “IAM_ROLE”
- IGNOREALLERRORS
- IGNOREBLANKLINES
- IGNOREHEADER
- JSON format for COPY
- LZOP
- MANIFEST
- MASTER_SYMMETRIC_KEY
- MAXERROR
- NOLOAD
- NULL AS
- READRATIO
- REGION
- REMOVEQUOTES
- ROUNDEC
- SHAPEFILE
- SSH
- STATUPDATE
- TIMEFORMAT

- TRIMBLANKS
- TRUNCATECOLUMNS
- ZSTD

Observações de uso

Tópicos

- Permissões para acessar outros recursos da AWS
- Usar COPY com aliases de ponto de acesso do Amazon S3
- Carregar dados multibyte do Amazon S3
- Carregar uma coluna do tipo de dados GEOMETRY ou GEOGRAPHY
- Carregando o tipo de dados HLLSKETCH
- Carregar uma coluna do tipo de dados VARBYTE
- Erros durante a leitura de vários arquivos
- COPY no formato JSON
- COPY de formatos de dados colunar
- Strings DATEFORMAT e TIMEFORMAT
- Usar o reconhecimento automático com DATEFORMAT e TIMEFORMAT

Permissões para acessar outros recursos da AWS

Para mover dados entre o cluster e outro recurso da AWS, como Amazon S3, Amazon DynamoDB, Amazon EMR, ou Amazon EC2, o cluster deve ter permissão para acessar o recurso e realizar as ações necessárias. Por exemplo, para carregar dados do Amazon S3, COPY deve ter acesso LIST para o bucket e acesso GET para os objetos do bucket. Para obter informações sobre permissões mínimas, consulte Permissões do IAM para COPY, UNLOAD e CREATE LIBRARY.

Para receber autorização a fim de acessar o recurso, o cluster deve ser autenticado usando o seguinte método de autenticação:

- Controle de acesso com base em função – Para controle de acesso baseado na função, você especifica uma função do AWS Identity and Access Management (IAM) usada pelo cluster na autenticação e na autorização. Para resguardar as credenciais da AWS e os dados sigilosos, é altamente recomendável usar a autenticação baseada na função.

Controle de acesso com base em função

Com controle de acesso baseado na função, o cluster pressupõe temporariamente uma função do IAM em seu nome. Então, com base nas autorizações concedidas para a função, seu cluster pode acessar os recursos da AWS necessários.

Criar um perfil do IAM é semelhante a conceder permissões a um usuário no sentido de ser uma identidade da AWS com políticas de permissão que determinam o que a identidade pode e não pode fazer na AWS. No entanto, em vez de ser exclusivamente associada a um usuário, uma função pode ser assumida por qualquer entidade que precise dela. Além disso, uma função não tem credenciais (uma senha ou chaves de acesso) associadas. Em vez disso, se uma função estiver associada a um cluster, as chaves de acesso serão criadas dinamicamente e fornecidas para o cluster.

Recomendamos o uso do controle de acesso baseado em função, pois ele fornece um controle refinado e mais seguro do acesso aos recursos da AWS e dados sigilosos do usuário, além de proteger suas credenciais da AWS.

A autenticação com base na função oferece os seguintes benefícios:

- Você pode usar as ferramentas do IAM padrão da AWS para definir uma função do IAM e associar a função a vários clusters. Quando você modifica a política de acesso de uma função, as alterações são aplicadas automaticamente a todos os clusters que usam a função.
- Você pode definir políticas do IAM refinadas que concedem permissões para clusters específicos e usuários de banco de dados acessarem recursos e ações específicos da AWS.
- O cluster obtém credenciais de sessão temporária em tempo de execução e atualiza as credenciais conforme necessário até a operação ser concluída.
- O ID de chave de acesso e o ID de chave de acesso secreta não são armazenados nem transmitidos no código SQL.

Para usar o controle de acesso com base em função, você primeiro deve criar uma função do IAM usando o tipo de função de serviço do Amazon Redshift e, em seguida, anexar a função ao seu cluster. A função deve ter, pelo menos, as permissões listadas em Permissões do IAM para COPY, UNLOAD e CREATE LIBRARY. Para conhecer as etapas de criação de um perfil do IAM e anexá-lo ao cluster, consulte “Autorizar o Amazon Redshift a acessar outros serviços da AWS em seu nome” no Guia de gerenciamento de clusters do Amazon Redshift.

Você pode adicionar uma função a um cluster ou visualizar as funções associadas a um cluster usando o Console de Gerenciamento do Amazon Redshift, a CLI ou a API. Para obter mais

informações, consulte “Autorizar operações COPY, UNLOAD, CREATE EXTERNAL FUNCTION e CREATE EXTERNAL SCHEMA usando funções do IAM” no Guia de gerenciamento de clusters do Amazon Redshift.

Quando você cria uma função do IAM, o IAM retorna um nome de recurso da Amazon (ARN) para a função. Para especificar uma função do IAM, forneça o ARN da função com o parâmetro the section called “Usar IAM_ROLE”.

Por exemplo, suponhamos que a função a seguir esteja anexada ao cluster.

```
"IamRoleArn": "arn:aws:iam::0123456789012:role/MyRedshiftRole"
```

O exemplo de comando COPY a seguir usa o parâmetro IAM_ROLE com o ARN no exemplo anterior para autenticação e acesso ao Amazon S3.

```
copy customer from 's3://amzn-s3-demo-bucket/mydata'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Além disso, um superusuário pode conceder o privilégio ASSUMEROLE aos usuários e grupos do banco de dados para fornecer acesso a uma função para operações COPY. Para mais informações, consulte GRANT.

Credenciais de segurança temporárias

Ao usar o controle de acesso baseado em perfil, o Amazon Redshift utiliza automaticamente credenciais de segurança temporárias em seu nome. As credenciais de segurança temporárias fornecem segurança avançada, pois possuem vidas úteis curtas e não podem ser reutilizadas depois que expiram.

Quando você autoriza o acesso usando um perfil do IAM, seu cluster assume temporariamente esse perfil e recebe credenciais de sessão temporárias durante a execução. Seu cluster atualiza essas credenciais conforme necessário até que a operação seja concluída, para que você não precise fornecê-las ou gerenciá-las sozinho. Para obter mais informações, consulte Controle de acesso com base em função.

Permissões do IAM para COPY, UNLOAD e CREATE LIBRARY

O perfil do IAM referido pelo parâmetro IAM_ROLE deve ter, pelo menos, as seguintes permissões:

- Para COPY no Amazon S3, a permissão para LIST o bucket do Amazon S3 e GET os objetos do Amazon S3 do carregados, além do arquivo manifesto, se algum for usado.

- Para COPY do Amazon S3, Amazon EMR e hosts remotos (SSH) com dados formatados em JSON, permissão para LIST e GET o arquivo JSONPaths no Amazon S3, se algum for usado.
- Para COPY no DynamoDB, permissão para SCAN e DESCRIBE a tabela do DynamoDB que está sendo carregada.
- Para COPY a partir de um cluster do Amazon EMR, permissão para a ação `ListInstances` no cluster do Amazon EMR.
- Para UNLOAD no Amazon S3, as permissões para GET, LIST e PUT para o bucket do Amazon S3 no qual os arquivos de dados estão sendo descarregados.
- Para CREATE LIBARY do Amazon S3, permissão para LIST o bucket do Amazon S3 e GET os objetos do Amazon S3 a serem importados

> **Note**
>
> Se você receber a mensagem de erro `S3ServiceException: Access Denied` ao executar um comando COPY, UNLOAD ou CREATE LIBRARY, significa que o cluster não tem permissões de acesso apropriadas para o Amazon S3.

É possível gerenciar permissões do IAM anexando uma política do IAM a um perfil do IAM anexado ao cluster, ao usuário ou ao grupo ao qual o usuário pertence. Por exemplo, a política gerenciada `AmazonS3ReadOnlyAccess` concede permissões LIST e GET a recursos do Amazon S3. Para obter mais informações sobre o gerenciamento de políticas, consulte Gerenciando políticas do IAM no Manual do usuário do IAM.

Usar COPY com aliases de ponto de acesso do Amazon S3

COPY oferece suporte a aliases de ponto de acesso do Amazon S3. Para obter mais informações, consulte Usar um alias em estilo de bucket para seu ponto de acesso no Guia do usuário do Amazon Simple Storage Service.

Carregar dados multibyte do Amazon S3

Se seus dados incluem caracteres multibyte não ASCII (tal como caracteres chineses ou cirílicos), você deve carregar os dados em colunas VARCHAR. O tipo de dados VARCHAR é compatível com caracteres UTF-8 de quatro bytes, mas o tipo de dados CHAR aceita apenas caracteres ASCII de único byte. Você não pode carregar caracteres de cinco ou mais bytes de comprimento em tabelas do Amazon Redshift. Para obter mais informações, consulte Caracteres multibyte.

Carregar uma coluna do tipo de dados GEOMETRY ou GEOGRAPHY

Você pode usar COPY e, colunas GEOMETRY ou GEOGRAPHY de dados em um arquivo de texto delimitado por caracteres, como um arquivo CSV. Os dados devem estar na forma do formato binário reconhecido (WKB ou EWKB) ou no formato de texto reconhecido (WKT ou EWKT) e caber no tamanho máximo de uma única linha de entrada para o comando COPY. Para obter mais informações, consulte COPY.

Para obter informações sobre como carregar a partir de um arquivo de forma, consulte Carregar um shapefile no Amazon Redshift.

Para obter mais informações sobre tipos de dados GEOMETRY ou GEOGRAPHY, consulte Consultar dados espaciais no Amazon Redshift.

Carregando o tipo de dados HLLSKETCH

Você pode copiar esboços HLL apenas em formato esparso ou denso compatível com o Amazon Redshift. Para usar o comando COPY em esboços HyperLogLog, use o formato Base64 para esboços de HyperLogLog densos e o formato JSON para esboços de HyperLogLog esparsos. Para obter mais informações, consulte Funções HyperLogLog.

O exemplo a seguir importa dados de um arquivo CSV para uma tabela usando CREATE TABLE e COPY. Primeiro, o exemplo cria a tabela t1 usando CREATE TABLE.

```
CREATE TABLE t1 (sketch hllsketch, a bigint);
```

Em seguida, ele usa COPY para importar dados de um arquivo CSV para a tabela t1.

```
COPY t1 FROM s3://amzn-s3-demo-bucket/unload/' IAM_ROLE
 'arn:aws:iam::0123456789012:role/MyRedshiftRole' NULL AS 'null' CSV;
```

Carregar uma coluna do tipo de dados VARBYTE

É possível carregar dados de arquivos no formato CSV, Parquet e ORC. Para CSV, os dados são carregados de um arquivo em representação hexadecimal dos dados VARBYTE. Não é possível carregar dados VARBYTE com a opção FIXEDWIDTH. Não há suporte para as opções ADDQUOTES ou REMOVEQUOTES de COPY. Uma coluna VARBYTE não pode ser usada como coluna de partição.

Erros durante a leitura de vários arquivos

O comando COPY é atômico e transacional. Em outras palavras, mesmo quando o comando COPY lê dados de vários arquivos, todo o processo é tratado como uma única transação. Se encontrar um erro ao ler um arquivo, COPY será repetido automaticamente até o processo expirar (consulte statement_timeout) ou se o download dos dados não puder ser feito no Amazon S3 por um período prolongado (entre 15 e 30 minutos), garantindo que cada arquivo seja carregado somente uma vez. Se o comando COPY falhar, toda a transação será cancelada, e todas as alterações serão revertidas. Para obter mais informações sobre como processar erros de carga, consulte Solução de problemas de carregamento de dados.

Depois que for iniciado com êxito, não ocorrerá uma falha em um comando COPY se a sessão for concluída, por exemplo, quando o cliente se desconectar. Porém, se o comando COPY estiver dentro de um bloco de transação BEGIN … END não concluído porque a sessão foi encerrada, toda a transação, inclusive COPY, será restaurada. Para obter mais informações sobre transações, consulte BEGIN.

COPY no formato JSON

A estrutura de dados JSON é composta de um conjunto de objetos ou matrizes. Um objeto JSON começa e termina com chaves e contém uma coleção de pares de nome-valor desordenada. Cada nome e valor são separados por um dois-pontos, e os pares são separados por vírgulas. O nome é uma string entre aspas duplas. As aspas devem ser simples (0x22), e não inclinadas ou "inteligentes".

Uma matriz JSON começa e termina com colchetes e contém uma coleção ordenada de valores separados por vírgulas. Um valor pode ser uma string entre aspas, um número, um verdadeiro ou falso Booliano, um nulo, um objeto JSON ou uma matriz.

Os objetos JSON e as matrizes podem ser aninhados, o que possibilita uma estrutura de dados hierárquica. O exemplo a seguir mostra uma estrutura de dados JSON com dois objetos válidos.

```
{
    "id": 1006410,
    "title": "Amazon Redshift Database Developer Guide"
}
{
    "id": 100540,
    "name": "Amazon Simple Storage Service User Guide"
}
```

A seguir, os mesmos dados como duas matrizes JSON.

```
[
    1006410,
    "Amazon Redshift Database Developer Guide"
]
[
    100540,
    "Amazon Simple Storage Service User Guide"
]
```

Opções COPY para JSON

Você pode especificar as seguintes opções ao usar COPY com dados de formato JSON:

- `'auto'` — COPY carrega automaticamente campos do arquivo JSON.
- `'auto ignorecase'` — COPY carrega automaticamente campos do arquivo JSON enquanto ignora as maiúsculas e minúsculas de nomes de campo.
- `s3://jsonpaths_file` — COPY usa um arquivo JSONPaths para analisar nos dados de origem do JSON. Um arquivo JSONPaths é um arquivo de texto que contém um objeto JSON com o nome `"jsonpaths"` em par com uma matriz de expressões JSONPath. Se o nome for qualquer string além de `"jsonpaths"`, COPY usará o argumento `'auto'`, em vez de usar o arquivo JSONPaths.

Para obter exemplos que mostrem como carregar dados usando `'auto'`, `'auto ignorecase'` ou um arquivo JSONPaths e usando objetos JSON ou matrizes, consulte Copiar de exemplos JSON.

Opção JSONPath

Na sintaxe do Amazon Redshift COPY, uma expressão JSONPath especifica o caminho explícito para um único elemento de nome em uma estrutura de dados hierárquica JSON, usando notação de colchetes ou notação de ponto. O Amazon Redshift não dá suporte a elementos JSONPath, como caracteres curinga ou expressões de filtro, que podem ser resolvidos para um caminho ambíguo ou vários elementos de nome. Dessa forma, o Amazon Redshift não pode analisar estruturas de dados complexas de vários níveis.

Este é um exemplo de um arquivo JSONPaths com expressões JSONPath usando a notação de colchetes. O cifrão ($) representa a estrutura no nível da raiz.

```
{
```

```
    "jsonpaths": [
       "$['id']",
       "$['store']['book']['title']",
 "$['location'][0]"
    ]
}
```

No exemplo anterior, `$['location'][0]` referencia o primeiro elemento em uma matriz. JSON usa uma indexação de matriz baseada em zero. Os índices de matriz devem ser inteiros positivos (maiores ou iguais a zero).

O exemplo a seguir mostra o caminho JSONPath anterior usando notação de pontos.

```
{
    "jsonpaths": [
       "$.id",
       "$.store.book.title",
 "$.location[0]"
    ]
}
```

Você não pode misturar notações de colchetes e pontos na matriz `jsonpaths`. Os colchetes podem ser usados nas notações de colchetes e de pontos para referenciar um elemento de matriz.

Ao usar uma notação de pontos, as expressões JSONPath não deverão conter os seguintes caracteres:

- Aspas retas únicas (')
- Ponto final ou ponto (.)
- Colchetes ([ ]), a menos que sejam usados para referenciar um elemento de matriz

Se o valor no par de nome-valor referenciado por uma expressão JSONPath for um objeto ou uma matriz, todo o objeto ou matriz será carregado como uma string, incluindo as chaves ou os colchetes. Por exemplo, suponhamos que os dados JSON contenham o objeto a seguir.

```
{
    "id": 0,
    "guid": "84512477-fa49-456b-b407-581d0d851c3c",
    "isActive": true,
    "tags": [
```

```
        "nisi",
        "culpa",
        "ad",
        "amet",
        "voluptate",
        "reprehenderit",
        "veniam"
    ],
    "friends": [
        {
            "id": 0,
            "name": "Martha Rivera"
        },
        {
            "id": 1,
            "name": "Renaldo"
        }
    ]
}
```

Em seguida, a expressão JSONPath `$['tags']` retorna o valor a seguir.

```
"["nisi","culpa","ad","amet","voluptate","reprehenderit","veniam"]"
```

Em seguida, a expressão JSONPath `$['friends'][1]` retorna o valor a seguir.

```
"{"id": 1,"name": "Renaldo"}"
```

Cada expressão JSONPath na matriz `jsonpaths` corresponde a uma coluna na tabela de destino do Amazon Redshift. A ordem dos elementos de matriz `jsonpaths` deverá corresponder à ordem das colunas na tabela de destino ou na lista de colunas, se uma lista de colunas for usada.

Para obter exemplos que mostrem como carregar dados usando o argumento `'auto'` ou um arquivo JSONPaths, e usando objetos JSON ou matrizes, consulte Copiar de exemplos JSON.

Para obter informações sobre como copiar vários arquivos JSON, consulte Uso de um manifesto para especificar arquivos de dados.

Caracteres de escape em JSON

COPY carrega `\n` como um caractere de nova linha e `\t` como um caractere de tabulação. Para carregar uma barra invertida, use uma barra invertida de escape (`\\`).

Por exemplo, suponhamos que você tenha o JSON a seguir em um arquivo chamado `escape.json` no bucket `s3://amzn-s3-demo-bucket/json/`.

```
{
  "backslash": "This is a backslash: \\",
  "newline": "This sentence\n is on two lines.",
  "tab": "This sentence \t contains a tab."
}
```

Execute os comandos a seguir para criar a tabela ESCAPES e carregar o JSON.

```
create table escapes (backslash varchar(25), newline varchar(35), tab varchar(35));

copy escapes from 's3://amzn-s3-demo-bucket/json/escape.json'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
format as json 'auto';
```

Consulte a tabela ESCAPES para exibir os resultados.

```
select * from escapes;

       backslash        |      newline      |               tab
------------------------+-------------------+-----------------------------------
 This is a backslash: \ | This sentence     | This sentence    contains a tab.
                        :  is on two lines.
(1 row)
```

Perda de precisão numérica

É possível perder precisão ao carregar números de arquivos de dados no formato JSON para uma coluna definida como um tipo de dados numérico. Alguns valores de ponto flutuante não são representados exatamente nos sistemas de computação. Como resultado, os dados copiados de um arquivo JSON podem não ser arredondados conforme esperado. Para evitar a perda de precisão, recomendamos que você use uma das seguintes alternativas:

- Representa o número como uma string, colocando o valor em caracteres de aspas duplas.
- Use ROUNDEC para arredondar o número em vez de truncar.
- Em vez de usar arquivos JSON ou Avro, use arquivos de texto CSV, delimitados por caracteres, ou de largura fixa.

COPY de formatos de dados colunar

COPY pode carregar dados do Amazon S3 nos seguintes formatos colunares:

- ORC
- Parquet

Para obter exemplos do uso de COPY a partir de formatos de dados colunares, consulte Exemplos de COPY.

O comando COPY aceita dados formatados colunares com as seguintes considerações:

- O bucket da Amazon S3 deve estar na mesma região da AWS que o banco de dados do Amazon Redshift.
- Para acessar seus dados do Amazon S3 por meio de um endpoint da VPC, configure o acesso usando políticas e perfis do IAM conforme descrito em Consultar tabelas de data lake com roteamento aprimorado da VPC no Guia de gerenciamento do Amazon Redshift.
- COPY não aplicará automaticamente as codificações de compactação.
- Apenas os parâmetros COPY a seguir são aceitos:
  - ACCEPTINVCHARS ao copiar de um arquivo ORC ou Parquet.
  - FILLRECORD
  - FROM
  - IAM_ROLE
  - STATUPDATE
  - MANIFEST
  - EXPLICIT_IDS
- Se COPY encontrar um erro ao carregar, o comando falhará. ACCEPTANYDATE e MAXERROR não são compatíveis com tipos de dados colunares.
- Mensagens de erro são enviadas para o cliente SQL. Alguns erros são registrados em log em STL_LOAD_ERRORS e STL_ERROR.
- COPY insere valores nas colunas da tabela de destino na mesma ordem das colunas ocorridas nos arquivos de dados colunares. O número de colunas na tabela de destino e o número de colunas no arquivo de dados devem combinar.
- Se o arquivo especificado para a operação COPY incluir uma das seguintes extensões, o Amazon Redshift descompactará os dados sem a necessidade de adicionar parâmetros:

  - `.gz`
  - `.snappy`
  - `.bz2`
- COPY dos formatos de arquivo Parquet e ORC usa o Redshift Spectrum e o acesso de bucket. Para usar COPY para esses formatos, verifique se não há políticas do IAM bloqueando o uso de URLs pré-assinados do Amazon S3. Os URLs pré-assinados gerados pelo Amazon Redshift são válidos por uma hora para que o Amazon Redshift tenha tempo suficiente para carregar todos os arquivos do bucket do Amazon S3. Um URL pré-assinado exclusivo é gerado para cada arquivo verificado pelo comando COPY com base em formatos de dados colunares. Para políticas de bucket que incluem uma ação `s3:signatureAge`, o valor deve ser definido como pelo menos 3.600.000 milissegundos. Para acessar mais informações, acesse Consultar tabelas de data lake com roteamento de VPC aprimorado.
- O parâmetro REGION não é compatível com COPY de formatos de dados colunares. Mesmo que o bucket do Amazon S3 e o banco de dados estejam na mesma Região da AWS, você poderá encontrar um erro, como REGION argument is not supported for PARQUET based COPY.
- COPY de formatos colunares agora aceita a escalabilidade de simultaneidade. Para habilitar a escalabilidade de simultaneidade, consulte Configurar filas de escalabilidade de simultaneidade.

Strings DATEFORMAT e TIMEFORMAT

O comando COPY usa as opções DATEFORMAT e TIMEFORMAT para analisar valores de data e hora nos dados de origem. DATEFORMAT e TIMEFORMAT são strings formatadas que devem corresponder ao formato dos valores de data e hora dos dados de origem. Por exemplo, um comando COPY que carrega dados de origem com o valor da data `Jan-01-1999` deve incluir a seguinte string DATEFORMAT:

```
COPY ...
            DATEFORMAT AS 'MON-DD-YYYY'
```

Para obter mais informações sobre como gerenciar conversões de dados COPY, consulte Parâmetros de conversão de dados.

As strings DATEFORMAT e TIMEFORMAT podem conter separadores de data e hora (como “`-`”, “`/`” ou “`:`”), bem como os formatos datepart e timepart na tabela a seguir.

> **Note**
>
> Se não for possível corresponder os formatos dos valores de data e hora com os dateparts e timeparts a seguir, ou se você tiver valores de data e hora com formatos diferentes entre si, use o argumento `'auto'` com o parâmetro DATEFORMAT ou TIMEFORMAT. O argumento `'auto'` reconhece diversos formatos incompatíveis ao usar uma string DATEFORMAT e TIMEFORMAT. Para obter mais informações, consulte Usar o reconhecimento automático com DATEFORMAT e TIMEFORMAT.

Datepart ou timepart	Significado
YY	Ano sem século
YYYY	Ano com século
MM	Mês como um número
MON	Mês como um nome (nome abreviado ou completo)
DD	Dia do mês como um número
HH ou HH24	Hora (relógio de 24 horas) **Note** Nas strings de formato DATETIME das funções SQL, HH é o mesmo que HH12. No entanto, nas strings DATEFORMAT e TIMEFORMAT de COPY, HH é o mesmo que HH24.
HH12	Hora (relógio de 12 horas)
MI	Minutos
SS	Segundos

Datepart ou timepart	Significado
AM ou PM	Indicador de meridiano (para relógio de 12 horas)

O formato de data padrão é YYYY-MM-DD. O formato de carimbo de data/hora padrão sem fuso horário (TIMESTAMP) é YYYY-MM-DD HH:MI:SS. O formato de carimbo de data/hora padrão com formato de fuso horário (TIMESTAMPTZ) é YYYY-MM-DD HH:MI:SSOF, em que OF é o deslocamento de UTC (por exemplo, -8:00). Você não pode incluir um especificador de fuso horário (TZ, tz ou OF) no timeformat_string. O campo de segundos (SS) também dá suporte a segundos fracionários até um nível de microssegundo. Para carregar dados TIMESTAMPTZ em um formato diferente do formato padrão, especifique 'auto'.

A seguir estão alguns exemplos de datas e horas que você pode encontrar em seus dados de origem e as strings DATEFORMAT ou TIMEFORMAT correspondentes a eles.

Exemplo de data e hora dos dados de origem	Sintaxe de DATEFORMAT ou TIMEFORMAT
03/31/2003	DATEFORMAT AS 'MM/DD/YYYY'
31 de março de 2003	DATEFORMAT AS 'MON DD, YYYY'
03.31.2003 18:45:05 03.31.2003 18:45:05.123456	TIMEFORMAT AS 'MM.DD.YYYY HH:MI:SS'

Exemplo

Para ver um exemplo do uso de TIMEFORMAT, consulte Carregar um time stamp ou date stamp.

Usar o reconhecimento automático com DATEFORMAT e TIMEFORMAT

Se você especificar `'auto'` como o argumento para o parâmetro DATEFORMAT ou TIMEFORMAT, o Amazon Redshift reconhecerá automaticamente e converterá o formato de data ou o formato de hora nos dados de origem. Por exemplo:

```
copy favoritemovies from 'dynamodb://ProductCatalog'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
dateformat 'auto';
```

Quando usado com o argumento `'auto'` para DATEFORMAT e TIMEFORMAT, COPY reconhece e converte os formatos de data e hora listados na tabela em Strings DATEFORMAT e TIMEFORMAT. Além disso, o argumento `'auto'` reconhece os seguintes formatos não compatíveis ao usar uma string DATEFORMAT e TIMEFORMAT.

Formato	Exemplo de string de entrada válida
ISO 8601	2019-02-11T05:09:12.195Z
Juliano	J2451187
BC	Jan-08-95 BC
YYYYMMDD HHMISS	19960108 040809
YYMMDD HHMISS	960108 040809
YYYY.DDD	1996.008
YYYY-MM-DD HH:MI:SS.SSS	1996-01-08 04:05:06.789
DD Mon HH:MI:SS YYYY TZ	17 Dez 07:37:16 1997 PST
MM/DD/YYYY HH:MI:SS.SS TZ	12/17/1997 07:37:16.00 PST
YYYY-MM-DD HH:MI:SS+/-TZ	1997-12-17 07:37:16-08
DD.MM.YYYY HH:MI:SS TZ	12.17.1997 07:37:16.00 PST

O reconhecimento automático não dá suporte a epochsecs e epochmillisecs.

Para testar se um valor de data ou de data e hora será convertido automaticamente, use uma função CAST para tentar converter a string em um valor de data ou de data e hora. Por exemplo, os comandos a seguir testam o valor de data e hora `'J2345678 04:05:06.789'`:

```
create table formattest (test char(21));
insert into formattest values('J2345678 04:05:06.789');
select test, cast(test as timestamp) as timestamp, cast(test as date) as date from
 formattest;

        test          |      timestamp      | date
----------------------+---------------------+------------
J2345678 04:05:06.789   1710-02-23 04:05:06 1710-02-23
```

Se os dados de origem de uma coluna DATA incluírem informações de hora, o componente de hora será truncado. Se os dados de origem de uma coluna TIMESTAMP omitirem informações de hora, 00:00:00 será usado para o componente de hora.

Exemplos de COPY

> **Note**
>
> Estes exemplos contêm quebras de linha para garantir legibilidade. Não inclua quebras de linha nem espaços na string credentials-args.

Tópicos

- Carregar FAVORITEMOVIES de uma tabela do DynamoDB
- Carregar LISTING de um bucket do Amazon S3
- Carregar LISTING de um cluster do Amazon EMR
- Uso de um manifesto para especificar arquivos de dados
- Carregar LISTING de um arquivo delimitado por barras (delimitador padrão)
- Carregar LISTING usando dados de colunas no formato Parquet
- Carregar LISTING usando dados de colunas no formato ORC
- Carregar EVENT com opções
- Carregar VENUE de um arquivo de dados de largura fixa
- Carregar CATEGORY de um arquivo CSV

- Carregar VENUE com valores explícitos para uma coluna IDENTITY
- Carregar TIME de um arquivo GZIP delimitado por barras
- Carregar um time stamp ou date stamp
- Carregar dados de um arquivo com valores padrão
- Dados de COPY com a opção ESCAPE
- Copiar de exemplos JSON
- Copiar de exemplos Avro
- Preparar arquivos para COPY com a opção ESCAPE
- Carregar um shapefile no Amazon Redshift
- Comando COPY com a opção NOLOAD
- Comando COPY com um delimitador multibyte e a opção ENCODING

Carregar FAVORITEMOVIES de uma tabela do DynamoDB

Os SDKs da AWS incluem um exemplo simples de como criar uma tabela do DynamoDB chamada Movies. (Para esse exemplo, consulte Conceitos básicos do DynamoDB.) O exemplo a seguir carrega a tabela MOVIES do Amazon Redshift com dados da tabela do DynamoDB. A tabela do Amazon Redshift já deve existir no banco de dados.

```
copy favoritemovies from 'dynamodb://Movies'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
readratio 50;
```

Carregar LISTING de um bucket do Amazon S3

O exemplo a seguir carrega LISTING de um bucket do Amazon S3. O comando COPY carrega todos os arquivos na pasta `/data/listing/`.

```
copy listing
from 's3://amzn-s3-demo-bucket/data/listing/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Carregar LISTING de um cluster do Amazon EMR

O exemplo a seguir carrega a tabela SALES com dados delimitados por tabulação de arquivos compactados por lzop em um cluster do Amazon EMR. COPY carrega cada arquivo na pasta `myoutput/` que começa com `part-`.

```
copy sales
from 'emr://j-SAMPLE2B500FC/myoutput/part-*'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter '\t' lzop;
```

O exemplo a seguir carrega a tabela SALES com dados formatados JSON em um cluster do Amazon EMR. COPY carrega cada arquivo na pasta `myoutput/json/`.

```
copy sales
from 'emr://j-SAMPLE2B500FC/myoutput/json/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
JSON 's3://amzn-s3-demo-bucket/jsonpaths.txt';
```

Uso de um manifesto para especificar arquivos de dados

É possível usar um manifesto para garantir que o comando COPY carregue todos os arquivos necessários, e somente os arquivos necessários, do Amazon S3. Você também pode usar um manifesto ao precisar carregar vários arquivos de buckets diferentes ou arquivos que não tenham o mesmo prefixo.

Por exemplo, suponha que você precise carregar os seguintes três arquivos: `custdata1.txt`, `custdata2.txt` e `custdata3.txt`. Você pode usar o seguinte comando para carregar todos os arquivos em `amzn-s3-demo-bucket` que comecem com `custdata` especificando um prefixo:

```
copy category
from 's3://amzn-s3-demo-bucket/custdata'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Se somente dois dos arquivos existirem por causa de um erro, COPY carrega somente esses dois arquivos e será concluído com êxito, resultando em uma carga de dados incompleta. Se o bucket também contiver um arquivo indesejado que acabe usando o mesmo prefixo, como um arquivo chamado `custdata.backup` por exemplo, COPY carregará esse arquivo também, resultando no carregamento de dados indesejados.

Para garantir que todos os arquivos necessários sejam carregados e evitar que arquivos indesejados sejam carregados, é possível usar um arquivo de manifesto. O manifesto é um arquivo de texto formatado em JSON que lista os arquivos a serem processados pelo comando COPY. Por exemplo, o manifesto a seguir carrega os três arquivos no exemplo anterior.

```
{
```

```
   "entries":[
      {
         "url":"s3://amzn-s3-demo-bucket/custdata.1",
         "mandatory":true
      },
      {
         "url":"s3://amzn-s3-demo-bucket/custdata.2",
         "mandatory":true
      },
      {
         "url":"s3://amzn-s3-demo-bucket/custdata.3",
         "mandatory":true
      }
   ]
}
```

O sinalizador `mandatory` opcional indica se COPY deverá ser encerrado se o arquivo não existir. O padrão é `false`. Independentemente de qualquer configuração obrigatória, o COPY encerra se nenhum arquivo for encontrado. Neste exemplo, COPY retornará um erro se algum dos arquivos não for encontrado. Arquivos indesejados que possam ter sido escolhidos se você tiver especificado somente um prefixo de chaves, como `custdata.backup`, são ignorados, pois não estão no manifesto.

Ao carregar arquivos de dados em ORC ou em formato Parquet, um campo `meta` é necessário, conforme exibido no seguinte exemplo.

```
{
   "entries":[
      {
         "url":"s3://amzn-s3-demo-bucket1/orc/2013-10-04-custdata",
         "mandatory":true,
         "meta":{
            "content_length":99
         }
      },
      {
         "url":"s3://amzn-s3-demo-bucket2/orc/2013-10-05-custdata",
         "mandatory":true,
         "meta":{
            "content_length":99
         }
      }
```

```
   ]
}
```

O exemplo a seguir usa um manifesto nomeado `cust.manifest`.

```
copy customer
from 's3://amzn-s3-demo-bucket/cust.manifest'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
format as orc
manifest;
```

Você pode usar um manifesto para carregar vários arquivos de buckets diferentes ou arquivos que não compartilham o mesmo prefixo. O exemplo a seguir mostra o JSON para carregar dados com arquivos cujos nomes começam com uma data e hora.

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket/2013-10-04-custdata.txt","mandatory":true},
    {"url":"s3://amzn-s3-demo-bucket/2013-10-05-custdata.txt","mandatory":true},
    {"url":"s3://amzn-s3-demo-bucket/2013-10-06-custdata.txt","mandatory":true},
    {"url":"s3://amzn-s3-demo-bucket/2013-10-07-custdata.txt","mandatory":true}
  ]
}
```

O manifesto pode listar arquivos em buckets diferentes, desde que os buckets estejam na mesma região da AWS do cluster.

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket1/custdata1.txt","mandatory":false},
    {"url":"s3://amzn-s3-demo-bucket2/custdata1.txt","mandatory":false},
    {"url":"s3://amzn-s3-demo-bucket2/custdata2.txt","mandatory":false}
  ]
}
```

Carregar LISTING de um arquivo delimitado por barras (delimitador padrão)

O exemplo a seguir é um caso muito simples em que nenhuma opção é especificada e o arquivo de entrada contém o delimitador padrão, um caractere de barra ('|').

```
copy listing
```

```
from 's3://amzn-s3-demo-bucket/data/listings_pipe.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Carregar LISTING usando dados de colunas no formato Parquet

O exemplo a seguir carrega dados de uma pasta no Amazon S3, chamada Parquet.

```
copy listing
from 's3://amzn-s3-demo-bucket/data/listings/parquet/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
format as parquet;
```

Carregar LISTING usando dados de colunas no formato ORC

O exemplo a seguir carrega dados de uma pasta no Amazon S3, chamada `orc`.

```
copy listing
from 's3://amzn-s3-demo-bucket/data/listings/orc/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
format as orc;
```

Carregar EVENT com opções

O exemplo a seguir carrega dados delimitados por barra na tabela EVENT e aplica as seguintes regras:

- Se forem usados para todas as strings de caracteres, os pares de aspas serão removidos.
- As strings vazias e as que contêm espaços em branco serão carregadas como valores NULL.
- A carga falha se mais de 5 erros forem retornados.
- Os valores de data e hora devem respeitar o formato especificado; por exemplo, uma data e hora válida é `2008-09-26 05:43:12`.

```
copy event
from 's3://amzn-s3-demo-bucket/data/allevents_pipe.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
removequotes
emptyasnull
blanksasnull
maxerror 5
delimiter '|'
```

```
timeformat 'YYYY-MM-DD HH:MI:SS';
```

Carregar VENUE de um arquivo de dados de largura fixa

```
copy venue
from 's3://amzn-s3-demo-bucket/data/venue_fw.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
fixedwidth 'venueid:3,venuename:25,venuecity:12,venuestate:2,venueseats:6';
```

O exemplo anterior pressupõe um arquivo de dados formatado da mesma maneira que os dados de exemplo mostrados. No exemplo a seguir, os espaços funcionam como espaços reservados, de maneira que todas as colunas tenham a mesma largura conforme observado na especificação:

```
1  Toyota Park              Bridgeview  IL0
2  Columbus Crew Stadium    Columbus    OH0
3  RFK Stadium              Washington  DC0
4  CommunityAmerica BallparkKansas City KS0
5  Gillette Stadium         Foxborough  MA68756
```

Carregar CATEGORY de um arquivo CSV

Suponhamos que você queira carregar CATEGORY com os valores mostrados na tabela a seguir.

catid	catgroup	catname	catdesc
12	Shows	Musicais	Casa de espetáculo
13	Shows	Peças	Todos os teatros "não musicais"
14	Shows	Opera	Todas as óperas, leves e óperas "rock"
15	Concertos	Clássicos	Todas as sinfonias, os concertos e os corais

O exemplo a seguir mostra o conteúdo de um arquivo de texto com os valores de campo separados por vírgulas.

```
12,Shows,Musicals,Musical theatre
```

```
13,Shows,Plays,All "non-musical" theatre
14,Shows,Opera,All opera, light, and "rock" opera
15,Concerts,Classical,All symphony, concerto, and choir concerts
```

Se você carregar o arquivo usando o parâmetro DELIMITER para especificar uma entrada delimitada por vírgulas, o comando COPY falha porque alguns campos de entrada contêm vírgulas. Você pode evitar esse problema usando o parâmetro CSV e colocando os campos que contenham vírgulas entre aspas. Se as aspas forem exibidas dentro de uma string com aspas, você precisará escapá-las dobrando as aspas. Como as aspas padrão são duplas, você precisa escapar todas as aspas duplas com aspas adicionais. O novo arquivo de entrada é algo assim.

```
12,Shows,Musicals,Musical theatre
13,Shows,Plays,"All ""non-musical"" theatre"
14,Shows,Opera,"All opera, light, and ""rock"" opera"
15,Concerts,Classical,"All symphony, concerto, and choir concerts"
```

Supondo que o nome de arquivo seja `category_csv.txt`, você pode carregar o arquivo usando o seguinte comando COPY:

```
copy category
from 's3://amzn-s3-demo-bucket/data/category_csv.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
csv;
```

Como alternativa, para evitar a necessidade de escapar as aspas duplas na entrada, você pode especificar aspas diferentes usando o parâmetro QUOTE AS. Por exemplo, a versão a seguir de `category_csv.txt` usa '%' como aspas.

```
12,Shows,Musicals,Musical theatre
13,Shows,Plays,%All "non-musical" theatre%
14,Shows,Opera,%All opera, light, and "rock" opera%
15,Concerts,Classical,%All symphony, concerto, and choir concerts%
```

O seguinte comando COPY usa QUOTE AS para carregar `category_csv.txt`:

```
copy category
from 's3://amzn-s3-demo-bucket/data/category_csv.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
csv quote as '%';
```

Carregar VENUE com valores explícitos para uma coluna IDENTITY

O exemplo a seguir pressupõe que, quando a tabela VENUE foi criada, pelo menos uma coluna (como a coluna `venueid`) foi especificada para ser uma coluna IDENTITY. Esse comando substitui o comportamento de IDENTITY padrão de valores gerados automaticamente para uma coluna IDENTITY e, em vez disso, carrega os valores explícitos do arquivo venue.txt. O Amazon Redshift não verifica se valores de IDENTITY duplicados são carregados na tabela ao usar a opção EXLICIT_IDS.

```
copy venue
from 's3://amzn-s3-demo-bucket/data/venue.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
explicit_ids;
```

Carregar TIME de um arquivo GZIP delimitado por barras

O seguinte exemplo carrega a tabela TIME de um arquivo GZIP delimitado por barras:

```
copy time
from 's3://amzn-s3-demo-bucket/data/timerows.gz'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
gzip
delimiter '|';
```

Carregar um time stamp ou date stamp

O exemplo a seguir carrega dados com uma data e hora formatada.

> **Note**
>
> O TIMEFORMAT de `HH:MI:SS` também pode dar suporte a segundos fracionários além do `SS` para um nível detalhado de microssegundos. O arquivo `time.txt` usado nesse exemplo contém uma linha `2009-01-12 14:15:57.119568`.

```
copy timestamp1
from 's3://amzn-s3-demo-bucket/data/time.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
timeformat 'YYYY-MM-DD HH:MI:SS';
```

O resultado dessa cópia é o seguinte:

```
select * from timestamp1;
c1
----------------------------
2009-01-12 14:15:57.119568
(1 row)
```

Carregar dados de um arquivo com valores padrão

O exemplo a seguir usa uma variação da tabela VENUE no banco de dados TICKIT. Leve em consideração uma tabela VENUE_NEW definida com o seguinte comando:

```
create table venue_new(
venueid smallint not null,
venuename varchar(100) not null,
venuecity varchar(30),
venuestate char(2),
venueseats integer not null default '1000');
```

Leve em consideração um arquivo de dados venue_noseats.txt que não contenha valores para a coluna VENUESEATS, conforme mostrado no seguinte exemplo:

```
1|Toyota Park|Bridgeview|IL|
2|Columbus Crew Stadium|Columbus|OH|
3|RFK Stadium|Washington|DC|
4|CommunityAmerica Ballpark|Kansas City|KS|
5|Gillette Stadium|Foxborough|MA|
6|New York Giants Stadium|East Rutherford|NJ|
7|BMO Field|Toronto|ON|
8|The Home Depot Center|Carson|CA|
9|Dick's Sporting Goods Park|Commerce City|CO|
10|Pizza Hut Park|Frisco|TX|
```

A seguinte instrução COPY carregará com êxito a tabela do arquivo e aplicará o valor DEFAULT ('1000') à coluna omitida:

```
copy venue_new(venueid, venuename, venuecity, venuestate)
from 's3://amzn-s3-demo-bucket/data/venue_noseats.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter '|';
```

Agora veja a tabela carregada:

```
select * from venue_new order by venueid;
venueid |         venuename          |    venuecity    | venuestate | venueseats
---------+----------------------------+-----------------+------------+------------
1 | Toyota Park                | Bridgeview      | IL         |       1000
2 | Columbus Crew Stadium      | Columbus        | OH         |       1000
3 | RFK Stadium                | Washington      | DC         |       1000
4 | CommunityAmerica Ballpark  | Kansas City     | KS         |       1000
5 | Gillette Stadium           | Foxborough      | MA         |       1000
6 | New York Giants Stadium    | East Rutherford | NJ         |       1000
7 | BMO Field                  | Toronto         | ON         |       1000
8 | The Home Depot Center      | Carson          | CA         |       1000
9 | Dick's Sporting Goods Park | Commerce City   | CO         |       1000
10 | Pizza Hut Park             | Frisco          | TX         |       1000
(10 rows)
```

Para o seguinte exemplo, além de pressupor que nenhum dado de VENUESEATS esteja incluído no arquivo, também pressuponha que não haja dados VENUENAME incluídos:

```
1||Bridgeview|IL|
2||Columbus|OH|
3||Washington|DC|
4||Kansas City|KS|
5||Foxborough|MA|
6||East Rutherford|NJ|
7||Toronto|ON|
8||Carson|CA|
9||Commerce City|CO|
10||Frisco|TX|
```

Usando a mesma definição de tabela, a seguinte instrução COPY falha porque nenhum valor DEFAULT foi especificado para VENUENAME, e VENUENAME é uma coluna NOT NULL:

```
copy venue(venueid, venuecity, venuestate)
from 's3://amzn-s3-demo-bucket/data/venue_pipe.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter '|';
```

Agora leve em consideração uma variação da tabela VENUE que usa uma coluna IDENTITY:

```
create table venue_identity(
```

```
venueid int identity(1,1),
venuename varchar(100) not null,
venuecity varchar(30),
venuestate char(2),
venueseats integer not null default '1000');
```

Assim como acontece com o exemplo anterior, pressuponha que a coluna VENUESEATS não tenha valores correspondentes no arquivo de origem. A instrução COPY a seguir carrega a tabela com êxito, inclusive os valores de dados IDENTITY predefinidos em vez de gerar automaticamente esses valores:

```
copy venue(venueid, venuename, venuecity, venuestate)
from 's3://amzn-s3-demo-bucket/data/venue_pipe.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter '|' explicit_ids;
```

Ocorre uma falha nessa instrução porque não inclui a coluna IDENTITY (VENUEID não é encontrado na lista de colunas), além de incluir um parâmetro EXPLICIT_IDS:

```
copy venue(venuename, venuecity, venuestate)
from 's3://amzn-s3-demo-bucket/data/venue_pipe.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter '|' explicit_ids;
```

Ocorre uma falha nessa instrução porque não inclui um parâmetro EXPLICIT_IDS:

```
copy venue(venueid, venuename, venuecity, venuestate)
from 's3://amzn-s3-demo-bucket/data/venue_pipe.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter '|';
```

Dados de COPY com a opção ESCAPE

O exemplo a seguir mostra como carregar caracteres correspondentes ao caractere delimitador (nesse caso, o caractere de barra). No arquivo de entrada, verifique se todos os caracteres de barra (|) que você deseja carregar sejam escapados com o caractere de barra invertida (\). Em seguida, carregue o arquivo com o parâmetro ESCAPE.

```
$ more redshiftinfo.txt
1|public\|event\|dwuser
```

```
2|public\|sales\|dwuser

create table redshiftinfo(infoid int,tableinfo varchar(50));

copy redshiftinfo from 's3://amzn-s3-demo-bucket/data/redshiftinfo.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter '|' escape;

select * from redshiftinfo order by 1;
infoid |       tableinfo
-------+--------------------
1      | public|event|dwuser
2      | public|sales|dwuser
(2 rows)
```

Sem o parâmetro ESCAPE, esse comando COPY falha com um erro `Extra column(s) found`.

> **Important**
>
> Se carregar os dados usando um COPY com o parâmetro ESCAPE, você também deverá especificar o parâmetro ESCAPE com o comando UNLOAD para gerar o arquivo de saída recíproco. Da mesma maneira, se usar UNLOAD com o parâmetro ESCAPE, você precisa usar ESCAPE quando usar o comando COPY nos mesmos dados.

Copiar de exemplos JSON

Nos exemplos a seguir, você carrega a tabela CATEGORY com os dados a seguir.

CATID	CATGROUP	CATNAME	CATDESC
1	Esportes	MLB	Major League Baseball
2	Esportes	NHL	National Hockey League
3	Esportes	NFL	National Football League
4	Esportes	NBA	National Basketball Association
5	Concertos	Clássicos	Todas as sinfonias, os concertos e os corais

Tópicos

- Carregar de dados JSON usando a opção "auto"
- Carregar de dados JSON usando a opção "auto ignorecase"
- Carregar de dados JSON usando um arquivo JSONPaths
- Carregar de matrizes JSON usando um arquivo JSONPaths

Carregar de dados JSON usando a opção "auto"

Para carregar de dados JSON usando o argumento `'auto'`, os dados JSON devem consistir em um conjunto de objetos. Os nomes de chave devem corresponder aos nomes de coluna, mas, nesse caso, a ordem não importa. A seguir, o conteúdo de um arquivo chamado `category_object_auto.json`.

```
{
    "catdesc": "Major League Baseball",
    "catid": 1,
    "catgroup": "Sports",
    "catname": "MLB"
}
{
    "catgroup": "Sports",
    "catid": 2,
    "catname": "NHL",
    "catdesc": "National Hockey League"
}
{
    "catid": 3,
    "catname": "NFL",
    "catgroup": "Sports",
    "catdesc": "National Football League"
}
{
    "bogus": "Bogus Sports LLC",
    "catid": 4,
    "catgroup": "Sports",
    "catname": "NBA",
    "catdesc": "National Basketball Association"
}
{
    "catid": 5,
    "catgroup": "Shows",
```

```
    "catname": "Musicals",
    "catdesc": "All symphony, concerto, and choir concerts"
}
```

Para carregar do arquivo de dados JSON no exemplo anterior, execute o comando COPY a seguir.

```
copy category
from 's3://amzn-s3-demo-bucket/category_object_auto.json'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
json 'auto';
```

Carregar de dados JSON usando a opção "auto ignorecase"

Para carregar de dados JSON usando o argumento `'auto ignorecase'`, os dados JSON devem consistir em um conjunto de objetos. As maiúsculas e minúsculas dos nomes de chave não precisam corresponder aos nomes de coluna e a ordem não importa. A seguir, o conteúdo de um arquivo chamado `category_object_auto-ignorecase.json`.

```
{
    "CatDesc": "Major League Baseball",
    "CatID": 1,
    "CatGroup": "Sports",
    "CatName": "MLB"
}
{
    "CatGroup": "Sports",
    "CatID": 2,
    "CatName": "NHL",
    "CatDesc": "National Hockey League"
}
{
    "CatID": 3,
    "CatName": "NFL",
    "CatGroup": "Sports",
    "CatDesc": "National Football League"
}
{
    "bogus": "Bogus Sports LLC",
    "CatID": 4,
    "CatGroup": "Sports",
    "CatName": "NBA",
    "CatDesc": "National Basketball Association"
```

```
}
{
    "CatID": 5,
    "CatGroup": "Shows",
    "CatName": "Musicals",
    "CatDesc": "All symphony, concerto, and choir concerts"
}
```

Para carregar do arquivo de dados JSON no exemplo anterior, execute o comando COPY a seguir.

```
copy category
from 's3://amzn-s3-demo-bucket/category_object_auto ignorecase.json'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
json 'auto ignorecase';
```

Carregar de dados JSON usando um arquivo JSONPaths

Se os objetos de dados JSON não corresponderem diretamente aos nomes de coluna, você poderá usar um arquivo JSONPaths a fim de mapear os elementos JSON para colunas. Além disso, a ordem não importa nos dados de origem JSON, mas a ordem das expressões do arquivo JSONPaths deve corresponder à ordem da coluna. Suponha que você tenha o seguinte arquivo de dados, chamado `category_object_paths.json`.

```
{
    "one": 1,
    "two": "Sports",
    "three": "MLB",
    "four": "Major League Baseball"
}
{
    "three": "NHL",
    "four": "National Hockey League",
    "one": 2,
    "two": "Sports"
}
{
    "two": "Sports",
    "three": "NFL",
    "one": 3,
    "four": "National Football League"
}
{
```

```
    "one": 4,
    "two": "Sports",
    "three": "NBA",
    "four": "National Basketball Association"
}
{
    "one": 6,
    "two": "Shows",
    "three": "Musicals",
    "four": "All symphony, concerto, and choir concerts"
}
```

O arquivo JSONPaths a seguir, chamado `category_jsonpath.json`, mapeia os dados de origem para as colunas de tabela.

```
{
    "jsonpaths": [
        "$['one']",
        "$['two']",
        "$['three']",
        "$['four']"
    ]
}
```

Para carregar do arquivo de dados JSON no exemplo anterior, execute o comando COPY a seguir.

```
copy category
from 's3://amzn-s3-demo-bucket/category_object_paths.json'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
json 's3://amzn-s3-demo-bucket/category_jsonpath.json';
```

Carregar de matrizes JSON usando um arquivo JSONPaths

Para carregar de dados JSON que consistem em um conjunto de matrizes, você deve usar um arquivo JSONPaths a fim de mapear os elementos de matriz para as colunas. Suponha que você tenha o seguinte arquivo de dados, chamado `category_array_data.json`.

```
[1,"Sports","MLB","Major League Baseball"]
[2,"Sports","NHL","National Hockey League"]
[3,"Sports","NFL","National Football League"]
[4,"Sports","NBA","National Basketball Association"]
```

```
[5,"Concerts","Classical","All symphony, concerto, and choir concerts"]
```

O arquivo JSONPaths a seguir, chamado `category_array_jsonpath.json`, mapeia os dados de origem para as colunas de tabela.

```
{
    "jsonpaths": [
        "$[0]",
        "$[1]",
        "$[2]",
        "$[3]"
    ]
}
```

Para carregar do arquivo de dados JSON no exemplo anterior, execute o comando COPY a seguir.

```
copy category
from 's3://amzn-s3-demo-bucket/category_array_data.json'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
json 's3://amzn-s3-demo-bucket/category_array_jsonpath.json';
```

Copiar de exemplos Avro

Nos exemplos a seguir, você carrega a tabela CATEGORY com os dados a seguir.

CATID	CATGROUP	CATNAME	CATDESC
1	Esportes	MLB	Major League Baseball
2	Esportes	NHL	National Hockey League
3	Esportes	NFL	National Football League
4	Esportes	NBA	National Basketball Association
5	Concertos	Clássicos	Todas as sinfonias, os concertos e os corais

Tópicos

- Carregar de dados Avro usando a opção "auto"

- Carregar de dados Avro usando a opção "auto ignorecase"
- Carregar de dados Avro usando um arquivo JSONPaths

Carregar de dados Avro usando a opção "auto"

Para carregar de dados Avro usando o argumento `'auto'`, os nomes de campo no esquema Avro devem corresponder aos nomes de coluna. Durante o uso do argumento `'auto'`, a ordem não importa. A seguir, o esquema de um arquivo chamado `category_auto.avro`.

```
{
    "name": "category",
    "type": "record",
    "fields": [
        {"name": "catid", "type": "int"},
        {"name": "catdesc", "type": "string"},
        {"name": "catname", "type": "string"},
        {"name": "catgroup", "type": "string"},
}
```

Como estão em formato binário, os dados em um arquivo Avro não são legíveis. A seguir, uma representação JSON dos dados no arquivo `category_auto.avro`.

```
{
   "catid": 1,
   "catdesc": "Major League Baseball",
   "catname": "MLB",
   "catgroup": "Sports"
}
{
   "catid": 2,
   "catdesc": "National Hockey League",
   "catname": "NHL",
   "catgroup": "Sports"
}
{
   "catid": 3,
   "catdesc": "National Basketball Association",
   "catname": "NBA",
   "catgroup": "Sports"
}
{
```

```
   "catid": 4,
   "catdesc": "All symphony, concerto, and choir concerts",
   "catname": "Classical",
   "catgroup": "Concerts"
}
```

Para carregar do arquivo de dados Avro no exemplo anterior, execute o comando COPY a seguir.

```
copy category
from 's3://amzn-s3-demo-bucket/category_auto.avro'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
format as avro 'auto';
```

Carregar de dados Avro usando a opção "auto ignorecase"

Para carregar de dados Avro usando o argumento `'auto ignorecase'`, a maiúscula e minúscula nos nomes de campo no esquema Avro devem corresponder aos nomes de coluna. Durante o uso do argumento `'auto ignorecase'`, a ordem não importa. A seguir, o esquema de um arquivo chamado `category_auto-ignorecase.avro`.

```
{
    "name": "category",
    "type": "record",
    "fields": [
        {"name": "CatID", "type": "int"},
        {"name": "CatDesc", "type": "string"},
        {"name": "CatName", "type": "string"},
        {"name": "CatGroup", "type": "string"},
}
```

Como estão em formato binário, os dados em um arquivo Avro não são legíveis. A seguir, uma representação JSON dos dados no arquivo `category_auto-ignorecase.avro`.

```
{
   "CatID": 1,
   "CatDesc": "Major League Baseball",
   "CatName": "MLB",
   "CatGroup": "Sports"
}
{
   "CatID": 2,
   "CatDesc": "National Hockey League",
```

```
   "CatName": "NHL",
   "CatGroup": "Sports"
}
{
   "CatID": 3,
   "CatDesc": "National Basketball Association",
   "CatName": "NBA",
   "CatGroup": "Sports"
}
{
   "CatID": 4,
   "CatDesc": "All symphony, concerto, and choir concerts",
   "CatName": "Classical",
   "CatGroup": "Concerts"
}
```

Para carregar do arquivo de dados Avro no exemplo anterior, execute o comando COPY a seguir.

```
copy category
from 's3://amzn-s3-demo-bucket/category_auto-ignorecase.avro'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
format as avro 'auto ignorecase';
```

Carregar de dados Avro usando um arquivo JSONPaths

Se os nomes de campo no esquema Avro não corresponderem diretamente aos nomes de coluna, você poderá usar um arquivo JSONPaths a fim de mapear os elementos de esquema para colunas. A ordem das expressões do arquivo JSONPaths deve corresponder à ordem das colunas.

Suponha que você tenha um arquivo de dados chamado `category_paths.avro` contendo os mesmos dados do exemplo anterior, mas com o esquema a seguir.

```
{
    "name": "category",
    "type": "record",
    "fields": [
        {"name": "id", "type": "int"},
        {"name": "desc", "type": "string"},
        {"name": "name", "type": "string"},
        {"name": "group", "type": "string"},
        {"name": "region", "type": "string"}
     ]
```

```
}
```

O arquivo JSONPaths a seguir, chamado `category_path.avropath`, mapeia os dados de origem para as colunas de tabela.

```
{
    "jsonpaths": [
        "$['id']",
        "$['group']",
        "$['name']",
        "$['desc']"
    ]
}
```

Para carregar do arquivo de dados Avro no exemplo anterior, execute o comando COPY a seguir.

```
copy category
from 's3://amzn-s3-demo-bucket/category_object_paths.avro'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
format avro 's3://amzn-s3-demo-bucket/category_path.avropath ';
```

Preparar arquivos para COPY com a opção ESCAPE

O exemplo a seguir descreve como convém preparar dados para "escapar" caracteres de nova linha antes de importar os dados para uma tabela do Amazon Redshift usando o comando COPY com o parâmetro ESCAPE. Sem preparar os dados para delimitar os caracteres de nova linha, o Amazon Redshift retorna erros de carga quando você executar o comando COPY, porque o caractere de nova linha normalmente é usado como um separador de registros.

Por exemplo, leve em consideração um arquivo ou uma coluna em uma tabela externa que você deseja copiar para uma tabela do Amazon Redshift. Se o arquivo ou a coluna apresentar conteúdo formatado em XML ou dados semelhantes, você precisa verificar se todos os caracteres de nova linha (\n) que fazem parte do conteúdo são escapados com o caractere de barra invertida (\).

Um arquivo ou tabela contendo caracteres de nova linha incorporados fornece um padrão relativamente fácil de combinar. Cada caractere de nova linha interno quase sempre vem depois de um caractere > com alguns caracteres de espaço em branco em potencial (`' '` ou tabulação) entre eles, como você pode ver no exemplo a seguir de um arquivo de texto chamado `nlTest1.txt`.

```
$ cat nlTest1.txt
```

```
<xml start>
<newline characters provide>
<line breaks at the end of each>
<line in content>
</xml>|1000
<xml>
</xml>|2000
```

Com o exemplo a seguir, você pode executar um utilitário de processamento de texto para pré-processar o arquivo de origem e inserir os caracteres de escape quando necessário. (O caractere | deve ser usado como delimitador para separar dados de coluna quando copiados para uma tabela do Amazon Redshift.)

```
$ sed -e ':a;N;$!ba;s/>[[:space:]]*\n/>\\\n/g' nlTest1.txt > nlTest2.txt
```

Da mesma maneira, você pode usar Perl para realizar uma operação semelhante:

```
cat nlTest1.txt | perl -p -e 's/>\s*\n/>\\\n/g' > nlTest2.txt
```

Para permitir o carregamento dos dados do arquivo `nlTest2.txt` no Amazon Redshift, este exemplo cria uma tabela de duas colunas no Amazon Redshift. A primeira coluna c1 é uma coluna de caracteres que mantém o conteúdo formatado do arquivo `nlTest2.txt`. A segunda coluna c2 mantém valores inteiros carregados do mesmo arquivo.

Depois de executar o comando `sed`, você poderá carregar corretamente dados do arquivo `nlTest2.txt` para uma tabela do Amazon Redshift usando o parâmetro ESCAPE.

> **Note**
>
> Quando você inclui o parâmetro ESCAPE com o comando COPY, ele escapa vários caracteres especiais, dentre os quais estão caractere de barra invertida (inclusive nova linha).

```
copy t2 from 's3://amzn-s3-demo-bucket/data/nlTest2.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
escape
delimiter as '|';

select * from t2 order by 2;
```

```
c1           |  c2
-------------+------
<xml start>
<newline characters provide>
<line breaks at the end of each>
<line in content>
</xml>
| 1000
<xml>
</xml>       | 2000
(2 rows)
```

Você pode preparar arquivos de dados exportados de bancos de dados externos de maneira semelhante. Por exemplo, com um banco de dados Oracle, você pode usar a função REPLACE em cada coluna afetada em uma tabela que deseja copiar para o Amazon Redshift.

```
SELECT c1, REPLACE(c2, \n',\\n' ) as c2 from my_table_with_xml
```

Além disso, muitas ferramentas Export e Extract, Transform, Load (ETL – Exportação e extração, transformação, carga) de banco de dados que processam sempre muitos dados oferecem opções para especificar caracteres de escape e delimitadores.

Carregar um shapefile no Amazon Redshift

Os exemplos a seguir demonstram como carregar um shapefile da Esri usando COPY. Para obter mais informações o carregamento de shapefiles, consulte Carregar um shapefile no Amazon Redshift.

Carregar um shapefile

As etapas a seguir mostram como ingerir dados do OpenStreetMap do Amazon S3 usando o comando COPY. Este exemplo pressupõe que o arquivo shapefile da Noruega do site de download de Geofabrik foi carregado para um bucket privado do Amazon S3 em sua região da AWS. Os arquivos `.shp`, `.shx` e `.dbf` devem compartilhar o mesmo prefixo do Amazon S3 e nome de arquivo.

Ingestão de dados sem simplificação

Os comandos a seguir criam tabelas e ingerem dados que podem caber no tamanho máximo da geometria sem qualquer simplificação. Abra o `gis_osm_natural_free_1.shp` em seu

software GIS preferido e inspecione as colunas nesta camada. Por padrão, as colunas IDENTITY ou GEOMETRY são as primeiras. Quando uma coluna GEOMETRY é a primeira, você pode criar a tabela como mostrado a seguir.

```
CREATE TABLE norway_natural (
   wkb_geometry GEOMETRY,
   osm_id BIGINT,
   code INT,
   fclass VARCHAR,
   name VARCHAR);
```

Ou, quando uma coluna IDENTITY é a primeira, você pode criar a tabela como mostrado a seguir.

```
CREATE TABLE norway_natural_with_id (
   fid INT IDENTITY(1,1),
   wkb_geometry GEOMETRY,
   osm_id BIGINT,
   code INT,
   fclass VARCHAR,
   name VARCHAR);
```

Agora você pode ingerir os dados usando COPY.

```
COPY norway_natural FROM 's3://bucket_name/shapefiles/norway/
gis_osm_natural_free_1.shp'
FORMAT SHAPEFILE
CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789012:role/MyRoleName';
INFO: Load into table 'norway_natural' completed, 83891 record(s) loaded successfully
```

Ou você pode ingerir os dados como mostrado a seguir.

```
COPY norway_natural_with_id FROM 's3://bucket_name/shapefiles/norway/
gis_osm_natural_free_1.shp'
FORMAT SHAPEFILE
CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789012:role/MyRoleName';
INFO: Load into table 'norway_natural_with_id' completed, 83891 record(s) loaded
 successfully.
```

Ingestão de dados com simplificação

Os comandos a seguir criam uma tabela e tentam ingerir dados que não podem caber no tamanho máximo da geometria sem qualquer simplificação. Inspecione o shapefile `gis_osm_water_a_free_1.shp` e crie a tabela apropriada como mostrado a seguir.

```
CREATE TABLE norway_water (
   wkb_geometry GEOMETRY,
   osm_id BIGINT,
   code INT,
   fclass VARCHAR,
   name VARCHAR);
```

Quando o comando COPY é executado, isso resulta em um erro.

```
COPY norway_water FROM 's3://bucket_name/shapefiles/norway/gis_osm_water_a_free_1.shp'
FORMAT SHAPEFILE
CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789012:role/MyRoleName';
ERROR:  Load into table 'norway_water' failed.  Check 'stl_load_errors' system table
 for details.
```

Consultar `STL_LOAD_ERRORS` mostra que a geometria é muito grande.

```
SELECT line_number, btrim(colname), btrim(err_reason) FROM stl_load_errors WHERE query
 = pg_last_copy_id();
 line_number |    btrim     |                                  btrim
-------------+--------------
+-----------------------------------------------------------------------
     1184705 | wkb_geometry | Geometry size: 1513736 is larger than maximum supported
 size: 1048447
```

Para superar isso, o parâmetro `SIMPLIFY AUTO` é adicionado ao comando COPY para simplificar geometrias.

```
COPY norway_water FROM 's3://bucket_name/shapefiles/norway/gis_osm_water_a_free_1.shp'
FORMAT SHAPEFILE
SIMPLIFY AUTO
CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789012:role/MyRoleName';

INFO:  Load into table 'norway_water' completed, 1989196 record(s) loaded successfully.
```

Para exibir as linhas e geometrias que foram simplificadas, consulte SVL_SPATIAL_SIMPLIFY.

```
SELECT * FROM svl_spatial_simplify WHERE query = pg_last_copy_id();
 query | line_number | maximum_tolerance | initial_size | simplified | final_size |
 final_tolerance
-------+-------------+-------------------+--------------+------------+------------
+----------------------
    20 |     1184704 |                -1 |      1513736 | t          |    1008808 |
 1.276386653895e-05
    20 |     1664115 |                -1 |      1233456 | t          |    1023584 |
 6.11707814796635e-06
```

Utilizar SIMPLIFY AUTO max_tolerance com a tolerância menor do que as calculadas automaticamente provavelmente resulta em um erro de ingestão. Nesse caso, use MAXERROR para ignorar erros.

```
COPY norway_water FROM 's3://bucket_name/shapefiles/norway/gis_osm_water_a_free_1.shp'
FORMAT SHAPEFILE
SIMPLIFY AUTO 1.1E-05
MAXERROR 2
CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789012:role/MyRoleName';

INFO:  Load into table 'norway_water' completed, 1989195 record(s) loaded successfully.
INFO:  Load into table 'norway_water' completed, 1 record(s) could not be loaded.
 Check 'stl_load_errors' system table for details.
```

Consulte SVL_SPATIAL_SIMPLIFY novamente para identificar o registro que COPY não conseguiu carregar.

```
SELECT * FROM svl_spatial_simplify WHERE query = pg_last_copy_id();
 query | line_number | maximum_tolerance | initial_size | simplified | final_size |
 final_tolerance
-------+-------------+-------------------+--------------+------------+------------
+-----------------
    29 |     1184704 |           1.1e-05 |      1513736 | f          |          0 |
            0
    29 |     1664115 |           1.1e-05 |      1233456 | t          |     794432 |
     1.1e-05
```

Neste exemplo, o primeiro registro não conseguiu caber, então o `simplified` está mostrando "false". O segundo registro foi carregado dentro da tolerância dada. No entanto, o tamanho final é maior do que usar a tolerância calculada automaticamente sem especificar a tolerância máxima.

Carregar a partir de um shapefile compactado

O Amazon Redshift COPY oferece suporte à ingestão de dados de um shapefile compactado. Todos os componentes shapefile devem ter o mesmo prefixo do Amazon S3 e o mesmo sufixo de compactação. Como exemplo, suponha que você deseja carregar os dados do exemplo anterior. Neste caso, os arquivos `gis_osm_water_a_free_1.shp.gz`, `gis_osm_water_a_free_1.dbf.gz`, e `gis_osm_water_a_free_1.shx.gz` devem compartilhar o mesmo diretório do Amazon S3. O comando COPY requer a opção GZIP e a cláusula FROM deve especificar o arquivo compactado correto, conforme mostrado a seguir.

```
COPY norway_natural FROM 's3://bucket_name/shapefiles/norway/compressed/
gis_osm_natural_free_1.shp.gz'
FORMAT SHAPEFILE
GZIP
CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789012:role/MyRoleName';
INFO:  Load into table 'norway_natural' completed, 83891 record(s) loaded successfully.
```

Carregar dados em uma tabela com uma ordem de coluna diferente

Se você tiver uma tabela que não tenha `GEOMETRY` como a primeira coluna, você pode usar o mapeamento de coluna para mapear colunas para a tabela de destino. Por exemplo, crie uma tabela com `osm_id` especificado como uma primeira coluna.

```
CREATE TABLE norway_natural_order (
   osm_id BIGINT,
   wkb_geometry GEOMETRY,
   code INT,
   fclass VARCHAR,
   name VARCHAR);
```

Em seguida, faça a ingestão de um shapefile usando o mapeamento de coluna.

```
COPY norway_natural_order(wkb_geometry, osm_id, code, fclass, name)
FROM 's3://bucket_name/shapefiles/norway/gis_osm_natural_free_1.shp'
FORMAT SHAPEFILE
```

```
CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789012:role/MyRoleName';
INFO:  Load into table 'norway_natural_order' completed, 83891 record(s) loaded
 successfully.
```

Carregar dados em uma tabela com uma coluna geography

Caso você tenha uma tabela com uma coluna GEOGRAPHY, ingira primeiro para uma coluna GEOMETRY e converta os objetos para objetos GEOGRAPHY. Por exemplo, depois de copiar seu arquivo de formato para uma coluna GEOMETRY, altere a tabela para adicionar uma coluna do tipo de dados GEOGRAPHY.

```
ALTER TABLE norway_natural ADD COLUMN wkb_geography GEOGRAPHY;
```

Então converta geometrias em geografias.

```
UPDATE norway_natural SET wkb_geography = wkb_geometry::geography;
```

Se preferir, você pode descartar a coluna GEOMETRY.

```
ALTER TABLE norway_natural DROP COLUMN wkb_geometry;
```

Comando COPY com a opção NOLOAD

Para validar os arquivos de dados antes de realmente carregar os dados, use a opção NOLOAD com o comando COPY. O Amazon Redshift analisa o arquivo de entrada e exibe todos os erros que ocorrem. O exemplo a seguir usa a opção NOLOAD e nenhuma linha é realmente carregada na tabela.

```
COPY public.zipcode1
FROM 's3://amzn-s3-demo-bucket/mydata/zipcode.csv'
DELIMITER ';'
IGNOREHEADER 1 REGION 'us-east-1'
NOLOAD
CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789012:role/myRedshiftRole';

Warnings:
Load into table 'zipcode1' completed, 0 record(s) loaded successfully.
```

Comando COPY com um delimitador multibyte e a opção ENCODING

O exemplo a seguir carrega LATIN1 de um arquivo do Amazon S3 que contém dados multibyte. O comando COPY especifica o delimitador `\302\246\303\254` em formato octal para separar os campos no arquivo de entrada que é codificado como ISO-8859-1. Para indicar o mesmo delimitador em UTF-8, especifique `DELIMITER '¦ì'`.

```
COPY latin1
FROM 's3://amzn-s3-demo-bucket/multibyte/myfile'
IAM_ROLE 'arn:aws:iam::123456789012:role/myRedshiftRole'
DELIMITER '\302\246\303\254'
ENCODING ISO88591
```

CREATE DATABASE

Cria um novo banco de dados.

Para criar um banco de dados, é necessário ser um superusuário ou ter o privilégio CREATEDB. Para criar um banco de dados associado a uma integração ETL zero, é necessário ser um superusuário ou ter os privilégios CREATEDB e CREATEUSER.

Não é possível executar CREATE DATABASE em um bloco de transação (BEGIN ... END). Para obter mais informações sobre transações, consulte Níveis de isolamento no Amazon Redshift.

Sintaxe

```
CREATE DATABASE database_name
[ { [
      FROM INTEGRATION '<integration_id>'[ DATABASE '<source_database>' ]
      [ SET ]
      [ ACCEPTINVCHARS [=] { TRUE | FALSE }]
      [ QUERY_ALL_STATES [=] { TRUE | FALSE }]
      [ REFRESH_INTERVAL <interval> ]
      [ TRUNCATECOLUMNS [=] { TRUE | FALSE } ]
      [ HISTORY_MODE [=] {TRUE | FALSE} ]
      [ AUTO_REMEDIATION [=] { TRUE | FALSE } ]
    ]
    [ WITH ]
    [ OWNER [=] db_owner ]
    [ CONNECTION LIMIT { limit | UNLIMITED } ]
    [ COLLATE { CASE_SENSITIVE | CS | CASE_INSENSITIVE | CI } ]
```

```
    [ ISOLATION LEVEL { SNAPSHOT | SERIALIZABLE } ]
  }
  | { FROM { { ARN '<arn>' } { WITH DATA CATALOG SCHEMA '<schema>' | WITH NO DATA
 CATALOG SCHEMA } } }
  | { IAM_ROLE  {default | 'SESSION' | 'arn:aws:iam::<account-id>:role/<role-name>' } }
  | { [ WITH PERMISSIONS ] FROM DATASHARE datashare_name OF [ ACCOUNT account_id ]
 NAMESPACE namespace_guid }
]
```

Parâmetros

database_name

Nome do novo banco de dados. Para obter mais informações sobre nomes válidos, consulte Nomes e identificadores.

FROM INTEGRATION '<integration_id>' [DATABASE '<source_database>']

Especifica se o banco de dados deve ser criado usando um identificador de integração ETL zero. É possível recuperar o `integration_id` da visualização do sistema SVV_INTEGRATION. Em integrações ETL zero do Aurora PostgreSQL, você também precisa especificar o nome `source_database`, que também pode ser recuperado em SVV_INTEGRATION.

Para ver um exemplo, consulte Criar bancos de dados para receber resultados de integrações ETL zero. Consulte mais informações sobre a criação de bancos de dados com integrações ETL zero em Criar bancos de dados de destino no Amazon Redshift no Guia de gerenciamento do Amazon Redshift.

SET

Palavra-chave opcional.

ACCEPTINVCHARS [=] { TRUE | FALSE }

A cláusula ACCEPTINVCHARS define se as tabelas de integração ETL zero devem continuar com a ingestão quando caracteres inválidos são detectados para o tipo de dados VARCHAR. Quando um caractere inválido é encontrado, ele é substituído por um caractere padrão `?`.

QUERY_ALL_STATES [=] { TRUE | FALSE }

A cláusula QUERY_ALL_STATES define se as tabelas de integração ETL zero podem ser consultadas em todos os estados (`Synced`, `Failed`, `ResyncRequired` e `ResyncInitiated`). Por padrão, uma tabela de integração ETL zero só pode ser consultada no estado `Synced`.

REFRESH_INTERVAL <intervalo>

A cláusula REFRESH_INTERVAL define o intervalo aproximado, em segundos, que o Amazon Redshift aguarda após a conclusão de um ciclo de atualização para iniciar o próximo. Cada ciclo atualiza os dados da origem da integração ETL zero para o banco de dados de destino, aplicando todas as alterações acumuladas desde o término do ciclo anterior. O valor 0 inicia o próximo ciclo assim que o anterior termina (replicação quase em tempo real). Um valor maior aumenta o intervalo entre os ciclos, reduzindo a sobrecarga das atualizações em troca de uma menor atualização imediata dos dados. O Amazon Redshift aguarda entre os ciclos somente quando a ingestão alcança o ritmo das alterações. Quando as alterações se acumulam mais rapidamente do que podem ser aplicadas, os ciclos são executados sem intervalo de espera.

O `interval` pode ser definido entre 0 e 432.000 segundos (5 dias) para integrações ETL zero cuja origem seja Aurora MySQL, Aurora PostgreSQL ou um dos mecanismos do RDS compatíveis. O valor padrão é 0. Para integrações ETL zero do Amazon DynamoDB, o `interval` pode ser definido entre 900 e 432.000 segundos (15 minutos a 5 dias), e o valor padrão é 900 segundos (15 minutos). O valor 0 não é compatível.

> **Note**
>
> Para integrações com alta intensidade de gravação, exceto as do DynamoDB, que geram um grande volume de alterações, defina REFRESH_INTERVAL como um pequeno valor diferente de zero (por exemplo, 60 a 120 segundos). Agrupar mais alterações em cada ciclo reduz a sobrecarga geral de computação da replicação.

TRUNCATECOLUMNS [=] { TRUE | FALSE }

A cláusula TRUNCATECOLUMNS define se as tabelas de Integração ETL zero devem continuar com a ingestão quando os valores dos atributos das colunas VARCHAR ou SUPER estão além do limite. Quando TRUE, os valores são truncados para caber na coluna e os valores dos atributos JSON em excesso são truncados para caber na coluna SUPER.

HISTORY_MODE [=] {TRUE | FALSE}

Uma cláusula que especifica se o Amazon Redshift vai definir o modo histórico para todas as novas tabelas no banco de dados especificado. Essa opção é aplicável somente para bancos de dados criados para integração ETL zero.

A cláusula HISTORY_MODE pode ser definida como TRUE ou FALSE. O padrão é FALSE. Consulte informações sobre HISTORY_MODE em History mode no Guia de gerenciamento do Amazon Redshift.

AUTO_REMEDIATION [=] { TRUE | FALSE }

Especifica se o Amazon Redshift deve sincronizar novamente de forma automática as tabelas afetadas por linhas duplicadas. Quando definido como TRUE, o Amazon Redshift marca as tabelas afetadas para ressincronização sem exigir intervenção manual. O padrão é FALSE.

É possível monitorar quais tabelas foram sinalizadas consultando SVV_INTEGRATION_TABLE_STATE. A configuração atual fica visível na coluna `auto_remediation` de SVV_INTEGRATION.

WITH

Palavra-chave opcional.

OWNER [=] db_owner

Especifica o nome de usuário do proprietário do banco de dados.

CONNECTION LIMIT { limite | UNLIMITED }

Número máximo de conexões de banco de dados que os usuários podem abrir simultaneamente. Não há aplicação de limite para superusuários. Use a palavra-chave UNLIMITED para permitir o número máximo de conexões simultâneas. Um limite no número de conexões para cada usuário também pode ser aplicável. Para obter mais informações, consulte CREATE USER. O valor padrão é UNLIMITED. Para visualizar as conexões atuais, consulte a exibição STV_SESSIONS do sistema.

> **Note**
>
> Se limites de usuário e de conexão de banco de dados forem aplicáveis, um slot de conexão não utilizado que esteja dentro de ambos os limites deve estar disponível quando um usuário tenta se conectar.

COLLATE { CASE_SENSITIVE | CS | CASE_INSENSITIVE | CI }

Uma cláusula que especifica se a pesquisa ou comparação de string distingue ou não maiúsculas e minúsculas. O padrão é distingue maiúsculas e minúsculas.

Não é possível usar COLLATE ao criar um banco de dados com base em uma unidade de compartilhamento de dados.

CASE_SENSITIVE e CS são intercambiáveis e geram os mesmos resultados. Da mesma forma, CASE_INSENSITIVE e CI são intercambiáveis e geram os mesmos resultados.

Para verificar o agrupamento atual de um banco de dados, use a função DB_COLLATION.

ISOLATION LEVEL { SNAPSHOT | SERIALIZABLE }

Uma cláusula que especifica o nível de isolamento usado quando são executadas consultas em um banco de dados. Para acessar mais informações sobre níveis de isolamento, consulte Níveis de isolamento no Amazon Redshift.

- Isolamento SNAPSHOT: fornece um nível de isolamento com proteção contra conflitos de atualização e exclusão. Esse é o padrão para um banco de dados criado em um cluster provisionado ou um namespace sem servidor.
- Isolamento SERIALIZABLE: fornece serialização total para transações simultâneas.

FROM ARN '<ARN>'

O ARN do banco de dados do AWS Glue a ser usado para criar o banco de dados.

{ WITH DATA CATALOG SCHEMA '<esquema>' | WITH NO DATA CATALOG SCHEMA }

> **Note**
>
> Esse parâmetro só será aplicável se o comando CREATE DATABASE também usar o parâmetro FROM ARN.

Especifica se o banco de dados deve ser criado usando um esquema para ajudar a acessar objetos no AWS Glue Data Catalog.

IAM_ROLE { default | 'SESSION' | 'arn:aws:iam::*<Conta da AWS-id>*:role/*<role-name>*' }

> **Note**
>
> Esse parâmetro só será aplicável se o comando CREATE DATABASE também usar o parâmetro FROM ARN.

Se você especificar um perfil do IAM associado ao cluster ao executar o comando CREATE DATABASE, o Amazon Redshift usará as credenciais do perfil ao executar consultas no banco de dados.

Especificar a palavra-chave `default` significa usar o perfil do IAM que está definido como padrão e associado ao cluster.

Use `'SESSION'` se você se conectar ao cluster do Amazon Redshift usando uma identidade federada e acesse as tabelas do esquema externo criado usando esse comando. Para obter um exemplo do uso de uma identidade federada, consulte Usar uma identidade federada para gerenciar o acesso do Amazon Redshift aos recursos locais e a tabelas externas do Amazon Redshift Spectrum, que explica como configurar uma identidade federada.

Use o nome do recurso da Amazon (ARN) de uma função do IAM que seu cluster usa para autenticação e autorização. No mínimo, a função do IAM deve ter permissão para executar uma operação LIST no bucket do Amazon S3 a ser acessado e uma operação GET nos objetos do Amazon S3 que constam no bucket. Para saber mais sobre como usar IAM_ROLE ao criar um banco de dados usando o AWS Glue Data Catalog para compartilhamentos de dados, consulte Trabalhar com unidades de compartilhamento de dados gerenciadas pelo Lake Formation como consumidor.

O exemplo a seguir mostra a sintaxe da string do parâmetro IAM_ROLE para um único ARN.

```
IAM_ROLE 'arn:aws:iam::<aws-account-id>:role/<role-name>'
```

Você pode encadear funções para que seu cluster possa assumir outra função do IAM, possivelmente pertencente a outra conta. Você pode encadear até 10 funções. Para obter mais informações, consulte Encadeamento de funções do IAM no Amazon Redshift Spectrum.

Anexe a essa função do IAM uma política de permissões do IAM semelhante à política descrita a seguir.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AccessSecret",
            "Effect": "Allow",
            "Action": [
```

```
                "secretsmanager:GetResourcePolicy",
                "secretsmanager:GetSecretValue",
                "secretsmanager:DescribeSecret",
                "secretsmanager:ListSecretVersionIds"
            ],
            "Resource": "arn:aws:secretsmanager:us-
west-2:123456789012:secret:my-rds-secret-VNenFy"
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword",
                "secretsmanager:ListSecrets"
            ],
            "Resource": "*"
        }
    ]
}
```

Para obter as etapas para criar uma função do IAM a ser usada com consulta federada, consulte Criar um segredo e uma função do IAM para usar consultas federadas.

> **Note**
>
> Não inclua espaços na lista de funções encadeadas.

O seguinte mostra a sintaxe do encadeamento de três funções.

```
IAM_ROLE 'arn:aws:iam::<aws-account-id>:role/<role-1-name>,arn:aws:iam::<aws-
account-id>:role/<role-2-name>,arn:aws:iam::<aws-account-id>:role/<role-3-name>'
```

Sintaxe para usar CREATE DATABASE com um datashare

A sintaxe a seguir descreve o comando CREATE DATABASE usado para criar bancos de dados a partir de uma unidade de compartilhamento de dados para compartilhar dados na mesma conta da AWS.

```
CREATE DATABASE database_name
```

```
[ [ WITH PERMISSIONS ] FROM DATASHARE datashare_name OF [ ACCOUNT account_id ]
 NAMESPACE namespace_guid
```

A sintaxe a seguir descreve o comando CREATE DATABASE usado para criar bancos de dados a partir de uma unidade de compartilhamento de dados para compartilhar dados entre contas da AWS.

```
CREATE DATABASE database_name
[ [ WITH PERMISSIONS ] FROM DATASHARE datashare_name OF ACCOUNT account_id
 NAMESPACE namespace_guid
```

Parâmetros para usar CREATE DATABASE com um datashare

FROM DATASHARE

Uma palavra-chave que indica onde o datashare está localizado.

datashare_name

O nome do datashare no qual o banco de dados do consumidor é criado.

WITH PERMISSIONS

Especifica que o banco de dados criado a partir da unidade de compartilhamento de dados requer permissões no nível de objeto para ter acesso a objetos do banco de dados individuais. Sem essa cláusula, os usuários ou as funções que receberem a permissão USAGE no banco de dados terão acesso automático a todos os objetos no banco de dados.

NAMESPACE namespace_guid

Valor que especifica o namespace de produtor ao qual a unidade de compartilhamento de dados pertence.

ACCOUNT account_id

Valor que especifica a conta de produtor à qual a unidade de compartilhamento de dados pertence.

Notas de uso de CREATE DATABASE para compartilhamento de dados

Como superusuário de banco de dados, ao usar CREATE DATABASE para criar bancos de dados a partir de unidades de compartilhamento de dados da conta da AWS, especifique a opção NAMESPACE. A opção ACCOUNT é opcional. Ao usar CREATE DATABASE para criar bancos de

dados de unidades de compartilhamento de dados entre contas da AWS, especifique os parâmetros ACCOUNT e NAMESPACE do produtor.

Você pode criar apenas um banco de dados de consumidor para uma unidade de compartilhamento de dados em um cluster de consumidores. Não é possível criar vários bancos de dados de consumidores referentes à mesma unidade de compartilhamento de dados.

CREATE DATABASE por meio do AWS Glue Data Catalog

Para criar um banco de dados usando um ARN do banco de dados do AWS Glue, especifique o ARN no comando CREATE DATABASE.

```
CREATE DATABASE sampledb FROM ARN <glue-database-arn> WITH NO DATA CATALOG SCHEMA;
```

Opcionalmente, você também pode fornecer um valor no parâmetro IAM_ROLE. Para obter mais informações sobre o parâmetro e os valores aceitos, consulte Parâmetros.

Veja a seguir exemplos que demonstram como criar um banco de dados com base em um ARN usando um perfil do IAM.

```
CREATE DATABASE sampledb FROM ARN <glue-database-arn> WITH NO DATA CATALOG SCHEMA
 IAM_ROLE <iam-role-arn>
```

```
CREATE DATABASE sampledb FROM ARN <glue-database-arn> WITH NO DATA CATALOG SCHEMA
 IAM_ROLE default;
```

Também é possível criar um banco de dados usando um DATA CATALOG SCHEMA.

```
CREATE DATABASE sampledb FROM ARN <glue-database-arn> WITH DATA CATALOG SCHEMA
 <sample_schema> IAM_ROLE default;
```

Criar bancos de dados para receber resultados de integrações ETL zero

Para criar um banco de dados usando uma identidade de integração ETL zero, especifique `integration_id` no comando CREATE DATABASE.

```
CREATE DATABASE destination_db_name FROM INTEGRATION 'integration_id';
```

Por exemplo, primeiro, recupere os IDs de integração de SVV_INTEGRATION:

```
SELECT integration_id FROM SVV_INTEGRATION;
```

Depois, use um dos IDs de integração recuperados para criar o banco de dados que recebe integrações ETL zero.

```
CREATE DATABASE sampledb FROM INTEGRATION 'a1b2c3d4-5678-90ab-cdef-EXAMPLE11111';
```

Quando o banco de dados de origem de integrações ETL zero for necessário, especifique, por exemplo:

```
CREATE DATABASE sampledb FROM INTEGRATION 'a1b2c3d4-5678-90ab-cdef-EXAMPLE11111'
 DATABASE sourcedb;
```

Você também pode definir um intervalo de atualização para o banco de dados. Por exemplo, para definir o intervalo de atualização como 7.200 segundos para dados de uma fonte de integração ETL zero:

```
CREATE DATABASE myacct_mysql FROM INTEGRATION 'a1b2c3d4-5678-90ab-cdef-EXAMPLE11111'
 SET REFRESH_INTERVAL 7200;
```

Consulte a visualização de catálogo SVV_INTEGRATION para ter informações sobre uma integração ETL zero, como integration_id, target_database, source, refresh_interval, e muito mais.

```
SELECT * FROM svv_integration;
```

O exemplo a seguir cria um banco de dados com base em uma integração com o modo histórico ativado.

```
CREATE DATABASE sample_integration_db FROM INTEGRATION 'a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111' SET HISTORY_MODE = true;
```

Limites de CREATE DATABASE

O Amazon Redshift aplica esses limites aos bancos de dados:

- Máximo de 60 bancos de dados definidos pelo usuário por cluster.
- Máximo de 127 bytes para um nome de banco de dados.

- Um nome de banco de dados não pode ser uma palavra reservada.

Agrupamento de banco de dados

O agrupamento é um conjunto de regras que define como o mecanismo de banco de dados compara e classifica os dados de tipo de caractere em SQL. O agrupamento sem distinção de maiúsculas de minúsculas é o agrupamento mais comumente usado. O Amazon Redshift usa agrupamento sem distinção de maiúsculas e minúsculas para facilitar a migração de outros sistemas de data warehouse. Com o suporte nativo de agrupamento sem distinção de maiúsculas e minúsculas, o Amazon Redshift continua a usar métodos importantes de ajuste ou otimização, como chaves de distribuição, chaves de classificação ou varredura restrita de intervalo.

A cláusula COLLATE especifica o agrupamento padrão para todas as colunas CHAR e VARCHAR no banco de dados. Se CASE_INSENSITIVE for especificado, todas as colunas CHAR ou VARCHAR usarão agrupamento sem distinção de maiúsculas e minúsculas. Para obter mais informações sobre agrupamento, consulte Sequências de colação.

Os dados inseridos ou ingeridos em colunas que não diferenciam maiúsculas e minúsculas manterão suas letras originais. Mas todas as operações de string baseadas em comparação, incluindo classificação e agrupamento, são insensíveis a maiúsculas e minúsculas. Operações de correspondência de padrões, como predicados LIKE, semelhantes a, e funções de expressão regular também são insensíveis a maiúsculas e minúsculas.

As seguintes operações SQL suportam semântica de agrupamento aplicável:

- Operadores de comparação: **=**, **<>**, **<**, **<=**, **>**, **>=**.
- Operador: LIKE
- Cláusula ORDER BY
- Cláusulas GROUP BY
- Funções agregadas que usam comparação de strings, como MIN, MAX e LISTAGG
- Funções da janela, como cláusulas PARTITION BY e cláusulas ORDER BY
- Funções escalares greatest() e least(), STRPOS(), REGEXP_COUNT(), REGEXP_REPLACE(), REGEXP_INSTR(), REGEXP_SUBSTR()
- Cláusula distinta
- UNION, INTERSECT e EXCEPT
- IN LIST

Para consultas externas, incluindo consultas federadas do Amazon Redshift Spectrum e do Aurora PostgreSQL, o agrupamento da coluna VARCHAR ou CHAR é o mesmo que o agrupamento em nível de banco de dados atual.

O seguinte exemplo consulta uma tabela do Amazon Redshift Spectrum:

```
SELECT ci_varchar FROM spectrum.test_collation
WHERE ci_varchar = 'AMAZON';

ci_varchar
----------
amazon
Amazon
AMAZON
AmaZon
(4 rows)
```

Para obter informações sobre como criar tabelas usando o agrupamento de banco de dados, consulte CRIAR TABELA.

Para obter mais informações sobre funções COLLATE, consulte Função COLLATE.

Limitações do agrupamento de banco de dados

Veja as seguintes limitações ao trabalhar com agrupamento de banco de dados no Amazon Redshift:

- Todas as tabelas ou exibições do sistema, incluindo tabelas de catálogo PG e tabelas de sistema do Amazon Redshift, diferenciam maiúsculas de minúsculas.
- Quando o banco de dados do consumidor e o banco de dados do produtor têm agrupamentos de nível de banco de dados diferentes, o Amazon Redshift não oferece suporte a consultas entre bancos de dados e entre clusters.
- O Amazon Redshift não oferece suporte a agrupamento sem distinção de maiúsculas e minúsculas na consulta somente nó líder.

  O seguinte exemplo mostra uma consulta sem distinção entre maiúsculas e minúsculas e o erro que o Amazon Redshift envia:

  ```
  SELECT collate(usename, 'case_insensitive') FROM pg_user;
  ERROR:  Case insensitive collation is not supported in leader node only query.
  ```

- O Amazon Redshift não oferece suporte à interação entre colunas que diferenciam maiúsculas e minúsculas, como operações de comparação, função, junção ou conjunto.

  Os exemplos a seguir mostram erros quando colunas com e sem diferenciação de maiúsculas e minúsculas interagem:

```
CREATE TABLE test
  (ci_col varchar(10) COLLATE case_insensitive,
   cs_col varchar(10) COLLATE case_sensitive,
   cint int,
   cbigint bigint);
```

```
SELECT ci_col = cs_col FROM test;
ERROR:  Query with different collations is not supported yet.
```

```
SELECT concat(ci_col, cs_col) FROM test;
ERROR:  Query with different collations is not supported yet.
```

```
SELECT ci_col FROM test UNION SELECT cs_col FROM test;
ERROR:  Query with different collations is not supported yet.
```

```
SELECT * FROM test a, test b WHERE a.ci_col = b.cs_col;
ERROR:  Query with different collations is not supported yet.
```

```
Select Coalesce(ci_col, cs_col) from test;
ERROR:  Query with different collations is not supported yet.
```

```
Select case when cint > 0 then ci_col else cs_col end from test;
ERROR:  Query with different collations is not supported yet.
```

Para fazer essas consultas funcionarem, use a função COLLATE para converter o agrupamento de uma coluna para corresponder à outra. Para obter mais informações, consulte Função COLLATE.

Exemplos

Criação de um banco de dados

O exemplo a seguir criar um banco de dados denominado TICKIT e de propriedade do usuário DWUSER.

```
create database tickit
with owner dwuser;
```

Consulte a tabela de catálogo PG_DATABASE_INFO para exibir detalhes sobre bancos de dados.

```
select datname, datdba, datconnlimit
from pg_database_info
where datdba > 1;

 datname     | datdba | datconnlimit
-------------+--------+-------------
 admin       |    100 | UNLIMITED
 reports     |    100 | 100
 tickit      |    100 | 100
```

O exemplo a seguir cria um banco de dados chamado **`sampledb`** com o nível de isolamento SNAPSHOT.

```
CREATE DATABASE sampledb ISOLATION LEVEL SNAPSHOT;
```

O exemplo a seguir cria o banco de dados sales_db a partir da unidade de compartilhamento de dados saleshare.

```
CREATE DATABASE sales_db FROM DATASHARE salesshare OF NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Exemplos de agrupamento de banco de dados

Criação de um banco de dados que não diferencia maiúsculas de minúsculas

O exemplo a seguir cria o banco de dados `sampledb`, a tabela T1 e insere dados na tabela T1.

```
create database sampledb collate case_insensitive;
```

Conecte-se ao novo banco de dados que você acabou de criar usando o cliente SQL. Ao usar o Editor de Consultas do Amazon Redshift v2, escolha o `sampledb` no Editor. Ao usar o RSQL, use um comando semelhante ao seguinte.

```
\connect sampledb;
```

```
CREATE TABLE T1 (
  col1 Varchar(20) distkey sortkey
);
```

```
INSERT INTO T1 VALUES ('bob'), ('john'), ('Mary'), ('JOHN'), ('Bob');
```

Em seguida, a consulta encontra resultados com John.

```
SELECT * FROM T1 WHERE col1 = 'John';

 col1
 ------
 john
 JOHN
(2 row)
```

Ordenar por distinção de maiúsculas de minúsculas

O exemplo a seguir mostra a ordenação sem distinção de maiúsculas e minúsculas com a tabela T1. A ordenação de Bob e bob ou John e john não é determinística porque os nomes são iguais em colunas que não diferenciam maiúsculas de minúsculas.

```
SELECT * FROM T1 ORDER BY 1;

 col1
 ------
 bob
 Bob
 JOHN
 john
 Mary
(5 rows)
```

Da mesma forma, o exemplo a seguir mostra a ordem sem distinção entre maiúsculas e minúsculas com a cláusula GROUP BY. Bob e bob são iguais e pertencem ao mesmo grupo. Não é determinístico qual aparece no resultado.

```
SELECT col1, count(*) FROM T1 GROUP BY 1;
```

```
 col1 | count
 -----+------
 Mary |  1
 bob  |  2
 JOHN |  2
(3 rows)
```

Consultar com uma função da janela em colunas que não diferenciam maiúsculas e minúsculas

O exemplo a seguir consulta uma função da janela em uma coluna que não diferencia maiúsculas e minúsculas.

```
SELECT col1, rank() over (ORDER BY col1) FROM T1;

 col1 | rank
 -----+------
 bob  |   1
 Bob  |   1
 john |   3
 JOHN |   3
 Mary |   5
(5 rows)
```

Consultar com a palavra-chave DISTINCT

O exemplo a seguir cria a tabela T1 com a palavra-chave DISTINCT.

```
SELECT DISTINCT col1 FROM T1;

 col1
 ------
 bob
 Mary
 john
(3 rows)
```

Consultar com a cláusula UNION

O exemplo a seguir mostra os resultados da UNION das tabelas T1 e T2.

```
CREATE TABLE T2 AS SELECT * FROM T1;
```

```
SELECT col1 FROM T1 UNION SELECT col1 FROM T2;

 col1
 ------
 john
 bob
 Mary
(3 rows)
```

CREATE DATASHARE

Cria um novo datashare no banco de dados atual. O proprietário desta tabela é o emissor do comando CREATE DATASHARE.

O Amazon Redshift associa cada datashare a um único banco de dados do Amazon Redshift. Você só pode adicionar objetos do banco de dados associado a um datashare. Você pode criar vários conjuntos de dados no mesmo banco de dados do Amazon Redshift.

Para obter mais informações sobre unidades de compartilhamento de dados, consulte Compartilhamento de dados no Amazon Redshift.

Para visualizar informações sobre os conjuntos de dados, use SHOW DATASHARES.

Privilégios obrigatórios

A seguir estão os privilégios obrigatórios para CREATE DATASHARE:

- Superusuário
- Usuários com o privilégio CREATE DATASHARE
- Proprietário do banco de dados

Sintaxe

```
CREATE DATASHARE datashare_name
[[SET] PUBLICACCESSIBLE [=] TRUE | FALSE ];
```

Parâmetros

datashare_name

O nome do datashare. O nome do datashare deve ser exclusivo no namespace do cluster.

[[SET] PUBLICACCESSIBLE]

Cláusula que especifica se o armazenamento de dados pode ser compartilhado para clusters que são acessíveis ao público.

O valor padrão para `SET PUBLICACCESSIBLE` é `FALSE`.

Observações de uso

Por padrão, o proprietário do datashare possui somente o compartilhamento, mas não objetos dentro do compartilhamento.

Somente superusuários e o proprietário do banco de dados podem usar CREATE DATASHARE e delegar privilégios ALTER a outros usuários ou grupos.

Exemplos

O exemplo a seguir cria a unidade de compartilhamento de dados `salesshare`.

```
CREATE DATASHARE salesshare;
```

O exemplo a seguir cria a unidade de compartilhamento de dados `demoshare` que o AWS Data Exchange gerencia.

```
CREATE DATASHARE demoshare SET PUBLICACCESSIBLE TRUE, MANAGEDBY ADX;
```

CREATE EXTERNAL FUNCTION

Cria uma função definida pelo usuário (UDF) escalar baseada em AWS Lambda para o Amazon Redshift. Para obter mais informações sobre as funções definidas pelo usuário do Lambda, consulte UDFs escalares do Lambda.

Privilégios obrigatórios

A seguir estão os privilégios necessários para CREATE EXTERNAL FUNCTION:

- Superusuário
- Usuários com o privilégio CREATE [OR REPLACE] EXTERNAL FUNCTION

Sintaxe

```
CREATE [ OR REPLACE ] EXTERNAL FUNCTION external_fn_name ( [data_type] [, ...] )
RETURNS data_type
{ VOLATILE | STABLE }
LAMBDA 'lambda_fn_name'
IAM_ROLE { default | 'arn:aws:iam::<Conta da AWS-id>:role/<role-name>'
RETRY_TIMEOUT milliseconds
MAX_BATCH_ROWS count
MAX_BATCH_SIZE size [ KB | MB ];
```

Parâmetros

OR REPLACE

Uma cláusula que especifica que se uma função com o mesmo nome e tipo de dados do argumento de entrada ou assinatura já existir, a função existente será substituída. Você só pode substituir uma função por uma nova função que defina um conjunto idêntico de tipos de dados. É preciso ser um superusuário para substituir uma função.

Se uma função for definida com o mesmo nome que uma função existente mas com uma assinatura diferente, uma nova função será criada. Em outras palavras, o nome da função fica sobrecarregado. Para obter mais informações, consulte Sobrecarga de nomes de função.

external_fn_name

O nome da função externa. Se você especificar um nome de esquema (como myschema.myfunction), a função será criada usando o esquema especificado. Caso contrário, a função será criada no esquema atual. Para obter mais informações sobre nomes válidos, consulte Nomes e identificadores.

Recomendamos que você prefixe os nomes de todos os UDF com `f_`. O Amazon Redshift reserva o prefixo `f_` para nomes UDF. Ao usar o prefixo `f_`, você ajuda a garantir que o nome da sua UDF não entre em conflito com nenhum nome de função SQL integrada do Amazon Redshift, atual ou futuro. Para obter mais informações, consulte Evitar conflitos na nomeação da UDF.

data_type

O tipo de dados dos argumentos de entrada. Para obter mais informações, consulte UDFs escalares do Python e UDFs escalares do Lambda.

RETURNS tipo_dados

Tipo de dados do valor retornado pela função. O tipo de dados RETURNS pode ser qualquer tipo de dados padrão do Amazon Redshift. Para obter mais informações, consulte UDFs escalares do Python e UDFs escalares do Lambda.

VOLATILE | STABLE

Informa o otimizador de consulta sobre a volatilidade da função.

Você terá a melhor otimização se rotular sua função com a categoria mais restrita de volatilidade válida para ela. Por ordem de nível de restrição, começando a partir da menos restrita, as categorias são:

- VOLATILE
- STABLE

VOLATILE

Dados os mesmos argumentos, a função pode retornar resultados diferentes em chamadas sucessivas, mesmo para as linhas em uma única instrução. O otimizador de consultas não pode fazer suposições sobre o comportamento de uma função volátil. Uma consulta que usa uma função volátil deve reavaliar a função para cada entrada.

STABLE

Em vista dos mesmos argumentos, a função com certeza retorna os mesmos resultados em chamadas sucessivas processadas em uma única instrução. A função pode retornar resultados diferentes quando chamada em instruções diferentes. Essa categoria possibilita que o otimizador reduza o número de vezes que a função é chamada em uma única instrução.

Se a rigidez escolhida não for válida para a função, existe o risco de o otimizador ignorar algumas chamadas com base nessa rigidez. Isso pode resultar em um conjunto de resultados incorreto.

No momento, a cláusula IMMUTABLE não é compatível com UDFs do Lambda.

LAMBDA 'lambda_fn_name'

O nome da função chamada pelo Amazon Redshift.

Para obter etapas para criar uma função AWS Lambda, consulte Criar uma função do Lambda com o console no AWS LambdaGuia do desenvolvedor.

Para obter informações sobre permissões necessárias para a função do Lambda, consulte Permissões do AWS Lambda no AWS LambdaGuia do desenvolvedor.

IAM_ROLE { default | 'arn:aws:iam::*<Conta da AWS-id>*:role/*<role-name>*'

Use a palavra-chave padrão para que o Amazon Redshift use a função do IAM definida como padrão e associada ao cluster quando o comando CREATE EXTERNAL FUNCTION for executado.

Use o nome do recurso da Amazon (ARN) de uma função do IAM que seu cluster usa para autenticação e autorização. O comando CREATE EXTERNAL FUNCTION está autorizado a invocar funções do Lambda por meio desta função do IAM. Se o cluster tiver uma função do IAM existente com permissões para invocar funções do Lambda anexadas, você poderá substituir o ARN da função. Para obter mais informações, consulte Configurar o parâmetro de autorização para UDFs do Lambda.

A seguir, a sintaxe do parâmetro IAM_ROLE.

```
IAM_ROLE 'arn:aws:iam::aws-account-id:role/role-name'
```

RETRY_TIMEOUT milliseconds

A quantidade de tempo total em milissegundos que o Amazon Redshift usa para os atrasos em recuos de novas tentativas.

Em vez de tentar novamente imediatamente para quaisquer consultas com falha, o Amazon Redshift realiza backoffs e espera por um certo período de tempo entre novas tentativas. Em seguida, o Amazon Redshift tenta novamente a solicitação para executar novamente a consulta com falha até que a soma de todos os atrasos seja igual ou exceda o valor RETRY_TIMEOUT especificado. O valor padrão é 20.000 milissegundos.

Quando uma função do Lambda é chamada, o Amazon Redshift tenta novamente consultas que recebem erros como `TooManyRequestsException`, `EC2ThrottledException`, `eServiceException`.

Você pode definir o parâmetro RETRY_TIMEOUT como 0 milissegundos para evitar quaisquer tentativas para um Lambda UDF.

Contagem de MAX_BATCH_ROWS

O número máximo de linhas que o Amazon Redshift envia em uma única solicitação em lote para uma invocação do Lambda.

O valor mínimo desse parâmetro é 1. O valor máximo é INT_MAX, ou 2.147.483.647.

Esse parâmetro é opcional. O valor padrão é INT_MAX, ou 2.147.483.647.

Tamanho de MAX_BATCH_SIZE [KB | MB]

O tamanho máximo da carga de dados que o Amazon Redshift envia em uma única solicitação em lote para uma invocação do Lambda.

O valor mínimo desse parâmetro é 1 KB. O valor máximo é 5 MB.

O valor padrão desse parâmetro é 5 MB.

KB e MB são opcionais. Se você não definir a unidade de medida, o Amazon Redshift usará KB como padrão.

Observações de uso

Considere o seguinte ao criar UDFs do Lambda:

- A ordem das chamadas da função do Lambda nos argumentos de entrada não é fixa nem garantida. Isso pode variar entre as instâncias de execução de consultas, dependendo da configuração do cluster.
- Não é garantido que as funções sejam aplicadas a cada argumento de entrada somente uma vez. A interação entre o Amazon Redshift e o AWS Lambda pode levar a chamadas repetitivas com as mesmas entradas.

Exemplos

A seguir estão exemplos de uso de funções definidas pelo usuário (UDFs) do Lambda escalar.

Exemplo de UDF Lambda escalar usando uma função Node.js Lambda

O exemplo a seguir cria uma função externa chamada `exfunc_sum` que leva dois inteiros como argumentos de entrada. Esta função retorna a soma como uma saída inteira. O nome da função do

Lambda a ser chamada é `lambda_sum`. A linguagem usada para essa função do Lambda é Node.js 12.x. Especifique a função do IAM. O exemplo usa `'arn:aws:iam::123456789012:user/johndoe'` como a função do IAM.

```
CREATE EXTERNAL FUNCTION exfunc_sum(INT,INT)
RETURNS INT
VOLATILE
LAMBDA 'lambda_sum'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-Exfunc-Test';
```

A função do Lambda recebe a carga útil de solicitação e itera sobre cada linha. Todos os valores em uma única linha são adicionados para calcular a soma dessa linha, que é salva na matriz de resposta. O número de linhas na matriz de resultados é semelhante ao número de linhas recebidas na carga útil da solicitação.

A carga útil da resposta JSON deve ter os dados do resultado no campo “resultados” para que ele seja reconhecido pela função externa. O campo argumentos na solicitação enviada para a função Lambda contém a carga útil de dados. Pode haver várias linhas na carga útil de dados no caso de uma solicitação em lote. A seguinte função do Lambda itera sobre todas as linhas na carga útil de dados de solicitação. Ele também itera individualmente sobre todos os valores dentro de uma única linha.

```
exports.handler = async (event) => {
    // The 'arguments' field in the request sent to the Lambda function contains the
 data payload.
    var t1 = event['arguments'];

    // 'len(t1)' represents the number of rows in the request payload.
    // The number of results in the response payload should be the same as the number
 of rows received.
    const resp = new Array(t1.length);

    // Iterating over all the rows in the request payload.
    for (const [i, x] of t1.entries())
    {
        var sum = 0;
        // Iterating over all the values in a single row.
        for (const y of x) {
            sum = sum + y;
        }
        resp[i] = sum;
```

```
    }
    // The 'results' field should contain the results of the lambda call.
    const response = {
        results: resp
    };
    return JSON.stringify(response);
};
```

O exemplo a seguir chama a função externa com valores literais.

```
select exfunc_sum(1,2);
exfunc_sum
------------
 3
(1 row)
```

O exemplo a seguir cria uma tabela chamada t_sum com duas colunas, c1 e c2, do tipo de dados inteiro e insere duas linhas de dados. Em seguida, a função externa é chamada passando os nomes de coluna desta tabela. As duas linhas da tabela são enviadas em uma solicitação de lote na carga útil de solicitação como uma única invocação do Lambda.

```
CREATE TABLE t_sum(c1 int, c2 int);
INSERT INTO t_sum VALUES (4,5), (6,7);
SELECT exfunc_sum(c1,c2) FROM t_sum;
 exfunc_sum
---------------
 9
 13
(2 rows)
```

Exemplo de UDF Lambda escalar usando o atributo RETRY_TIMEOUT

Na seção a seguir, você pode encontrar um exemplo de como usar o atributo RETRY_TIMEOUT em UDFs do Lambda.

AWS LambdaFunções têm limites de simultaneidade que você pode definir para cada função. Para obter mais informações sobre limites de simultaneidade, consulte Gerenciamento da simultaneidade para uma função do Lambda no Guia do desenvolvedor do AWS Lambda e a publicação Gerenciamento da simultaneidade da função do AWS Lambda no Blog de Computação da AWS.

Quando o número de solicitações que estão sendo atendidas por um UDF Lambda excede os limites de simultaneidade, as novas solicitações recebem o erro `TooManyRequestsException`. O UDF Lambda tenta novamente neste erro até que a soma de todos os atrasos entre as solicitações enviadas para a função do Lambda seja igual ou exceda o valor RETRY_TIMEOUT definido. O valor padrão de RETRY_TIMEOUT é 20.000 milissegundos.

O exemplo a seguir cria uma função do Lambda chamada `exfunc_sleep_3`. Esta função recebe a carga útil de solicitação, itera sobre cada linha e converte a entrada para maiúsculas. Em seguida, dorme por 3 segundos e retorna o resultado. A linguagem usada para esta função do Lambda é Python 3.8.

O número de linhas na matriz de resultados é semelhante ao número de linhas recebidas na carga útil da solicitação. A carga útil da resposta JSON deve ter os dados do resultado no campo `results` para que ele seja reconhecido pela função externa. O campo `arguments` na solicitação enviada para a função do Lambda contém a carga útil de dados. Pode haver várias linhas na carga útil de dados no caso de uma solicitação em lote.

O limite de simultaneidade para essa função é definido especificamente como 1 em simultaneidade reservada para demonstrar o uso do atributo RETRY_TIMEOUT. Quando o atributo é definido como 1, a função do Lambda só pode servir uma solicitação por vez.

```
import json
import time
def lambda_handler(event, context):
    t1 = event['arguments']
    # 'len(t1)' represents the number of rows in the request payload.
    # The number of results in the response payload should be the same as the number of
 rows received.
    resp = [None]*len(t1)

    # Iterating over all rows in the request payload.
    for i, x in enumerate(t1):
        # Iterating over all the values in a single row.
        for j, y in enumerate(x):
            resp[i] = y.upper()

    time.sleep(3)
    ret = dict()
    ret['results'] = resp
    ret_json = json.dumps(ret)
    return ret_json
```

A seguir, dois exemplos adicionais ilustram o atributo RETRY_TIMEOUT. Cada um deles invoca um único UDF Lambda. Ao invocar o UDF Lambda, cada exemplo executa a mesma consulta SQL para invocar o UDF Lambda de duas sessões simultâneas de banco de dados ao mesmo tempo. Quando a primeira consulta que invoca o UDF Lambda está sendo atendida pelo UDF, a segunda consulta recebe o erro `TooManyRequestsException`. Esse resultado ocorre porque você define especificamente a simultaneidade reservada no UDF como 1. Para obter informações sobre como definir a simultaneidade reservada para funções do Lambda, consulte Configurar a simultaneidade reservada.

O primeiro exemplo a seguir define o atributo RETRY_TIMEOUT para o UDF Lambda como 0 milissegundos. Se a solicitação do Lambda receber exceções da função do Lambda, o Amazon Redshift não fará novas tentativas. Esse resultado ocorre porque o atributo RETRY_TIMEOUT está definido como 0.

```
CREATE OR REPLACE EXTERNAL FUNCTION exfunc_upper(varchar)
RETURNS varchar
VOLATILE
LAMBDA 'exfunc_sleep_3'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-Exfunc-Test'
RETRY_TIMEOUT 0;
```

Com o RETRY_TIMEOUT definido como 0, você pode executar as duas consultas a seguir de sessões de banco de dados separadas para ver resultados diferentes.

A primeira consulta SQL que usa o UDF Lambda é executada com êxito.

```
select exfunc_upper('Varchar');
 exfunc_upper
 --------------
 VARCHAR
(1 row)
```

A segunda consulta, que é executada a partir de uma sessão de banco de dados separada ao mesmo tempo, recebe o erro `TooManyRequestsException`.

```
select exfunc_upper('Varchar');
ERROR:  Rate Exceeded.; Exception: TooManyRequestsException; ShouldRetry: 1
DETAIL:
-----------------------------------------------
error:  Rate Exceeded.; Exception: TooManyRequestsException; ShouldRetry: 1
code:      32103
```

```
context:query:     0
location:  exfunc_client.cpp:102
process:   padbmaster [pid=26384]
-----------------------------------------------
```

O segundo exemplo a seguir define o atributo RETRY_TIMEOUT para o UDF Lambda como 3.000 milissegundos. Mesmo se a segunda consulta for executada simultaneamente, o UDF Lambda tenta novamente até que o total de atrasos seja de 3.000 milissegundos. Assim, ambas as consultas são executadas com êxito.

```
CREATE OR REPLACE EXTERNAL FUNCTION exfunc_upper(varchar)
RETURNS varchar
VOLATILE
LAMBDA 'exfunc_sleep_3'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-Exfunc-Test'
RETRY_TIMEOUT 3000;
```

Com o RETRY_TIMEOUT definido como 3.000 milissegundos, você pode executar as duas consultas a seguir de sessões de banco de dados separadas para ver os mesmos resultados.

A primeira consulta SQL que usa o UDF Lambda é executada com êxito.

```
select exfunc_upper('Varchar');
 exfunc_upper
 --------------
 VARCHAR
(1 row)
```

A segunda consulta é executada simultaneamente e o UDF Lambda tenta novamente até que o atraso total seja de 3.000 milissegundos.

```
select exfunc_upper('Varchar');
 exfunc_upper
--------------
 VARCHAR
(1 row)
```

Exemplo de UDF Lambda escalar usando uma função Python Lambda

O exemplo a seguir cria uma função externa chamada `exfunc_multiplication` que multiplica números e retorna um inteiro. Este exemplo incorpora o campo success e `error_msg` na resposta

do Lambda. O campo success é definido como false quando há um estouro de inteiro no resultado da multiplicação, e a propriedade `error_msg` está definida como `Integer multiplication overflow`. A função `exfunc_multiplication` leva três inteiros como argumentos de entrada e retorna a soma como uma saída inteira.

O nome da função do Lambda chamada é `lambda_multiplication`. A linguagem usada para esta função do Lambda é Python 3.8. Especifique a função do IAM.

```
CREATE EXTERNAL FUNCTION exfunc_multiplication(int, int, int)
RETURNS INT
VOLATILE
LAMBDA 'lambda_multiplication'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-Exfunc-Test';
```

A função do Lambda recebe a carga útil de solicitação e itera sobre cada linha. Todos os valores em uma única linha são multiplicados para calcular o resultado dessa linha, que é salvo na lista de respostas. Este exemplo usa um valor de sucesso booleano definido como true por padrão. Se o resultado da multiplicação de uma linha tiver um estouro de inteiro, o valor de sucesso será definido como false. Em seguida, o loop de iteração quebra.

Ao criar a carga útil de resposta, se o valor de sucesso for false, a seguinte função do Lambda adiciona o campo `error_msg` na carga útil. Também define a mensagem de erro como `Integer multiplication overflow`. Se o valor de sucesso for true, os dados do resultado serão adicionados no campo de resultados. O número de linhas na matriz de resultados, se houver, é semelhante ao número de linhas recebidas na carga útil da solicitação.

O campo argumentos na solicitação enviada para a função Lambda contém a carga útil de dados. Pode haver várias linhas na carga útil de dados no caso de uma solicitação em lote. A seguinte função do Lambda itera sobre todas as linhas na carga útil de dados de solicitação e itera individualmente sobre todos os valores dentro de uma única linha.

```
import json
def lambda_handler(event, context):
    t1 = event['arguments']
    # 'len(t1)' represents the number of rows in the request payload.
    # The number of results in the response payload should be the same as the number of
 rows received.
    resp = [None]*len(t1)

    # By default success is set to 'True'.
    success = True
```

```
    # Iterating over all rows in the request payload.
    for i, x in enumerate(t1):
        mul = 1
        # Iterating over all the values in a single row.
        for j, y in enumerate(x):
            mul = mul*y

        # Check integer overflow.
        if (mul >= 9223372036854775807 or mul <= -9223372036854775808):
            success = False
            break
        else:
            resp[i] = mul
    ret = dict()
    ret['success'] = success
    if not success:
        ret['error_msg'] = "Integer multiplication overflow"
    else:
        ret['results'] = resp
    ret_json = json.dumps(ret)

    return ret_json
```

O exemplo a seguir chama a função externa com valores literais.

```
SELECT exfunc_multiplication(8, 9, 2);
  exfunc_multiplication
--------------------------
          144
(1 row)
```

O exemplo a seguir cria uma tabela chamada t_multi com três colunas, c1, c2 e c3, do tipo de dados inteiro. A função externa é chamada passando os nomes das colunas desta tabela. Os dados são inseridos de forma a causar estouro de inteiro para mostrar como o erro é propagado.

```
CREATE TABLE t_multi (c1 int, c2 int, c3 int);
INSERT INTO t_multi VALUES (2147483647, 2147483647, 4);
SELECT exfunc_multiplication(c1, c2, c3) FROM t_multi;
DETAIL:
  -----------------------------------------------
  error:  Integer multiplication overflow
  code:      32004context:
  context:
```

```
  query:     38
  location:  exfunc_data.cpp:276
  process:   query2_16_38 [pid=30494]
  -----------------------------------------------
```

CREATE EXTERNAL MODEL

Tópicos

- Pré-requisitos referentes a CREATE EXTERNAL MODEL
- Privilégios obrigatórios
- Controle de custos
- Sintaxe de CREATE EXTERNAL MODEL
- Parâmetros e configurações de CREATE EXTERNAL MODEL
- Parâmetros da função de inferência CREATE EXTERNAL MODEL

Pré-requisitos referentes a CREATE EXTERNAL MODEL

Antes de usar a instrução CREATE EXTERNAL MODEL, conclua os pré-requisitos em Configuração de cluster para usar o Amazon Redshift ML. A seguir está um resumo de alto nível dos pré-requisitos.

- Crie um cluster do Amazon Redshift com o Console de Gerenciamento da AWS ou a Interface de linha de comando da AWS (AWS CLI).
- Anexe a política de Identity and Access Management (IAM) da AWS ao criar o cluster.
- Para permitir que o Amazon Redshift e o Amazon Bedrock assumam o perfil para interagir com outros serviços, adicione a política de confiança apropriada ao perfil do IAM.
- Habilite o acesso aos LLMs específicos que você deseja usar no console do Amazon Bedrock.
- (Opcional) Se você encontrar exceções de controle de utilização provenientes do Amazon Bedrock, como `Too many requests, please wait before trying again`, mesmo com pequenas quantidades de dados, verifique as cotas em Cotas de serviço na conta do Amazon Bedrock. Verifique se a cota aplicada no nível da conta é pelo menos igual ao valor da cota padrão da AWS para as solicitações InvokeModel do modelo que você está usando.

Para obter detalhes sobre a função do IAM, a política de confiança e outros pré-requisitos, consulte Configuração de cluster para usar o Amazon Redshift ML.

Privilégios obrigatórios

Estes são os privilégios obrigatórios para CREATE EXTERNAL MODEL:

- Superusuário
- Usuários com o privilégio CREATE MODEL
- Funções com privilégio GRANT CREATE MODEL

Controle de custos

O Amazon Redshift ML usa os recursos de cluster existentes para criar modelos de previsão. Portanato, você não precisa pagar a mais. No entanto, a AWS aplica cobranças ao uso do Amazon Bedrock com base no modelo selecionado. Para obter mais informações, consulte Custos de uso do Amazon Redshift ML.

Sintaxe de CREATE EXTERNAL MODEL

A seguir é apresentada a sintaxe completa da instrução CREATE EXTERNAL MODEL.

```
CREATE EXTERNAL MODEL model_name
FUNCTION function_name
IAM_ROLE {default/'arn:aws:iam::<account-id>:role/<role-name>'}
MODEL_TYPE BEDROCK
SETTINGS (
   MODEL_ID model_id
   [, PROMPT 'prompt prefix']
   [, SUFFIX 'prompt suffix']
   [, REQUEST_TYPE {RAW|UNIFIED}]
   [, RESPONSE_TYPE {VARCHAR|SUPER}]
);
```

O comando `CREATE EXTERNAL MODEL` cria uma função de inferência usada para gerar conteúdo.

Esta é a sintaxe de uma função de inferência que `CREATE EXTERNAL MODEL` cria usando `RAW` como `REQUEST_TYPE`:

```
SELECT inference_function_name(request_super)
[FROM table];
```

Esta é a sintaxe de uma função de inferência que `CREATE EXTERNAL MODEL` cria usando `UNIFIED` como `REQUEST_TYPE`:

```
SELECT inference_function_name(input_text, [, inference_config [,
 additional_model_request_fields]])
[FROM table];
```

Para ter informações sobre como usar a função de inferência, consulte Usar um modelo externo para a integração do Amazon Redshift ML com o Amazon Bedrock.

Parâmetros e configurações de CREATE EXTERNAL MODEL

Esta seção descreve os parâmetros e as configurações do comando `CREATE EXTERNAL MODEL`.

Tópicos

- Parâmetros de CREATE EXTERNAL MODEL
- Configurações de CREATE EXTERNAL MODEL

Parâmetros de CREATE EXTERNAL MODEL

model_name

O nome do modelo externo. O nome do modelo em um esquema deve ser exclusivo.

FUNCTION function_name (data_type [,...])

O nome da função de inferência que cria `CREATE EXTERNAL MODEL`. Use a função de inferência para enviar solicitações ao Amazon Bedrock e recuperar texto gerado por ML.

IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>' }

O perfil do IAM que o Amazon Redshift usa para acessar o Amazon Bedrock. Para obter informações sobre a função do IAM, consulte Criar ou atualizar um perfil do IAM para a integração do Amazon Redshift ML com o Amazon Bedrock.

MODEL_TYPE BEDROCK

Especifica o tipo de modelo. O único valor válido é `BEDROCK`.

SETTINGS (MODEL_ID model_id [,...])

Especifica as configurações externas do modelo. Para obter detalhes, consulte a seção a seguir.

Configurações de CREATE EXTERNAL MODEL

MODEL_ID model_id

O identificador do modelo externo; por exemplo, `anthropic.claude-v2`. Para ter informações sobre IDs de modelo do Amazon Bedrock, consulte Amazon Bedrock model IDs.

PROMPT 'prompt prefix'

Especifica um prompt estático que o Amazon Redshift adiciona ao início de cada solicitação de inferência. Compatível apenas com o `REQUEST_TYPE UNIFIED`.

SUFFIX 'prompt suffix'

Especifica um prompt estático que o Amazon Redshift adiciona ao final de cada solicitação de inferência. Compatível apenas com o `REQUEST_TYPE UNIFIED`.

REQUEST_TYPE { RAW | UNIFIED }

Especifica o formato da solicitação enviada ao Amazon Bedrock. Entre os valores válidos estão os seguintes:

- RAW: a função de inferência considera a entrada como um único valor super e sempre envia de volta um valor super. O formato do valor super é específico para o modelo do Amazon Bedrock selecionado. Super é um modelo de previsão que combina vários algoritmos para produzir uma previsão única e aprimorada.
- UNIFIED: a função de inferência usa a API unificada. Todos os modelos têm uma interface unificada e consistente com o Amazon Bedrock. Isso funciona para todos os modelos que comportam mensagens. Esse valor é o padrão.

  Para ter mais informações, consulte a documentação da API Converse na documentação de API do Amazon Bedrock.

RESPONSE_TYPE { VARCHAR | SUPER }

Especifica o formato da resposta. Se `REQUEST_TYPE` for `RAW`, o `RESPONSE_TYPE` é obrigatório e o único valor válido é `SUPER`. Com relação a todos os outros valores de `REQUEST  TYPE`, o valor padrão é `VARCHAR` e `RESPONSE_TYPE` é opcional. Entre os valores válidos estão os seguintes:

- VARCHAR: o Amazon Redshift retorna somente a resposta de texto gerada pelo modelo.
- SUPER: o Amazon Redshift envia de volta toda a resposta JSON gerada pelo modelo como um super. Isso inclui a resposta de texto, além de informações como o motivo da parada e o uso do token de entrada e saída do modelo. Super é um modelo de previsão que combina vários algoritmos para produzir uma previsão única e aprimorada.

Parâmetros da função de inferência CREATE EXTERNAL MODEL

Esta seção descreve os parâmetros válidos para a função de inferência criada pelo comando `CREATE EXTERNAL MODEL`.

Parâmetros da função de inferência CREATE EXTERNAL MODEL usando **`RAW`** como **`REQUEST_TYPE`**

Uma função de inferência criada com `RAW` como `REQUEST_TYPE` tem um argumento super de entrada e sempre envia de volta um tipo de dados super. A sintaxe do super de entrada segue a sintaxe da solicitação do modelo específico selecionado no Amazon Bedrock.

Parâmetros da função de inferência CREATE EXTERNAL MODEL usando **`UNIFIED`** como **`REQUEST_TYPE`**

input_text

O texto que o Amazon Redshift envia ao Amazon Bedrock.

inference_config

Um valor super que contém parâmetros opcionais que o Amazon Redshift envia ao Amazon Bedrock. Isso pode incluir o seguinte:

- maxTokens
- stopSequences
- temperature
- topP

Esses parâmetros são todos opcionais e diferenciam maiúsculas e minúsculas. Para ter informações sobre esses parâmetros, consulte InferenceConfiguration na Referência de API do Amazon Bedrock.

CREATE EXTERNAL SCHEMA

Cria um novo esquema externo no banco de dados atual. Use esse esquema externo para se conectar a bancos de dados do Amazon RDS for PostgreSQL ou Amazon Aurora Edição compatível com PostgreSQL. Também é possível criar um esquema externo que faça referência a um banco de dados em um catálogo de dados externo, como AWS Glue, Athena ou um banco de dados em um metastore do Apache Hive, como Amazon EMR.

O proprietário deste esquema é o emissor do comando CREATE EXTERNAL SCHEMA. Para transferir a propriedade de um esquema externo, use ALTER SCHEMA para alterar o proprietário. Para conceder acesso ao esquema a outros usuários ou grupos de usuário, use o comando GRANT.

Você não pode usar os comandos GRANT ou REVOKE para permissões em uma tabela externa. Em vez disso, conceda ou revogue permissões no esquema externo.

> **Note**
>
> Se você tiver tabelas externas do Redshift Spectrum no catálogo de dados do Amazon Athena, poderá migrar o catálogo de dados do Athena para um AWS Glue Data Catalog. Para usar o catálogo de dados do AWS Glue com o Redshift Spectrum, talvez seja necessário alterar as políticas do AWS Identity and Access Management (IAM). Para obter mais informações, consulte Atualizar para o catálogo de dados do AWS Glue no Manual do usuário do Athena.

Para visualizar detalhes dos esquemas externos, consulte a exibição do sistema SVV_EXTERNAL_SCHEMAS.

Sintaxe

A sintaxe a seguir descreve o comando CREATE EXTERNAL SCHEMA usado para fazer referência a dados usando um catálogo de dados externo. Para obter mais informações, consulte Amazon Redshift Spectrum.

```
CREATE EXTERNAL SCHEMA [IF NOT EXISTS] local_schema_name
FROM [ [ DATA CATALOG ] | HIVE METASTORE | POSTGRES | MYSQL | KINESIS | MSK | REDSHIFT
 | KAFKA ]
[ DATABASE 'database_name' ]
[ SCHEMA 'schema_name' ]
[ REGION 'aws-region' ]
[ IAM_ROLE [ default | 'SESSION' | 'arn:aws:iam::<Conta da AWS-id>:role/<role-
name>' ] ]
[ AUTHENTICATION [ none | iam | mtls] ]
[ AUTHENTICATION_ARN 'acm-certificate-arn' | SECRET_ARN 'ssm-secret- arn' ]
[ URI ['hive_metastore_uri' [ PORT port_number ] | 'hostname' [ PORT port_number ] |
 'Kafka bootstrap URL'] ]
[ CLUSTER_ARN 'arn:aws:kafka:<region>:<Conta da AWS-id>:cluster/msk/<cluster uuid>' ]
[ CATALOG_ROLE [ 'SESSION' | 'catalog-role-arn-string' ] ]
[ CREATE EXTERNAL DATABASE IF NOT EXISTS ]
```

```
[ CATALOG_ID 'Amazon Web Services account ID containing Glue or Lake Formation
 database' ]
```

A sintaxe a seguir descreve o comando CREATE EXTERNAL SCHEMA usado para fazer referência a dados usando uma consulta federada ao RDS POSTGRES ou Aurora PostgreSQL. Você também pode criar um esquema externo que faça referência a fontes de transmissão, como o Kinesis Data Streams. Para obter mais informações, consulte Consultar dados com consultas federadas no Amazon Redshift.

```
CREATE EXTERNAL SCHEMA [IF NOT EXISTS] local_schema_name
FROM POSTGRES
DATABASE 'federated_database_name' [SCHEMA 'schema_name']
URI 'hostname' [ PORT port_number ]
IAM_ROLE [ default | 'arn:aws:iam::<Conta da AWS-id>:role/<role-name>' ]
SECRET_ARN 'ssm-secret-arn'
```

A sintaxe a seguir descreve o comando CREATE EXTERNAL SCHEMA usado para fazer referência a dados usando uma consulta federada ao RDS MySQL ou Aurora MySQL. Para obter mais informações, consulte Consultar dados com consultas federadas no Amazon Redshift.

```
CREATE EXTERNAL SCHEMA [IF NOT EXISTS] local_schema_name
FROM MYSQL
DATABASE 'federated_database_name'
URI 'hostname' [ PORT port_number ]
IAM_ROLE [ default | 'arn:aws:iam::<Conta da AWS-id>:role/<role-name>' ]
SECRET_ARN 'ssm-secret-arn'
```

A sintaxe a seguir descreve o comando CREATE EXTERNAL SCHEMA usado para fazer referência a dados em uma transmissão do Kinesis. Para obter mais informações, consulte Ingestão de streaming para uma visão materializada.

```
CREATE EXTERNAL SCHEMA [IF NOT EXISTS] schema_name
FROM KINESIS
IAM_ROLE [ default | 'arn:aws:iam::<Conta da AWS-id>:role/<role-name>' ]
```

A sintaxe a seguir descreve o comando CREATE EXTERNAL SCHEMA usado para fazer referência ao cluster do Amazon Managed Streaming for Apache Kafka ou Confluent Cloud e os respectivos tópicos para ingestão. Para se conectar, forneça o URI do agente. Para obter mais informações, consulte Ingestão de streaming para uma visão materializada.

```
CREATE EXTERNAL SCHEMA [IF NOT EXISTS] schema_name
FROM KAFKA
[ IAM_ROLE [ default | 'arn:aws:iam::<Conta da AWS-id>:role/<role-name>' ] ]
URI 'Kafka bootstrap URI'
AUTHENTICATION [ none | iam | mtls ]
[ AUTHENTICATION_ARN 'acm-certificate-arn' | SECRET_ARN 'ssm-secret- arn' ];
```

A sintaxe a seguir descreve o comando CREATE EXTERNAL SCHEMA usado para fazer referência a dados usando uma consulta entre bancos de dados.

```
CREATE EXTERNAL SCHEMA local_schema_name
FROM  REDSHIFT
DATABASE 'redshift_database_name' SCHEMA 'redshift_schema_name'
```

Parâmetros

IF NOT EXISTS

Cláusula que indica que, se o esquema especificado existe, o comando não deve fazer alterações e retorna uma mensagem informando que o esquema existe, em vez de encerrar com um erro. Esta cláusula é útil para realizar desenvolvimento de scripts para que o script não falhe se o comando CREATE EXTERNAL SCHEMA tentar criar um esquema que já existe.

local_schema_name

Nome do novo esquema externo. Para obter mais informações sobre nomes válidos, consulte Nomes e identificadores.

FROM [DATA CATALOG] | HIVE METASTORE | POSTGRES | MYSQL | KINESIS | MSK | REDSHIFT

Uma palavra-chave que indica onde o banco de dados externo está localizado.

DATA CATALOG indica que o banco de dados externo está definido no catálogo de dados do Athena ou do AWS Glue Data Catalog.

Se o banco de dados externo estiver definido em um catálogo de dados em uma região da AWS diferente, o parâmetro REGION será obrigatório. DATA CATALOG é o valor padrão.

HIVE METASTORE indica que o banco de dados externo está definido em um metastore do Apache Hive. Se HIVE METASTORE estiver especificado, URI será obrigatório.

POSTGRES indica que o banco de dados externo está definido em RDS PostgreSQL ou Aurora PostgreSQL.

O MYSQL indica que o banco de dados externo está definido no RDS MySQL ou no Aurora MySQL.

O KINESIS indica que a fonte de dados é uma transmissão do Kinesis Data Streams.

O MSK indica que a fonte de dados é um cluster provisionado ou sem servidor do Amazon MSK.

KAFKA indica que a fonte de dados é um cluster do Kafka. Você pode usar essa palavra-chave para o Amazon MSK e o Confluent Cloud.

FROM REDSHIFT

Uma palavra-chave que indica que o banco de dados está localizado no Amazon Redshift.

DATABASE 'redshift_database_name' SCHEMA 'redshift_schema_name'

O nome do banco de dados do Amazon Redshift.

O redshift_schema_name indica o esquema no Amazon Redshift. O valor redshift_schema_name é `public`.

DATABASE 'federated_database_name'

Uma palavra-chave que indica o nome do banco de dados externo em um mecanismo de banco de dados PostgreSQL ou MySQL compatível.

[SCHEMA 'schema_name']

O schema_name indica o esquema em um mecanismo de banco de dados PostgreSQL compatível. O schema_name padrão é `public`.

Você não pode especificar um SCHEMA ao configurar uma consulta federada para um mecanismo de banco de dados MySQL compatível.

REGION 'aws-region'

Se o banco de dados externo for definido em um catálogo de dados do Athena ou do AWS Glue Data Catalog, a região da AWS na qual o banco de dados estará localizado. Esse parâmetro é necessário se o banco de dados for definido em um catálogo de dados .

URI ['hive_metastore_uri' [PORT port_number] | 'hostname' [PORT port_number] | 'Kafka bootstrap URI']

O URI de nome de host e port_number de um mecanismo de banco de dados PostgreSQL ou MySQL compatível. O hostname é o nó de cabeçalho do conjunto de réplicas. O endpoint deve ser acessível (roteável) pelo cluster do Amazon Redshift. O port_number padrão do PostgreSQL é 5432. O port_number padrão do MySQL é 3306.

> **Note**
>
> O mecanismo de banco de dados PostgreSQL ou MySQL compatível deve estar na mesma VPC do cluster do Amazon Redshift com um grupo de segurança vinculando o Amazon Redshift e o RDS url-rsPostgreSQL ou o Aurora PostgreSQL. Além disso, você pode usar o roteamento aprimorado de VPC para configurar um caso de uso entre VPCs. Para obter mais informações, consulte Trabalhando com endpoints da VPC gerenciados por Redshift.

Especificar um URI de metastore do Hive

Se o banco de dados estiver em um metastore do Hive, especifique o URI e, como opção, o número da porta do metastore. O número da porta padrão é 9083.

Um URI não contém uma especificação de protocolo (“http://”). Um exemplo de URI válido: `uri '172.10.10.10'`.

Especificar um URI de agente para ingestão de streaming

Com a inclusão do URI bootstrap-broker, é possível conectar-se com um cluster do Amazon MSK ou Confluent Cloud e receber dados transmitidos. Consulte mais informações e um exemplo em Conceitos básicos da ingestão de streaming do Amazon Managed Streaming para Apache Kafka.

IAM_ROLE [default | 'SESSION' | 'arn:aws:iam::*<Conta da AWS-id>*:role/*<role-name>*']

Use a palavra-chave padrão para que o Amazon Redshift use a função do IAM definida como padrão e associada ao cluster quando o comando CREATE EXTERNAL SCHEMA for executado.

Use `'SESSION'` se você se conectar ao cluster do Amazon Redshift usando uma identidade federada e acesse as tabelas do esquema externo criado usando esse comando. Para obter mais informações, consulte “Using a federated identity to manage Amazon Redshift access to local

resources and Amazon Redshift Spectrum external tables" (Usar uma identidade federada para gerenciar o acesso do Amazon Redshift aos recursos locais e a tabelas externas do Amazon Redshift Spectrum), que explica como configurar uma identidade federada. Observe que essa configuração, que usa `'SESSION'` no lugar do ARN, só poderá ser usada se o esquema for criado usando `DATA CATALOG`.

Use o nome do recurso da Amazon (ARN) de uma função do IAM que seu cluster usa para autenticação e autorização. No mínimo, a função do IAM deve ter permissão para executar uma operação LIST no bucket do Amazon S3 a ser acessado e uma operação GET nos objetos do Amazon S3 que constam no bucket.

O exemplo a seguir mostra a sintaxe da string do parâmetro IAM_ROLE para um único ARN.

```
IAM_ROLE 'arn:aws:iam::<aws-account-id>:role/<role-name>'
```

Você pode encadear funções para que seu cluster possa assumir outra função do IAM, possivelmente pertencente a outra conta. Você pode encadear até 10 funções. Para ver um exemplo de perfis de encadeamento, consulte Encadeamento de funções do IAM no Amazon Redshift Spectrum.

Anexe a essa função do IAM uma política de permissões do IAM semelhante à política descrita a seguir.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AccessSecret",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetResourcePolicy",
                "secretsmanager:GetSecretValue",
                "secretsmanager:DescribeSecret",
                "secretsmanager:ListSecretVersionIds"
            ],
            "Resource": "arn:aws:secretsmanager:us-
west-2:123456789012:secret:my-rds-secret-VNenFy"
        },
        {
```

```
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword",
                "secretsmanager:ListSecrets"
            ],
            "Resource": "*"
        }
    ]
}
```

Para obter as etapas para criar uma função do IAM a ser usada com consulta federada, consulte Criar um segredo e uma função do IAM para usar consultas federadas.

> **Note**
>
> Não inclua espaços na lista de funções encadeadas.

O seguinte mostra a sintaxe do encadeamento de três funções.

```
IAM_ROLE 'arn:aws:iam::<aws-account-id>:role/<role-1-name>,arn:aws:iam::<aws-account-id>:role/<role-2-name>,arn:aws:iam::<aws-account-id>:role/<role-3-name>'
```

SECRET_ARN 'ssm-secret-arn'

O nome do recurso da Amazon (ARN) de um segredo de mecanismo de banco de dados PostgreSQL ou MySQL compatível criado usando o AWS Secrets Manager. Para ter informações sobre como criar e recuperar um ARN de um segredo, consulte Manage secrets with AWS Secrets Manager no Guia do usuário do AWS Secrets Manager e Recuperar o nome do recurso da Amazon (ARN) do segredo no Amazon Redshift.

CATALOG_ROLE ['SESSION' | catalog-role-arn-string]

Use `'SESSION'` para se conectar ao cluster do Amazon Redshift usando uma identidade federada para autenticação e autorização para o catálogo de dados. Para obter mais informações sobre como concluir as etapas da identidade federada, consulte “Using a federated identity to manage Amazon Redshift access to local resources and Amazon Redshift Spectrum external tables” (Usar uma identidade federada para gerenciar o acesso do Amazon Redshift aos recursos locais e a tabelas externas do Amazon Redshift Spectrum). Observe que o perfil `'SESSION'` só poderá ser usado se o esquema for criado no CATÁLOGO DE DADOS.

Use o nome do recurso da Amazon (ARN) de um perfil do IAM que o cluster usa para autenticação e autorização do catálogo de dados.

Se CATALOG_ROLE não for especificado, o Amazon Redshift usará a IAM_ROLE especificada. A função do catálogo deve ter permissão para acessar o catálogo de dados no AWS Glue ou Athena. Para obter mais informações, consulte Políticas do IAM do Amazon Redshift Spectrum.

O exemplo a seguir mostra a sintaxe da string do parâmetro CATALOG_ROLE para um único ARN.

```
CATALOG_ROLE 'arn:aws:iam::<aws-account-id>:role/<catalog-role>'
```

Você pode encadear funções para que seu cluster possa assumir outra função do IAM, possivelmente pertencente a outra conta. Você pode encadear até 10 funções. Para obter mais informações, consulte Encadeamento de funções do IAM no Amazon Redshift Spectrum.

> **Note**
>
> A lista de funções encadeadas não deve incluir espaços.

O seguinte mostra a sintaxe do encadeamento de três funções.

```
CATALOG_ROLE 'arn:aws:iam::<aws-account-id>:role/<catalog-role-1-
name>,arn:aws:iam::<aws-account-id>:role/<catalog-role-2-name>,arn:aws:iam::<aws-
account-id>:role/<catalog-role-3-name>'
```

CREATE EXTERNAL DATABASE IF NOT EXISTS

Cláusula que cria um banco de dados externo com o nome especificado pelo argumento DATABASE, se o banco de dados externo especificado não existir. Se o banco de dados externo especificado existir, o comando não realizará alterações. Nesse caso, o comando retorna uma mensagem informando que o banco de dados externo existe, em vez de finalizar com um erro.

> **Note**
>
> CREATE EXTERNAL DATABASE IF NOT EXISTS não pode ser usado com o comando HIVE METASTORE.

Para usar CREATE EXTERNAL DATABASE IF NOT EXISTS com um catálogo de dados habilitado para AWS Lake Formation, você precisa da permissão `CREATE_DATABASE` no catálogo de dados.

CATALOG_ID “ID da conta da Amazon Web Services que contém o banco de dados do Glue ou do Lake Formation”

O ID da conta em que o banco de dados do catálogo de dados está armazenado.

`CATALOG_ID` só pode ser especificado se você planeja se conectar ao cluster do Amazon Redshift ou ao Amazon Redshift Serverless usando uma identidade federada para autenticação e autorização do catálogo de dados por meio da definição de uma das seguintes opções:

- `CATALOG_ROLE` para `'SESSION'`
- `IAM_ROLE` como `'SESSION'` e `'CATALOG_ROLE'` definido como o padrão

Para obter mais informações sobre como concluir as etapas da identidade federada, consulte “Using a federated identity to manage Amazon Redshift access to local resources and Amazon Redshift Spectrum external tables” (Usar uma identidade federada para gerenciar o acesso do Amazon Redshift aos recursos locais e a tabelas externas do Amazon Redshift Spectrum).

AUTHENTICATION

O tipo de autenticação definido para ingestão de streaming. A ingestão de streaming com tipos de autenticação funciona com o Managed Streaming for Apache Kafka. Os tipos de `AUTHENTICATION` são os seguintes:

- none: especifica que não há nenhuma etapa obrigatória. Isso corresponde ao acesso não autenticado no MSK ou ao texto simples com TLS no Apache Kafka.
- iam: especifica a autenticação do IAM. Ao escolher isso, o perfil do IAM tem permissões para autenticação do IAM. Para obter mais informações sobre o esquema externo, consulte Conceitos básicos da ingestão de streaming de origens do Apache Kafka.
- mtls: especifica que a segurança da camada de transporte mútuo oferece comunicação segura ao facilitar a autenticação entre um cliente e um servidor. Nesse caso, o cliente é o Redshift e o servidor é o Amazon MSK. Para ter mais informações sobre como configurar a ingestão de streaming com mTLS, consulte Autenticação com mTLS para ingestão de streaming do Redshift por meio de origens do Apache Kafka.

AUTHENTICATION_ARN

O ARN do certificado do Gerenciador de certificados da AWS usado pelo Amazon Redshift para autenticação mTLS com o Amazon MSK. O ARN está disponível no console do ACM quando você escolhe o certificado emitido.

CLUSTER_ARN

Para ingestão de streaming, o CLUSTER_ARN é o identificador do cluster do Amazon Managed Streaming para Apache Kafka do qual você está transmitindo. Ao usar CLUSTER_ARN, é necessária uma política de perfil do IAM que inclua a permissão `kafka:GetBootstrapBrokers`. Essa opção é fornecida para compatibilidade com versões anteriores. No momento, recomendamos usar a opção de URI do agente de bootstrap para conexão com os clusters do Amazon Managed Streaming para Apache Kafka. Para obter mais informações, consulte Ingestão de streaming.

Observações de uso

Para limites ao usar o catálogo de dados do Athena, consulte Limites do Athena na Referência geral da AWS.

Para os limites ao usar o AWS Glue Data Catalog, consulte Limites do AWS Glue no Referência geral da AWS.

Esses limites não se aplicam a um metastore do Hive.

O máximo de esquemas permitido por banco de dados é 9,9 mil. Para obter mais informações, consulte Cotas e limites no Guia de gerenciamento do Amazon Redshift.

Para cancelar o registro do esquema, use o comando DROP SCHEMA.

Para visualizar detalhes dos esquemas externos, consulte estas visualizações do sistema:

- SVV_EXTERNAL_SCHEMAS
- SVV_EXTERNAL_TABLES
- SVV_EXTERNAL_COLUMNS

Exemplos

O exemplo a seguir cria um esquema externo usando um banco de dados em um catálogo de dados denominado `sampledb` na região Oeste dos EUA (Oregon). Use esse exemplo com um catálogo de dados do Athena ou AWS Glue.

```
create external schema spectrum_schema
from data catalog
database 'sampledb'
region 'us-west-2'
iam_role 'arn:aws:iam::123456789012:role/MySpectrumRole';
```

O exemplo a seguir cria um esquema externo e um novo banco de dados externo denominado `spectrum_db`.

```
create external schema spectrum_schema
from data catalog
database 'spectrum_db'
iam_role 'arn:aws:iam::123456789012:role/MySpectrumRole'
create external database if not exists;
```

O exemplo a seguir cria um esquema externo usando um banco de dados do metastore do Hive denominado `hive_db`.

```
create external schema hive_schema
from hive metastore
database 'hive_db'
uri '172.10.10.10' port 99
iam_role 'arn:aws:iam::123456789012:role/MySpectrumRole';
```

O exemplo a seguir encadeia funções para usar a função `myS3Role` para acessar o Amazon S3 e usa `myAthenaRole` para acesso ao catálogo de dados. Para obter mais informações, consulte Encadeamento de funções do IAM no Amazon Redshift Spectrum.

```
create external schema spectrum_schema
from data catalog
database 'spectrum_db'
iam_role 'arn:aws:iam::123456789012:role/myRedshiftRole,arn:aws:iam::123456789012:role/
myS3Role'
catalog_role 'arn:aws:iam::123456789012:role/myAthenaRole'
```

```
create external database if not exists;
```

O exemplo a seguir cria um esquema externo que faz referência a um banco de dados do Aurora PostgreSQL.

```
CREATE EXTERNAL SCHEMA [IF NOT EXISTS] myRedshiftSchema
FROM POSTGRES
DATABASE 'my_aurora_db' SCHEMA 'my_aurora_schema'
URI 'endpoint to aurora hostname' PORT 5432
IAM_ROLE 'arn:aws:iam::123456789012:role/MyAuroraRole'
SECRET_ARN 'arn:aws:secretsmanager:us-east-2:123456789012:secret:development/
MyTestDatabase-AbCdEf'
```

O exemplo a seguir cria um esquema externo para referenciar o sales_db importado no cluster de consumidor.

```
CREATE EXTERNAL SCHEMA sales_schema FROM REDSHIFT DATABASE 'sales_db' SCHEMA 'public';
```

O exemplo a seguir cria um esquema externo que faz referência a um banco de dados do Aurora MySQL.

```
CREATE EXTERNAL SCHEMA [IF NOT EXISTS] myRedshiftSchema
FROM MYSQL
DATABASE 'my_aurora_db'
URI 'endpoint to aurora hostname'
IAM_ROLE 'arn:aws:iam::123456789012:role/MyAuroraRole'
SECRET_ARN 'arn:aws:secretsmanager:us-east-2:123456789012:secret:development/
MyTestDatabase-AbCdEf'
```

CREATE EXTERNAL TABLE

Cria uma nova tabela externa no esquema especificado. Todas as tabelas externas devem ser criadas em um esquema externo. O caminho de pesquisa não é compatível com esquemas e tabelas externos. Para obter mais informações, consulte CREATE EXTERNAL SCHEMA.

Além das tabelas externas criadas usando o comando de CREATE EXTERNAL TABLE, o Amazon Redshift pode fazer referência a tabelas externas definidas em um catálogo do AWS Glue ou do AWS Lake Formation ou em uma metastore do Apache Hive. Use o comando CREATE EXTERNAL SCHEMA para registrar um banco de dados externo definido no catálogo externo e disponibilize

as tabelas externas para uso no Amazon Redshift. Se a tabela externa existir em um catálogo do AWS Glue ou do AWS Lake Formation ou na metastore do Hive, você não precisará criar uma tabela usando CREATE EXTERNAL TABLE. Para visualizar as tabelas externas, consulte a exibição do sistema SVV_EXTERNAL_TABLES.

Ao executar o comando CREATE EXTERNAL TABLE AS, você cria uma tabela externa com base na definição da coluna de uma consulta e grava os resultados dessa consulta no Amazon S3. Os resultados estão no Apache Parquet ou no formato de texto delimitado. Se a tabela externa tiver uma chave ou chaves de partições, o Amazon Redshift particionará novos arquivos de acordo com essas chaves de partição e registrará novas partições no catálogo externo automaticamente. Para obter mais informações sobre CREATE EXTERNAL TABLE AS, consulte Observações de uso.

Você pode consultar uma tabela externa usando a mesma sintaxe de SELECT que usa com outras tabelas do Amazon Redshift. Também é possível usar a sintaxe INSERT para gravar novos arquivos no local da tabela externa no Amazon S3. Para obter mais informações, consulte INSERT (tabela externa).

Para criar uma exibição com uma tabela externa, inclua a cláusula WITH NO SCHEMA BINDING na instrução CREATE VIEW.

Não é possível executar CREATE EXTERNAL TABLE em uma transação (BEGIN … END). Para obter mais informações sobre transações, consulte Níveis de isolamento no Amazon Redshift.

Privilégios obrigatórios

Para criar tabelas externas, você deve ser proprietário do esquema externo ou um superusuário. Para transferir a propriedade de um esquema externo, use ALTER SCHEMA para alterar o proprietário. O acesso a tabelas externas é controlado pelo acesso ao esquema externo. Não é possível usar o comando GRANT ou REVOKE para permissões em uma tabela externa. Em vez disso, conceda ou revogue USAGE no esquema externo.

As Observações de uso têm informações adicionais sobre permissões específicas para tabelas externas.

Sintaxe

```
CREATE EXTERNAL TABLE
external_schema.table_name
(column_name data_type [, …] )
[ PARTITIONED BY (col_name data_type [, … ] )]
```

```
[ { ROW FORMAT DELIMITED row_format |
  ROW FORMAT SERDE 'serde_name'
  [ WITH SERDEPROPERTIES ( 'property_name' = 'property_value' [, ...] ) ] } ]
STORED AS file_format
LOCATION { 's3://bucket/folder/' | 's3://bucket/manifest_file' }
[ TABLE PROPERTIES ( 'property_name'='property_value' [, ...] ) ]
```

Veja a seguir a sintaxe de CREATE EXTERNAL TABLE AS.

```
CREATE EXTERNAL TABLE
external_schema.table_name
[ PARTITIONED BY (col_name [, … ] ) ]
[ ROW FORMAT DELIMITED row_format ]
STORED AS file_format
LOCATION { 's3://bucket/folder/' }
[ TABLE PROPERTIES ( 'property_name'='property_value' [, ...] ) ]
 AS
 { select_statement }
```

Parâmetros

external_schema.table_name

Nome da tabela a ser criada, qualificada por um nome de esquema externo. As tabelas externas devem ser criadas em um esquema externo. Para obter mais informações, consulte CREATE EXTERNAL SCHEMA.

O tamanho máximo de um nome de tabela é 127 bytes; nomes mais longos são truncados para ter no máximo 127 bytes. É possível usar caracteres multibyte UFT-8 até um máximo de quatro bytes. O Amazon Redshift aplica um limite de 9.900 tabelas por cluster, incluindo tabelas temporárias definidas pelo usuário e tabelas temporárias criadas pelo Amazon Redshift durante o processamento de consultas ou a manutenção do sistema. Como opção, é possível qualificar o nome da tabela com o nome do banco de dados. No exemplo a seguir, o nome do banco de dados é `spectrum_db`, o nome do esquema externo é `spectrum_schema` e o nome da tabela é `test`.

```
create external table spectrum_db.spectrum_schema.test (c1 int)
stored as parquet
location 's3://amzn-s3-demo-bucket/myfolder/';
```

Se o banco de dados ou esquema especificado não existir, a tabela não será criada e a instrução retornará um erro. Você não pode criar tabelas ou exibições nos bancos de dados do sistema `template0`, `template1`, `padb_harvest` ou `sys:internal`.

O nome da tabela deve ser exclusivo para o esquema especificado.

Para obter mais informações sobre nomes válidos, consulte Nomes e identificadores.

(nome_coluna tipo_dados)

O nome e o tipo de dados de cada coluna que está sendo criada.

O tamanho máximo de um nome de coluna é 127 bytes; nomes mais longos são truncados para ter no máximo 127 bytes. É possível usar caracteres multibyte UFT-8 até um máximo de quatro bytes. Você não pode especificar nomes de coluna `"$path"` ou `"$size"`. Para obter mais informações sobre nomes válidos, consulte Nomes e identificadores.

Por padrão, o Amazon Redshift cria tabelas externas com as pseudocolunas `$path` e `$size`. Você pode desabilitar a criação de pseudocolunas em uma sessão. Basta definir o parâmetro de configuração `spectrum_enable_pseudo_columns` como `false`. Para obter mais informações, consulte Pseudocolunas.

Se as pseudocolunas forem habilitadas, o número máximo de colunas que você poderá definir em uma única tabela será 1.598. Se pseudocolunas não estiverem habilitadas, o número máximo de colunas que poderá ser definido em uma única tabela será 1.600.

Se você estiver criando uma "tabela larga", assegure que sua lista de colunas não exceda os limites de largura de linha para resultados intermediários durante cargas e processamento de consultas. Para obter mais informações, consulte Observações de uso.

Para um comando CREATE EXTERNAL TABLE AS, não é necessária uma lista de colunas, pois elas são derivadas da consulta.

data_type

Os seguintes Tipos de dados são compatíveis:

- SMALLINT (INT2)
- INTEGER (INT, INT4)
- BIGINT (INT8)
- DECIMAL (NUMERIC)
- REAL (FLOAT4)

- DOUBLE PRECISION (FLOAT8)
- BOOLEAN (BOOL)
- CHAR (CHARACTER)
- VARCHAR (CHARACTER VARYING)
- VARBYTE (CHARACTER VARYING): pode ser usado com arquivos de dados Parquet e ORC, e somente com tabelas não particionadas.
- DATE: pode ser usado somente com arquivos de dados de texto, Parquet ou ORC, ou como uma coluna de partição.
- TIMESTAMP

Em DATE, você pode usar os formatos conforme descrito a seguir. Para valores mensais representados usando dígitos, estes formatos são compatíveis:

- `mm-dd-yyyy` Por exemplo, `05-01-2017`. Esse é o padrão.
- `yyyy-mm-dd`, onde o ano é representado por mais de 2 dígitos. Por exemplo, `2017-05-01`.

Para valores mensais representados usando abreviações de três letras, estes formatos são compatíveis:

- `mmm-dd-yyyy` Por exemplo, `may-01-2017`. Esse é o padrão.
- `dd-mmm-yyyy`, onde o ano é representado por mais de 2 dígitos. Por exemplo, `01-may-2017`.
- `yyyy-mmm-dd`, onde o ano é representado por mais de 2 dígitos. Por exemplo, `2017-may-01`.

Para valores de ano consistentemente inferiores a 100, o ano é calculado desta maneira:

- Se o ano for inferior a 70, o ano será calculado como o ano mais 2000. Por exemplo, a data 05-01-17 no formato `mm-dd-yyyy` é convertida para `05-01-2017`.
- Se o ano for inferior a 100 e maior que 69, o ano será calculado como o ano mais 1900. Por exemplo, a data 05-01-89 no formato `mm-dd-yyyy` é convertida para `05-01-1989`.
- Para valores de ano representados por dois dígitos, adicione zeros à esquerda para representar o ano em 4 dígitos.

Os valores de data e hora nos arquivos de texto devem estar no formato `yyyy-mm-dd HH:mm:ss.SSSSSS`, como mostra o seguinte valor de data e hora de exemplo: `2017-05-01 11:30:59.000000`.

O comprimento de uma coluna VARCHAR é definido em bytes, não em caracteres. Por exemplo, uma coluna VARCHAR(12) pode conter 12 caracteres de único byte ou 6 caracteres de dois

bytes. Quando você consulta uma tabela externa, os resultados são truncados para se adequar ao tamanho da coluna definido sem retornar um erro. Para obter mais informações, consulte Armazenamento e intervalos.

Para obter uma melhor performance, recomendamos especificar o menor tamanho de coluna que se ajusta aos seus dados. Para encontrar o tamanho máximo em bytes dos valores em uma coluna, use a função OCTET_LENGTH. O exemplo a seguir retorna o tamanho máximo de valores na coluna de email.

```
select max(octet_length(email)) from users;

max
---
 62
```

PARTITIONED BY (nome_col tipo_dados [, …])

Cláusula que define uma tabela particionada com uma ou mais colunas de partição. Um diretório de dados separado é usado para cada combinação específica, o que pode melhorar a performance da consulta em algumas condições. As colunas particionadas não existem na própria tabela de dados. Se você usar um valor para col_name que é o mesmo valor usado na coluna da tabela, obterá um erro.

Depois de criar uma tabela particionada, altere-a usando uma instrução ALTER TABLE … ADD PARTITION para registrar novas partições no catálogo externo. Ao adicionar uma partição, você define a localização da subpasta no Amazon S3 que contém dados da partição.

Por exemplo, se a tabela `spectrum.lineitem_part` for definida com `PARTITIONED BY (l_shipdate date)`, execute o comando ALTER TABLE a seguir para adicionar uma partição.

```
ALTER TABLE spectrum.lineitem_part ADD PARTITION (l_shipdate='1992-01-29')
LOCATION 's3://spectrum-public/lineitem_partition/l_shipdate=1992-01-29';
```

Se você estiver usando CREATE EXTERNAL TABLE AS, não será necessário executar ALTER TABLE…ADD PARTITION. O Amazon Redshift registra novas partições no catálogo externo. O Amazon Redshift também grava automaticamente os dados correspondentes nas partições no Amazon S3 com base na chave ou nas chaves de partição definidas na tabela.

Para visualizar partições, consulte a exibição do sistema SVV_EXTERNAL_PARTITIONS.

> **Note**
>
> Para um comando CREATE EXTERNAL TABLE AS, não é necessário especificar o tipo de dados da coluna de partição, pois essa coluna é derivada da consulta.

ROW FORMAT DELIMITED formatodelinha

Cláusula que especifica o formato de dados subjacentes. Os valores possíveis para rowformat são os seguintes:

- LINES TERMINATED BY 'delimiter'
- FIELDS TERMINATED BY 'delimiter'

Especifique um único caractere ASCII para 'delimiter'. É possível especificar caracteres ASCII não imprimíveis usando o sistema octal no formato `'\ddd'`, em que *d* é um dígito octal (0-7) até “\177”. O exemplo a seguir especifica o caractere BEL (sino) usando o sistema octal.

```
ROW FORMAT DELIMITED FIELDS TERMINATED BY '\007'
```

Se ROW FORMAT for omitido, o formato padrão será DELIMITED FIELDS TERMINATED BY '\A' (início do cabeçalho) e LINES TERMINATED BY '\n' (nova linha).

ROW FORMAT SERDE 'serde_name' , [WITH SERDEPROPERTIES ('property_name' = 'property_value' [, ...])]

Uma cláusula que especifica o formato SERDE para os dados subjacentes.

'serde_name'

O nome de SerDe. Você pode especificar os seguintes formatos:

- org.apache.hadoop.hive.serde2.RegexSerDe
- com.amazonaws.glue.serde.GrokSerDe
- org.apache.hadoop.hive.serde2.OpenCSVSerde

Esse parâmetro é compatível com a seguinte propriedade SerDe para OpenCSVSerde:

```
'wholeFile' = 'true'
```

Defina a propriedade `wholeFile` para `true` a fim de analisar corretamente novos caracteres de linha (\n) dentro de strings entre aspas para solicitações de OpenCSV.

- org.openx.data.jsonserde.JsonSerDe
  - O JSON SERDE também dá suporte aos arquivos Ion.
  - O JSON bastante deve ser bem formado.
  - Os timestamps em Ion e JSON precisam ter formato ISO8601.
  - Esse parâmetro é compatível com a seguinte propriedade SerDe para JsonSerDe:

    ```
    'strip.outer.array'='true'
    ```

    Processa arquivos Ion/JSON contendo uma matriz muito grande entre colchetes externos ([…]) como se contivesse vários registros JSON dentro da matriz.
- com.amazon.ionhiveserde.IonHiveSerDe

  O formato Amazon ION fornece formatos de texto e binário, além dos tipos de dados. Para uma tabela externa que faz referência a dados no formato ION, mapeie cada coluna na tabela externa para o elemento correspondente nos dados do formato ION. Para obter mais informações, consulte Amazon Ion. Também é necessário especificar os formatos de entrada e saída.

WITH SERDEPROPERTIES ('property_name' = 'property_value' [, ...])]

Opcionalmente, especifique nomes e valores de propriedade, separados por vírgulas.

Se ROW FORMAT for omitido, o formato padrão será DELIMITED FIELDS TERMINATED BY '\A' (início do cabeçalho) e LINES TERMINATED BY '\n' (nova linha).

STORED AS formato do arquivo

Formato para arquivos de dados.

Os formatos válidos são:

- PARQUET
- RCFILE (somente para dados que usem ColumnarSerDe, não LazyBinaryColumnarSerDe)
- SEQUENCEFILE
- TEXTFILE (para arquivos de texto, inclusive arquivos JSON).
- ORC
- AVRO
- INPUTFORMAT 'input_format_classname' OUTPUTFORMAT 'output_format_classname'

O comando CREATE EXTERNAL TABLE AS oferece suporte somente a dois formatos de arquivo, TEXTFILE e PARQUET.

Para INPUTFORMAT e OUTPUTFORMAT, especifique um nome de classe, conforme exibido no exemplo a seguir:

```
'org.apache.hadoop.mapred.TextInputFormat'
```

LOCATION { 's3://bucket/folder/' | 's3://bucket/manifest_file'}

O caminho para a pasta ou bucket do Amazon S3 que contém arquivos de dados ou um arquivo manifesto que contém uma lista de caminhos de objetos do Amazon S3. Os buckets devem estar na mesma região da AWS que o cluster do Amazon Redshift. Para obter uma lista de regiões da AWS compatíveis, consulte Limitações do Amazon Redshift Spectrum.

Se o caminho especificar uma pasta ou bucket, por exemplo `'s3://amzn-s3-demo-bucket/custdata/'`, o Redshift Spectrum fará a varredura dos arquivos na pasta ou bucket especificado e em todas as subpastas. O Redshift Spectrum ignora os arquivos ocultos e os arquivos que começam com um ponto ou um sublinhado.

Se o caminho especificar um arquivo manifesto, o argumento `'s3://bucket/manifest_file'` deverá fazer referência explícita a um único arquivo, por exemplo, `'s3://amzn-s3-demo-bucket/manifest.txt'`. Ele não pode fazer referência a um prefixo de chaves.

O manifesto é um arquivo de texto em formato JSON que lista o URL de cada arquivo a ser carregado a partir do Amazon S3 e o tamanho do arquivo em bytes. O URL inclui o nome do bucket e o caminho de objeto completo do arquivo. Os arquivos especificados no manifesto podem estar em buckets diferentes, mas todos os buckets devem estar na mesma região da AWS que o cluster do Amazon Redshift. Se for listado duas vezes, o arquivo será carregado duas vezes. O exemplo a seguir mostra o JSON de um manifesto que carrega três arquivos.

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket1/custdata.1", "meta": { "content_length":
 5956875 } },
    {"url":"s3://amzn-s3-demo-bucket1/custdata.2", "meta": { "content_length":
 5997091 } },
    {"url":"s3://amzn-s3-demo-bucket2/custdata.1", "meta": { "content_length":
 5978675 } }
  ]
```

```
}
```

Você pode tornar a inclusão de um arquivo específico obrigatória. Para fazer isso, inclua uma opção `mandatory` em nível de arquivo no manifesto. Ao consultar uma tabela externa com um ficheiro obrigatório faltando, a instrução SELECT falha. Verifique se todos os arquivos incluídos na definição da tabela externa estão presentes. Se nem todos estiverem presentes, um erro será exibido mostrando o primeiro arquivo obrigatório que não foi encontrado. O exemplo a seguir mostra o JSON para um manifesto com a opção `mandatory` definida como `true`.

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket1/custdata.1", "mandatory":true, "meta":
 { "content_length": 5956875 } },
    {"url":"s3://amzn-s3-demo-bucket1/custdata.2", "mandatory":false, "meta":
 { "content_length": 5997091 } },
    {"url":"s3://amzn-s3-demo-bucket2/custdata.1", "meta": { "content_length":
 5978675 } }
  ]
}
```

Para fazer referência a arquivos criados usando UNLOAD, use o manifesto criado usando UNLOAD com o parâmetro MANIFEST. O arquivo manifesto é compatível com um arquivo manifesto para COPY do Amazon S3, mas utiliza chaves diferentes. Chaves que não são usadas são ignoradas.

TABLE PROPERTIES ('property_name'='property_value' [, ...])

Uma cláusula que estabelece a definição da tabela para propriedades da tabela.

> **Note**
>
> As propriedades de tabela fazem distinção entre maiúsculas e minúsculas.

'compression_type'='valor'

Uma propriedade que define o tipo de compactação a ser usado se o nome de arquivo não contiver uma extensão. Se você definir essa propriedade e houver uma extensão de arquivo, a extensão será ignorada e o valor definido pela propriedade será usado. Os valores válidos para o tipo de compactação são os seguintes:

- bzip2

- gzip
- nenhuma
- snappy

'data_cleansing_enabled'='true / false’

Essa propriedade define se o tratamento de dados está ativado para a tabela. Quando 'data_cleansing_enabled' está definido como true, o tratamento de dados está ativado para a tabela. Quando 'data_cleansing_enabled' está definido como false, o tratamento de dados está desativado para a tabela. A seguir, há uma lista das propriedades de tratamento de dados em nível de tabela controladas por essa propriedade:

- column_count_mismatch_handling
- invalid_char_handling
- numeric_overflow_handling
- replacement_char
- surplus_char_handling

Para obter exemplos, consulte Exemplos de tratamento de dados.

'invalid_char_handling'='valor'

Especifica a ação a ser realizada quando os resultados da consulta contêm valores de caracteres UTF-8 inválidos. Você pode especificar as seguintes ações:

DESATIVADA

Não trata os caracteres inválidos.

FAIL

Cancela as consultas que retornam dados contendo valores UTF-8 inválidos.

SET_TO_NULL

Substitui os valores UTF-8 inválidos por null.

DROP_ROW

Substitui todos os valores da linha por null.

REPLACE

Substitui o caractere inválido pelo caractere de substituição especificado usando `replacement_char`.

'replacement_char'='caractere’

Especifica o caractere de substituição a ser usado quando você define `invalid_char_handling` como `REPLACE`.

'numeric_overflow_handling'='valor’

Especifica a ação a ser realizada quando os dados ORC contêm um inteiro (por exemplo, BIGINT ou int64) que é maior que a definição da coluna (por exemplo, SMALLINT ou int16). Você pode especificar as seguintes ações:

DESATIVADA

O tratamento de caracteres inválidos é desativado.

FAIL

Cancelar a consulta quando os dados incluírem caracteres inválidos.

SET_TO_NULL

Definir os caracteres inválidos como null.

DROP_ROW

Definir todos os valores da linha como null.

'surplus_bytes_handling'='value'

Especifica como lidar com os dados sendo carregados que excederem o comprimento do tipo de dado definido para colunas contendo dados VARBYTE. Por padrão, o Redshift Spectrum define o valor como null para dados que excedem a largura da coluna.

Você pode especificar as seguintes ações a serem realizadas quando a consulta retorna dados que excedem o comprimento do tipo de dado:

SET_TO_NULL

Substitui os dados que excedem a largura da coluna por null.

DESATIVADA

Não lida com excesso de bytes.

FAIL

Cancela as consultas que retornam dados que excedem a largura da coluna.

DROP_ROW

Elimine todas as linhas que contêm dados que excedam a largura da coluna.

TRUNCATE

Remove os caracteres que excedem o número máximo de caracteres definido para a coluna.

'surplus_char_handling'='valor'

Especifica como lidar com os dados sendo carregados que excederem o comprimento do tipo de dados definido para colunas contendo VARCHAR, CHAR ou dados em string. Por padrão, o Redshift Spectrum define o valor como null para dados que excedem a largura da coluna.

Você pode especificar as seguintes ações a serem realizadas quando a consulta retorna dados que excedem a largura da coluna:

SET_TO_NULL

Substitui os dados que excedem a largura da coluna por null.

DESATIVADA

Não trata de caracteres em excesso.

FAIL

Cancela as consultas que retornam dados que excedem a largura da coluna.

DROP_ROW

Substitui todos os valores da linha por null.

TRUNCATE

Remove os caracteres que excedem o número máximo de caracteres definido para a coluna.

'column_count_mismatch_handling'='value’

Identifica se o arquivo contém um número menor ou maior de valores para uma linha do que o de colunas especificado na definição da tabela externa. Essa propriedade só está disponível para um formato de arquivo de texto não compactado. Você pode especificar as seguintes ações:

DESATIVADA

O tratamento de incompatibilidade de contagem de colunas está desativado.

FAIL

Há falha na consulta se for detectada incompatibilidade na contagem de colunas.

SET_TO_NULL

Preencha os valores ausentes com NULL e ignore os valores adicionais em cada linha.

DROP_ROW

Elimine todas as linhas que contêm erro de incompatibilidade de contagem de colunas na verificação.

'numRows'='row_count'

Uma propriedade que define o valor de numRows para a definição da tabela. Para atualizar de maneira explícita as estatísticas de uma tabela externa, defina a propriedade numRows de maneira a indicar o tamanho da tabela. O Amazon Redshift não analisa as tabelas externas para gerar as estatísticas das tabelas que o otimizador de consultas utiliza para gerar um plano de consulta. Se as estatísticas de tabelas não estiverem definidas para uma tabela externa, o Amazon Redshift gera um plano de execução de consulta com base na suposição de que as tabelas externas são maiores e as tabelas locais são menores.

'skip.header.line.count'='line_count'

Uma propriedade que define o número de linhas a serem ignoradas no início de cada arquivo de origem.

'serialization.null.format'=' '

Uma propriedade que especifica o Spectrum deve retornar um valor NULL quando houver uma correspondência exata com o texto fornecido em um campo.

'orc.schema.resolution'='mapping_type'

Uma propriedade que define o tipo de mapeamento de coluna para tabelas que usam o formato de dados ORC. Essa propriedade é ignorada para outros formatos de dados.

Os valores válidos para o tipo de mapeamento de coluna são os seguintes:

- name
- position

Se a propriedade orc.schema.resolution for omitida, as colunas serão mapeadas por nome por padrão. Se orc.schema.resolution estiver definido como qualquer valor diferente de 'name'

ou 'position', as colunas serão mapeadas por posição. Para obter mais informações sobre o mapeamento de colunas, consulte Mapeamento de colunas de tabela externa para colunas do ORC.

> **Note**
>
> O comando COPY mapeia para arquivos de dados ORC apenas por posição. A propriedade de tabela orc.schema.resolution não tem efeito sobre o comportamento do comando COPY.

'write.parallel'='on / off'

Uma propriedade que define se CREATE EXTERNAL TABLE AS deve gravar dados em paralelo. Por padrão, CREATE EXTERNAL TABLE AS grava dados em paralelo em vários arquivos, de acordo com o número de fatias no cluster. A opção padrão é ativado. Quando 'write.parallel' está definido como desativado, CREATE EXTERNAL TABLE AS grava em um ou mais arquivos de dados serialmente no Amazon S3. Essa propriedade de tabela também se aplica a qualquer instrução INSERT subsequente na mesma tabela externa.

‘write.maxfilesize.mb’=‘size’

Uma propriedade que define o tamanho máximo (em MB) de cada arquivo gravado no Amazon S3 por CREATE EXTERNAL TABLE AS. O tamanho deve ser um número inteiro entre 5 e 6.200. O tamanho máximo padrão do arquivo é 6.200 MB. Essa propriedade de tabela também se aplica a qualquer instrução INSERT subsequente na mesma tabela externa.

‘write.kms.key.id’=‘value’

É possível especificar uma chave AWS Key Management Service para habilitar a criptografia do lado do servidor (SSE) para objetos do Amazon S3, onde value é uma das seguintes ações:

- `auto` para usar o chave padrão do AWS KMS armazenada no bucket do Amazon S3.
- kms-keyque você especifica para criptografar dados.

select_statement

Uma instrução que insere uma ou mais linhas na tabela externa ao definir qualquer consulta. Todas as linhas que a consulta gera são gravadas no Amazon S3 no formato de texto ou Parquet com base na definição da tabela.

Exemplos

Uma coleção de exemplos está disponível em Exemplos.

Observações de uso

Este tópico contém notas de uso do CREATE EXTERNAL TABLE. Não é possível exibir os detalhes das tabelas do Amazon Redshift Spectrum usando os mesmos recursos das tabelas padrão do Amazon Redshift, como PG_TABLE_DEF, STV_TBL_PERM, PG_CLASS, ou information_schema. Caso sua ferramenta de business intelligence ou análise não reconheça as tabelas externas do Redshift Spectrum, configure sua aplicação para consultar SVV_EXTERNAL_TABLES e SVV_EXTERNAL_COLUMNS.

CREATE EXTERNAL TABLE AS

Em alguns casos, é possível executar o comando CREATE EXTERNAL TABLE AS em um catálogo de dados do AWS Glue, em um catálogo externo do AWS Lake Formation ou em um metastore do Apache Hive. Nesses casos, use uma função do AWS Identity and Access Management (IAM) para criar o esquema externo. Essa função do IAM deve conter as permissões de leitura e gravação no Amazon S3.

Se você usar um catálogo do Lake Formation, a função do IAM deve ter a permissão para criar tabelas no catálogo. Nesse caso, ela também deve ter a permissão do local do data lake no caminho de destino do Amazon S3. Essa função do IAM se torna proprietária da nova tabela do AWS Lake Formation.

Para garantir que os nomes dos arquivos sejam exclusivos, o Amazon Redshift usa, por padrão, o formato a seguir para o nome de cada arquivo carregado no Amazon S3.

*<date>*_*<time>*_*<microseconds>*_*<query_id>*_*<slice-number>*_part_*<part-number>*.*<format>*.

Um exemplo é `20200303_004509_810669_1007_0001_part_00.parquet`.

Considere o seguinte ao executar o comando CREATE EXTERNAL TABLE AS:

- O local do Amazon S3 deve estar vazio.
- O Amazon Redshift oferece suporte somente aos formatos PARQUET e TEXTFILE ao usar a cláusula STORED AS.
- Não é necessário criar uma lista de definições de coluna. Os nomes de colunas e os tipos de dados de coluna da nova tabela externa são derivados diretamente da consulta SELECT.

- Não é necessário definir o tipo de dados da coluna de partição na cláusula PARTITIONED BY. Se você especificar uma chave de partição, o nome dessa coluna deverá existir no resultado da consulta SELECT. Ao ter várias colunas de partição, a ordem na consulta SELECT não importa. O Amazon Redshift usa a ordem definida na cláusula PARTITIONED BY para criar a tabela externa.
- O Amazon Redshift automaticamente particiona arquivos de saída em pastas de partição com base nas chaves-valor de partição. Por padrão, o Amazon Redshift remove as colunas de partição dos arquivos de saída.
- Não há suporte para a cláusula LINES TERMINATED BY 'delimiter'.
- Não há suporte para a cláusula ROW FORMAT SERDE 'serde_name'.
- Não há suporte para o uso de arquivos de manifesto. Dessa forma, não é possível definir a cláusula LOCATION em um arquivo manifesto no Amazon S3.
- O Amazon Redshift automaticamente atualiza a propriedade da tabela “numRows” no final do comando.
- A propriedade da tabela 'compression_type' aceita somente 'none' ou 'snappy' para o formato de arquivo PARQUET.
- O Amazon Redshift não permite a cláusula LIMIT na consulta SELECT externa. Em vez disso, você pode usar uma cláusula LIMIT aninhada.
- É possível usar STL_UNLOAD_LOG para rastrear os arquivos que são gravados no Amazon S3 por cada operação CREATE EXTERNAL TABLE AS.

Permissões para criar e consultar tabelas externas

Para criar tabelas externas, verifique se você é o proprietário do esquema externo ou um superusuário. Para transferir a propriedade de um esquema externo, use ALTER SCHEMA. O exemplo a seguir altera o proprietário do esquema `spectrum_schema` para `newowner`.

```
alter schema spectrum_schema owner to newowner;
```

Para executar uma consulta do Redshift Spectrum, você precisa das seguintes permissões:

- Permissão de uso no esquema
- Permissão para criar tabelas temporárias no banco de dados atual

O exemplo a seguir concede permissão de uso no esquema `spectrum_schema` para o grupo de usuários `spectrumusers`.

```
grant usage on schema spectrum_schema to group spectrumusers;
```

O exemplo a seguir concede permissão temporária no banco de dados `spectrumdb` para o grupo de usuários `spectrumusers`.

```
grant temp on database spectrumdb to group spectrumusers;
```

Pseudocolunas

Por padrão, o Amazon Redshift cria tabelas externas com as pseudocolunas $path e $size. Selecione essas colunas para exibir o caminho que levará aos arquivos de dados no Amazon S3 e o tamanho dos arquivos de dados em cada linha retornada por uma consulta. Os nomes de coluna $path e $size devem ser delimitados por aspas duplas. A cláusula SELECT * não retornará as pseudocolunas. Você deve incluir explicitamente os nomes de coluna $path e $size na consulta, como mostra o exemplo a seguir.

```
select "$path", "$size"
from spectrum.sales_part
where saledate = '2008-12-01';
```

Você pode desabilitar a criação de pseudocolunas em uma sessão. Basta definir o parâmetro de configuração spectrum_enable_pseudo_columns como false.

> **Important**
>
> A seleção de $size ou $path gera cobranças porque o Redshift Spectrum verifica os arquivos de dados no Amazon S3 para determinar o tamanho do conjunto de resultados. Para obter mais informações, consulte Preço do Amazon Redshift.

Para tabelas do Iceberg v3, duas pseudocolunas adicionais, _row_id e _last_updated_sequence_number, estão disponíveis. Para obter mais informações, consulte the section called “Linhagem de linhas”.

Definir opções de tratamento de dados

Você pode definir parâmetros de tabela para especificar o tratamento de entrada para dados que estão sendo consultados em tabelas externas, incluindo:

- Caracteres excedentes em colunas contendo VARCHAR, CHAR e dados em string. Para obter mais informações, consulte a propriedade de tabela externa `surplus_char_handling`.
- Caracteres inválidos em colunas contendo VARCHAR, CHAR e dados em string. Para obter mais informações, consulte a propriedade de tabela externa `invalid_char_handling`.
- Caractere de substituição a ser usado quando você especificar REPLACE como a propriedade de tabela externa `invalid_char_handling`.
- Tratamento de transbordamento de conversão em colunas contendo dados inteiros e decimais. Para obter mais informações, consulte a propriedade de tabela externa `numeric_overflow_handling`.
- Surplus_bytes_handling para especificar o tratamento de entradas para bytes excedentes em colunas que contêm dados varbyte. Para obter mais informações, consulte a propriedade de tabela externa `surplus_bytes_handling`.

Exemplos

O exemplo a seguir cria uma tabela chamada SALES no esquema externo do Amazon Redshift denominado `spectrum`. Os dados estão em arquivos de texto delimitados por tabulação. A cláusula TABLE PROPERTIES define a propriedade numRows como 170.000 linhas.

Dependendo da identidade usada para executar CREATE EXTERNAL TABLE, pode haver permissões do IAM que você precisa configurar. Como prática recomendada, anexe políticas de permissões a um perfil do IAM e, depois, atribua-as a usuários e grupos, conforme necessário. Para obter mais informações, consulte Gerenciamento de identidade e acesso no Amazon Redshift.

```
create external table spectrum.sales(
salesid integer,
listid integer,
sellerid integer,
buyerid integer,
eventid integer,
saledate date,
qtysold smallint,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp)
row format delimited
fields terminated by '\t'
stored as textfile
```

```
location 's3://redshift-downloads/tickit/spectrum/sales/'
table properties ('numRows'='170000');
```

O exemplo a seguir cria uma tabela que usa o JsonSerDe para fazer referência aos dados em formato JSON.

```
create external table spectrum.cloudtrail_json (
event_version int,
event_id bigint,
event_time timestamp,
event_type varchar(10),
awsregion varchar(20),
event_name varchar(max),
event_source varchar(max),
requesttime timestamp,
useragent varchar(max),
recipientaccountid bigint)
row format serde 'org.openx.data.jsonserde.JsonSerDe'
with serdeproperties (
'dots.in.keys' = 'true',
'mapping.requesttime' = 'requesttimestamp'
) location 's3://amzn-s3-demo-bucket/json/cloudtrail';
```

O exemplo de CREATE EXTERNAL TABLE AS a seguir cria uma tabela externa não particionada. Depois, ele grava o resultado da consulta SELECT como Apache Parquet no local de destino do Amazon S3.

```
CREATE EXTERNAL TABLE spectrum.lineitem
STORED AS parquet
LOCATION 'S3://amzn-s3-demo-bucket/cetas/lineitem/'
AS SELECT * FROM local_lineitem;
```

O exemplo a seguir cria uma tabela externa particionada e inclui as colunas de partição na consulta SELECT.

```
CREATE EXTERNAL TABLE spectrum.partitioned_lineitem
PARTITIONED BY (l_shipdate, l_shipmode)
STORED AS parquet
LOCATION 'S3://amzn-s3-demo-bucket/cetas/partitioned_lineitem/'
AS SELECT l_orderkey, l_shipmode, l_shipdate, l_partkey FROM local_table;
```

Para obter uma lista de bancos de dados existentes no catálogo de dados externo, consulte a exibição de sistema SVV_EXTERNAL_DATABASES.

```
select eskind,databasename,esoptions from svv_external_databases order by databasename;
```

```
eskind | databasename | esoptions
-------+--------------
+-------------------------------------------------------------------------------
     1 | default      | {"REGION":"us-
west-2","IAM_ROLE":"arn:aws:iam::123456789012:role/mySpectrumRole"}
     1 | sampledb     | {"REGION":"us-
west-2","IAM_ROLE":"arn:aws:iam::123456789012:role/mySpectrumRole"}
     1 | spectrumdb   | {"REGION":"us-
west-2","IAM_ROLE":"arn:aws:iam::123456789012:role/mySpectrumRole"}
```

Para visualizar detalhes das tabelas externas, consulte as exibições SVV_EXTERNAL_TABLES e SVV_EXTERNAL_COLUMNS do sistema.

O exemplo a seguir consulta a exibição SVV_EXTERNAL_TABLES.

```
select schemaname, tablename, location from svv_external_tables;
```

```
schemaname | tablename            | location
-----------+----------------------
+-----------------------------------------------------
spectrum   | sales                | s3://redshift-downloads/tickit/spectrum/sales
spectrum   | sales_part           | s3://redshift-downloads/tickit/spectrum/
sales_partition
```

O exemplo a seguir consulta a exibição SVV_EXTERNAL_COLUMNS.

```
select * from svv_external_columns where schemaname like 'spectrum%' and tablename
 ='sales';
```

```
schemaname | tablename | columnname | external_type | columnnum | part_key
-----------+-----------+------------+---------------+-----------+---------
spectrum   | sales     | salesid    | int           |         1 |        0
spectrum   | sales     | listid     | int           |         2 |        0
spectrum   | sales     | sellerid   | int           |         3 |        0
```

```
spectrum   | sales     | buyerid    | int           |         4 |        0
spectrum   | sales     | eventid    | int           |         5 |        0
spectrum   | sales     | saledate   | date          |         6 |        0
spectrum   | sales     | qtysold    | smallint      |         7 |        0
spectrum   | sales     | pricepaid  | decimal(8,2)  |         8 |        0
spectrum   | sales     | commission | decimal(8,2)  |         9 |        0
spectrum   | sales     | saletime   | timestamp     |        10 |        0
```

Para exibir as partições de tabela, use a consulta a seguir.

```
select schemaname, tablename, values, location
from svv_external_partitions
where tablename = 'sales_part';
```

```
schemaname | tablename  | values         | location
-----------+------------+----------------
+-------------------------------------------------------------------------
spectrum   | sales_part | ["2008-01-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-01
spectrum   | sales_part | ["2008-02-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-02
spectrum   | sales_part | ["2008-03-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-03
spectrum   | sales_part | ["2008-04-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-04
spectrum   | sales_part | ["2008-05-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-05
spectrum   | sales_part | ["2008-06-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-06
spectrum   | sales_part | ["2008-07-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-07
spectrum   | sales_part | ["2008-08-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-08
spectrum   | sales_part | ["2008-09-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-09
spectrum   | sales_part | ["2008-10-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-10
spectrum   | sales_part | ["2008-11-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-11
spectrum   | sales_part | ["2008-12-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-12
```

O exemplo a seguir retorna o tamanho total de arquivos de dados relacionados de uma tabela externa.

```
select distinct "$path", "$size"
   from spectrum.sales_part;

 $path                                                                      | $size
----------------------------------------------------------------------------+-------
s3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-01/ |  1616
s3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-02/ |  1444
s3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-02/ |  1444
```

Exemplos de particionamento

Para criar uma tabela externa particionada por data, execute o seguinte comando.

```
create external table spectrum.sales_part(
salesid integer,
listid integer,
sellerid integer,
buyerid integer,
eventid integer,
dateid smallint,
qtysold smallint,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp)
partitioned by (saledate date)
row format delimited
fields terminated by '|'
stored as textfile
location 's3://redshift-downloads/tickit/spectrum/sales_partition/'
table properties ('numRows'='170000');
```

Para adicionar as partições, execute os seguintes comandos ALTER TABLE.

```
alter table spectrum.sales_part
add if not exists partition (saledate='2008-01-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-01/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-02-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-02/';
```

```
alter table spectrum.sales_part
add if not exists partition (saledate='2008-03-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-03/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-04-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-04/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-05-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-05/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-06-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-06/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-07-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-07/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-08-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-08/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-09-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-09/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-10-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-10/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-11-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-11/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-12-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-12/';
```

Para selecionar dados na tabela particionada, execute a consulta a seguir.

```
select top 10 spectrum.sales_part.eventid, sum(spectrum.sales_part.pricepaid)
from spectrum.sales_part, event
where spectrum.sales_part.eventid = event.eventid
  and spectrum.sales_part.pricepaid > 30
  and saledate = '2008-12-01'
group by spectrum.sales_part.eventid
order by 2 desc;
```

```
eventid | sum
--------+---------
```

```
   914 | 36173.00
  5478 | 27303.00
  5061 | 26383.00
  4406 | 26252.00
  5324 | 24015.00
  1829 | 23911.00
  3601 | 23616.00
  3665 | 23214.00
  6069 | 22869.00
  5638 | 22551.00
```

Para visualizar as partições da tabela externa, consulte a exibição do sistema SVV_EXTERNAL_PARTITIONS.

```
select schemaname, tablename, values, location from svv_external_partitions
where tablename = 'sales_part';
```

```
schemaname | tablename  | values         | location
-----------+------------+----------------
+-------------------------------------------------
spectrum   | sales_part | ["2008-01-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-01
spectrum   | sales_part | ["2008-02-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-02
spectrum   | sales_part | ["2008-03-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-03
spectrum   | sales_part | ["2008-04-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-04
spectrum   | sales_part | ["2008-05-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-05
spectrum   | sales_part | ["2008-06-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-06
spectrum   | sales_part | ["2008-07-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-07
spectrum   | sales_part | ["2008-08-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-08
spectrum   | sales_part | ["2008-09-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-09
spectrum   | sales_part | ["2008-10-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-10
spectrum   | sales_part | ["2008-11-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-11
```

```
spectrum   | sales_part | ["2008-12-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-12
```

Exemplos de formato de linha

Este é um exemplo da especificação dos parâmetros ROW FORMAT SERDE para arquivos de dados armazenados no formato AVRO.

```
create external table spectrum.sales(salesid int, listid int, sellerid int,
 buyerid int, eventid int, dateid int, qtysold int, pricepaid decimal(8,2), comment
 VARCHAR(255))
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.avro.AvroSerDe'
WITH SERDEPROPERTIES ('avro.schema.literal'='{\"namespace\": \"dory.sample\",\"name\":
 \"dory_avro\",\"type\": \"record\", \"fields\": [{\"name\":\"salesid\", \"type\":\"int
\"},
{\"name\":\"listid\", \"type\":\"int\"},
{\"name\":\"sellerid\", \"type\":\"int\"},
{\"name\":\"buyerid\", \"type\":\"int\"},
{\"name\":\"eventid\",\"type\":\"int\"},
{\"name\":\"dateid\",\"type\":\"int\"},
{\"name\":\"qtysold\",\"type\":\"int\"},
{\"name\":\"pricepaid\", \"type\": {\"type\": \"bytes\", \"logicalType\": \"decimal\",
 \"precision\": 8, \"scale\": 2}}, {\"name\":\"comment\",\"type\":\"string\"}]}')
STORED AS AVRO
location 's3://amzn-s3-demo-bucket/avro/sales' ;
```

O exemplo a seguir mostra como especificar os parâmetros ROW FORMAT SERDE usando RegEx.

```
create external table spectrum.types(
cbigint bigint,
cbigint_null bigint,
cint int,
cint_null int)
row format serde 'org.apache.hadoop.hive.serde2.RegexSerDe'
with serdeproperties ('input.regex'='([^\\x01]+)\\x01([^\\x01]+)\\x01([^\\x01]+)\
\x01([^\\x01]+)')
stored as textfile
location 's3://amzn-s3-demo-bucket/regex/types';
```

O exemplo a seguir mostra como especificar os parâmetros ROW FORMAT SERDE usando Grok.

```
create external table spectrum.grok_log(
```

```
timestamp varchar(255),
pid varchar(255),
loglevel varchar(255),
progname varchar(255),
message varchar(255))
row format serde 'com.amazonaws.glue.serde.GrokSerDe'
with serdeproperties ('input.format'='[DFEWI], \\[%{TIMESTAMP_ISO8601:timestamp} #
%{POSINT:pid:int}\\] *(?<loglevel>:DEBUG|FATAL|ERROR|WARN|INFO) -- +%{DATA:progname}:
 %{GREEDYDATA:message}')
stored as textfile
location 's3://DOC-EXAMPLE-BUCKET/grok/logs';
```

A tabela a seguir mostra um exemplo de definição de um log de acesso ao servidor do Amazon S3 em um bucket do S3. Use o Redshift Spectrum para consultar logs de acesso do Amazon S3.

```
CREATE EXTERNAL TABLE spectrum.mybucket_s3_logs(
bucketowner varchar(255),
bucket varchar(255),
requestdatetime varchar(2000),
remoteip varchar(255),
requester varchar(255),
requested varchar(255),
operation varchar(255),
key varchar(255),
requesturi_operation varchar(255),
requesturi_key varchar(255),
requesturi_httpprotoversion varchar(255),
httpstatus varchar(255),
errorcode varchar(255),
bytessent bigint,
objectsize bigint,
totaltime varchar(255),
turnaroundtime varchar(255),
referrer varchar(255),
useragent varchar(255),
versionid varchar(255)
)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'
WITH SERDEPROPERTIES (
'input.regex' = '([^ ]*) ([^ ]*) \\[(.*?)\\] ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*)
 \"([^ ]*)\\s*([^ ]*)\\s*([^ ]*)\" (- |[^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*)
 ([^ ]*) (\"[^\"]*\") ([^ ]*).*$')
LOCATION 's3://amzn-s3-demo-bucket/s3logs’;
```

Este é um exemplo da especificação dos parâmetros ROW FORMAT SERDE para dados no formato ION.

```
CREATE EXTERNAL TABLE tbl_name (columns)
ROW FORMAT SERDE 'com.amazon.ionhiveserde.IonHiveSerDe'
STORED AS
INPUTFORMAT 'com.amazon.ionhiveserde.formats.IonInputFormat'
OUTPUTFORMAT 'com.amazon.ionhiveserde.formats.IonOutputFormat'
LOCATION 's3://amzn-s3-demo-bucket/prefix'
```

Exemplos de tratamento de dados

Os exemplos a seguir acessam o arquivo: spi_global_rankings.csv. Você pode carregar o arquivo `spi_global_rankings.csv` em um bucket do Amazon S3 para experimentar esses exemplos.

O exemplo a seguir cria o esquema externo `schema_spectrum_uddh` e o banco de dados externo `spectrum_db_uddh`. Para `aws-account-id`, insira o ID da conta da AWS e, para `role-name`, insira seu nome de função do Redshift Spectrum.

```
create external schema schema_spectrum_uddh
from data catalog
database 'spectrum_db_uddh'
iam_role 'arn:aws:iam::aws-account-id:role/role-name'
create external database if not exists;
```

O exemplo a seguir cria uma tabela externa `soccer_league` no esquema externo `schema_spectrum_uddh`.

```
CREATE EXTERNAL TABLE schema_spectrum_uddh.soccer_league
(
  league_rank smallint,
  prev_rank   smallint,
  club_name   varchar(15),
  league_name varchar(20),
  league_off  decimal(6,2),
  league_def  decimal(6,2),
  league_spi  decimal(6,2),
  league_nspi integer
)
ROW FORMAT DELIMITED
```

```
    FIELDS TERMINATED BY ','
    LINES TERMINATED BY '\n\l'
stored as textfile
LOCATION 's3://spectrum-uddh/league/'
table properties ('skip.header.line.count'='1');
```

Confira o número de linhas da tabela `soccer_league`.

```
select count(*) from schema_spectrum_uddh.soccer_league;
```

Os números de linhas são exibidos.

```
count
645
```

A consulta a seguir exibe os 10 principais clubes. Como clube `Barcelona` tem um caractere inválido na string, um NULL é exibido para o nome.

```
select league_rank,club_name,league_name,league_nspi
from schema_spectrum_uddh.soccer_league
where league_rank between 1 and 10;
```

```
league_rank club_name league_name   league_nspi
1  Manchester City Barclays Premier Lea  34595
2  Bayern Munich German Bundesliga  34151
3  Liverpool Barclays Premier Lea  33223
4  Chelsea  Barclays Premier Lea  32808
5  Ajax  Dutch Eredivisie  32790
6  Atletico  Madrid Spanish Primera Divi 31517
7  Real Madrid Spanish Primera Divi  31469
8  NULL         Spanish Primera Divi            31321
9  RB Leipzig German Bundesliga  31014
10  Paris Saint-Ger French Ligue 1   30929
```

O exemplo a seguir altera a tabela `soccer_league` para especificar as propriedades `invalid_char_handling`, `replacement_char` e `data_cleansing_enabled` da tabela externa e inserir um ponto de interrogação (?) como substituto de caracteres inesperados.

```
alter  table schema_spectrum_uddh.soccer_league
```

```
set table properties
 ('invalid_char_handling'='REPLACE','replacement_char'='?','data_cleansing_enabled'='true');
```

O exemplo a seguir consulta a tabela `soccer_league` para times com classificação de 1 a 10.

```
select league_rank,club_name,league_name,league_nspi
from schema_spectrum_uddh.soccer_league
where league_rank between 1 and 10;
```

Como as propriedades da tabela foram alteradas, os resultados mostram os dez principais clubes, com o ponto de interrogação (?) como caractere substituto na oitava linha para o clube `Barcelona`.

```
league_rank club_name league_name  league_nspi
1  Manchester City Barclays Premier Lea 34595
2  Bayern Munich German Bundesliga 34151
3  Liverpool Barclays Premier Lea 33223
4  Chelsea  Barclays Premier Lea 32808
5  Ajax  Dutch Eredivisie 32790
6  Atletico Madrid Spanish Primera Divi 31517
7  Real Madrid Spanish Primera Divi 31469
8  Barcel?na Spanish Primera Divi 31321
9  RB Leipzig German Bundesliga 31014
10  Paris Saint-Ger French Ligue 1  30929
```

O exemplo a seguir altera a tabela `soccer_league` para especificar que as propriedades `invalid_char_handling` da tabela externa descartem as linhas com caracteres inesperados.

```
alter table schema_spectrum_uddh.soccer_league
set table properties
 ('invalid_char_handling'='DROP_ROW','data_cleansing_enabled'='true');
```

O exemplo a seguir consulta a tabela `soccer_league` para times com classificação de 1 a 10.

```
select league_rank,club_name,league_name,league_nspi
from schema_spectrum_uddh.soccer_league
where league_rank between 1 and 10;
```

Os resultados exibem os principais clubes, sem incluir a oitava linha para o clube `Barcelona`.

```
league_rank   club_name         league_name            league_nspi
```

```
1            Manchester City   Barclays Premier Lea   34595
2            Bayern Munich     German Bundesliga      34151
3            Liverpool         Barclays Premier Lea   33223
4            Chelsea           Barclays Premier Lea   32808
5            Ajax              Dutch Eredivisie       32790
6            Atletico Madrid   Spanish Primera Divi   31517
7            Real Madrid       Spanish Primera Divi   31469
9            RB Leipzig        German Bundesliga      31014
10           Paris Saint-Ger   French Ligue 1         30929
```

CREATE EXTERNAL VIEW

O recurso de versão prévia das visualizações do Data Catalog só está disponível nas regiões a seguir.

- Leste dos EUA (Ohio) (us-east-2)
- Leste dos EUA (Norte da Virgínia) (us-east-1)
- Oeste dos EUA (N. da Califórnia) (us-west-1)
- Ásia Pacific (Tóquio) (ap-northeast-1)
- Europa (Irlanda) (eu-west-1)
- Europa (Estocolmo) (eu-north-1)

Cria uma exibição do Data Catalog. As exibições do Data Catalog são um esquema de exibição única que funciona com outros mecanismos SQL, como Amazon Athena e Amazon EMR. Você pode consultar a exibição no mecanismo de sua escolha. Consulte mais informações sobre visualizações do Catálogo de Dados em Criação de uma visualização do Catálogo de Dados.

Sintaxe

```
CREATE EXTERNAL VIEW schema_name.view_name [ IF NOT EXISTS ]
{catalog_name.schema_name.view_name | awsdatacatalog.dbname.view_name |
 external_schema_name.view_name}
AS query_definition;
```

Parâmetros

schema_name.view_name

O esquema anexado ao banco de dados do AWS Glue, seguido do nome da exibição.

PROTECTED

Especifica que o comando CREATE EXTERNAL VIEW só deverá ser concluído se a consulta dentro da query_definition puder ser concluída com êxito.

IF NOT EXISTS

Cria a exibição se ela ainda não existir.

catalog_name.schema_name.view_name | awsdatacatalog.dbname.view_name | external_schema_name.view_name

A notação do esquema a ser usado durante a criação da exibição. Você pode especificar o uso do AWS Glue Data Catalog, um banco de dados do Glue criado por você, ou um esquema externo também criado por você. Consulte CREATE DATABASE e CREATE EXTERNAL SCHEMA para obter mais informações.

query_definition

A definição da consulta SQL executada pelo Amazon Redshift para alterar a exibição.

Exemplos

O exemplo a seguir cria uma exibição do Data Catalog chamada sample_schema.glue_data_catalog_view.

```
CREATE EXTERNAL PROTECTED VIEW sample_schema.glue_data_catalog_view IF NOT EXISTS
AS SELECT * FROM sample_database.remote_table "remote-table-name";
```

CREATE FUNCTION

Crie uma nova função definida pelo usuário (UDF) escalar usando uma cláusula SQL SELECT ou um programa Python.

Para ter mais informações e exemplos, consulte Funções definidas pelo usuário (UDFs) no Amazon Redshift.

Privilégios obrigatórios

Você deve ter permissão de uma das seguintes formas para executar CREATE OR REPLACE FUNCTION:

- Para CREATE FUNCTION:

- O superusuário pode usar linguagens confiáveis e não confiáveis para criar funções.
- Usuários com o privilégio de CREATE [OR REPLACE] FUNCTION podem criar funções com linguagens confiáveis.

- Para REPLACE FUNCTION:
  - Superusuário
  - Usuários com o privilégio CREATE [OR REPLACE] FUNCTION
  - Proprietário da função

Sintaxe

```
CREATE [ OR REPLACE ] FUNCTION f_function_name
( { [py_arg_name  py_arg_data_type |
sql_arg_data_type } [ , ... ] ] )
RETURNS data_type
{ VOLATILE | STABLE | IMMUTABLE }
AS $$
  { python_program | SELECT_clause }
$$ LANGUAGE { plpythonu | sql }
```

Parâmetros

OR REPLACE

Especifica que, se uma função com os mesmos tipos de dados de argumento de nome e entrada, ou assinatura, como esta já existir, a função existente será substituída. Você só pode substituir uma função por uma nova função que defina um conjunto idêntico de tipos de dados. É preciso ser um superusuário para substituir uma função.

Se uma função for definida com o mesmo nome que uma função existente mas com uma assinatura diferente, uma nova função será criada. Em outras palavras, o nome da função fica sobrecarregado. Para obter mais informações, consulte Sobrecarga de nomes de função.

f_nome_função

Nome da função. Se você especificar um nome de esquema (como `myschema.myfunction`), a função será criada usando o esquema especificado. Caso contrário, a função será criada no esquema atual. Para obter mais informações sobre nomes válidos, consulte Nomes e identificadores.

Recomendamos que você prefixe os nomes de todos os UDF com `f_`. O Amazon Redshift reserva o prefixo `f_` para nomes de UDF, portanto, ao usar o prefixo `f_`, você garante que o nome da UDF não entre em conflito com nenhum nome de função SQL integrada existente ou futuro do Amazon Redshift. Para obter mais informações, consulte Evitar conflitos na nomeação da UDF.

Você pode definir mais de uma função com o mesmo nome se os tipos de dados dos argumentos de entrada forem diferentes. Em outras palavras, o nome da função fica sobrecarregado. Para obter mais informações, consulte Sobrecarga de nomes de função.

py_arg_name py_arg_data_type | sql_arg_data_type

Para uma UDF Python, uma lista de nomes de argumento de entrada e tipos de dados. Para uma UDF SQL, uma lista de tipos de dados sem nomes de argumento. Em uma UDF Python, consulte argumentos usando nomes de argumento. Em uma UDF SQL, consulte argumentos usando $1, $2, etc. com base na ordem dos argumentos na lista de argumentos.

Para uma UDF SQL, os tipos de dados de entrada e de retorno podem ser qualquer tipo de dados padrão do Amazon Redshift. Para uma UDF Python, os tipos de dados de entrada e retorno podem ser SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE PRECISION, BOOLEAN, CHAR, VARCHAR, DATE ou TIMESTAMP. Além disso, as funções definidas pelo usuário (UDFs) do Python são compatíveis com um tipo de dados ANYELEMENT. Ele é convertido automaticamente em um tipo de dados padrão com base no tipo de dados do argumento correspondente fornecido no tempo de execução. Se vários argumentos usarem ANYELEMENT, todos encontrarão o mesmo tipo de dados no tempo de execução, com base no primeiro argumento ANYELEMENT da lista. Para obter mais informações, consulte Tipos de dados da UDF Python e Tipos de dados.

Você pode especificar um máximo de 32 argumentos.

RETURNS tipo_dados

Tipo de dados do valor retornado pela função. O tipo de dados RETURNS pode ser qualquer tipo de dados padrão do Amazon Redshift. Além disso, as UDFs Python podem usar um tipo de dados ANYELEMENT, que é convertido automaticamente em um tipo de dados padrão com base no argumento fornecido no tempo de execução. Se você especificar ANYELEMENT para o tipo de dados de retorno, pelo menos um argumento deverá usar ANYELEMENT. O tipo de dados de retorno corresponde ao tipo de dados fornecido pelo argumento ANYELEMENT quando a função for chamada. Para obter mais informações, consulte Tipos de dados da UDF Python.

VOLATILE | STABLE | IMMUTABLE

Informa o otimizador de consulta sobre a volatilidade da função.

Você terá a melhor otimização se identificar sua função com a categoria mais restrita de volatilidade válida para ela. Entretanto, se a categoria for restrita demais, o otimizador poderá ignorar erroneamente algumas chamadas, o que resultará em um conjunto incorreto de resultados. Por ordem de nível de restrição, começando a partir da menos restrita, as categorias são:

- VOLATILE
- STABLE
- IMMUTABLE

VOLATILE

Dados os mesmos argumentos, a função pode retornar resultados diferentes em chamadas sucessivas, mesmo para as linhas em uma única instrução. O otimizador de consulta não pode fazer suposições sobre o comportamento de uma função volátil, portanto, uma consulta que usa uma função volátil deve reavaliar a função para cada linha de entrada.

STABLE

Dados os mesmos argumentos, a função certamente retorna os mesmos resultados para todas as linhas processadas em uma única instrução. A função pode retornar resultados diferentes quando chamada em instruções diferentes. Esta categoria permite que o otimizador aperfeiçoe várias chamadas da função em uma única instrução para uma única chamada da instrução.

IMMUTABLE

Dados os mesmos argumentos, a função sempre retorna o mesmo resultado. Quando uma consulta chama uma função `IMMUTABLE` com argumentos constantes, o otimizador faz uma avaliação prévia da função.

AS $$ instrução $$

Uma construção que delimita a instrução a ser executada. As palavras-chave literais `AS $$` e `$$` são obrigatórias.

O Amazon Redshift exige que você inclua a instrução na sua função usando um formato chamado cotação de dólar. Qualquer item incluído na instrução é aprovado exatamente como

está. Não é preciso inserir caracteres de escape antes dos caracteres especiais, pois o conteúdo da string é gravado literalmente.

Com a cotação de dólar, um par de símbolos de cifrão ($$) deve ser usado no início e no final da instrução para que ela seja executada, conforme exibido no exemplo a seguir.

```
$$ my statement $$
```

Como opção, entre cada par de símbolos de cifrão, você pode especificar uma string para ajudar a identificar a instrução. A string que você usa deve ser a mesma no início e no final dos pares de sinais. A sequência diferencia maiúsculas de minúsculas e segue as mesmas restrições de um identificador sem aspas, com a exceção de que não pode conter cifrões. O exemplo a seguir usa a string `test`.

```
$test$ my statement $test$
```

Para obter mais informações sobre o uso de símbolos de cifrão, consulte “Constantes de string entre símbolos de cifrão” em Estrutura lexical na documentação do PostgreSQL.

programa_python

Programa executável válido em Python que retorna um valor. A instrução que você aprovar com a função deve estar em conformidade com os requisitos de recuo como especificado no Guia de estilo do código Python no site do Python. Para obter mais informações, consulte Suporte da linguagem Python para UDFs.

SQL_clause

Uma cláusula SQL SELECT.

A cláusula SELECT não pode conter estes tipos de cláusulas:

- FROM
- INTO
- WHERE
- GROUP BY
- ORDER BY
- LIMIT

LANGUAGE { plpythonu | sql }

Para Python, especifique `plpythonu`. Para SQL, especifique `sql`. Você deve ter permissão de uso na linguagem para SQL ou plpythonu. Para obter mais informações, consulte Segurança e permissões de UDFs.

Observações de uso

Funções aninhadas

Você pode chamar outra função definida pelo usuário (UDF) SQL a partir de uma UDF SQL. A função aninhada deve existir quando você executa o comando CREATE FUNCTION. O Amazon Redshift não rastreia as dependências das UDFs, portanto, se você remover a função aninhada, o Amazon Redshift não retornará um erro. Contudo, a UDF falhará se a função aninhada não existir. Por exemplo, a função a seguir chama a função `f_sql_greater` na cláusula SELECT.

```
create function f_sql_commission (float, float )
  returns float
stable
as $$
  select f_sql_greater ($1, $2)
$$ language sql;
```

Segurança e privilégios de UDF

Para criar uma UDF, você deve ter permissão de uso na linguagem para SQL ou plpythonu (Python). Por padrão, USAGE ON LANGUAGE SQL é concedido para PUBLIC. No entanto, é necessário conceder explicitamente USAGE ON LANGUAGE PLPYTHONU a usuários ou grupos específicos.

Para revogar o uso na SQL, revogue primeiro o uso em PUBLIC. Depois, conceda o uso na SQL somente a usuários ou grupos específicos que tenham permissão para criar UDFs SQL. O exemplo a seguir revoga o uso na SQL em PUBLIC e concede o uso ao grupo de usuários `udf_devs`.

```
revoke usage on language sql from PUBLIC;
grant usage on language sql to group udf_devs;
```

Para executar uma UDF, é necessário ter permissão de execução em cada função. Por padrão, a permissão de execução de novas UDFs é concedida a PUBLIC. Para restringir o uso, revogue a execução da permissão de PUBLIC para a função. Depois, conceda o privilégio a pessoas ou grupos específicos.

Este exemplo revoga a permissão de execução da função `f_py_greater` em PUBLIC e concede o uso ao grupo de usuários `udf_devs`.

```
revoke execute on function f_py_greater(a float, b float) from PUBLIC;
grant execute on function f_py_greater(a float, b float) to group udf_devs;
```

Os superusuários têm todos os privilégios por padrão.

Para obter mais informações, consulte GRANT e REVOKE.

Exemplos

Exemplo de UDF Python escalar

O exemplo a seguir cria uma UDF Python que compara dois inteiros e retorna o valor maior.

```
create function f_py_greater (a float, b float)
  returns float
stable
as $$
  if a > b:
    return a
  return b
$$ language plpythonu;
```

O exemplo a seguir consulta a tabela SALES e chama a nova função `f_py_greater` para retornar COMMISSION ou 20% de PRICEPAID, o que for maior.

```
select f_py_greater (commission, pricepaid*0.20) from sales;
```

Exemplo de UDF SQL escalar

O seguinte exemplo cria uma função que compara dois números e retorna o maior valor.

```
create function f_sql_greater (float, float)
  returns float
stable
as $$
  select case when $1 > $2 then $1
    else $2
```

```
  end
$$ language sql;
```

A consulta a seguir chama a nova função `f_sql_greater` para consultar a tabela SALES e retorna COMMISSION ou 20% de PRICEPAID, o que for maior.

```
select f_sql_greater (commission, pricepaid*0.20) from sales;
```

CREATE GROUP

Define um novo grupo de usuários. Somente um superusuário pode criar um grupo.

Sintaxe

```
CREATE GROUP group_name
[ [ WITH ] [ USER username ] [, ...] ]
```

Parâmetros

nome_grupo

Nome do novo grupo de usuários. Nomes de grupos que começam com dois sublinhados são reservados para uso interno do Amazon Redshift. Para obter mais informações sobre nomes válidos, consulte Nomes e identificadores.

WITH

Sintaxe opcional para indicar parâmetros adicionais de CREATE GROUP.

USER

Adiciona um ou mais usuários ao grupo.

Nome de usuário do

Nome do usuário a ser adicionado ao grupo.

Exemplos

O exemplo a seguir cria um grupo de usuários chamado ADMIN_GROUP com um dois usuários, ADMIN1 e ADMIN2.

```
create group admin_group with user admin1, admin2;
```

CREATE IDENTITY PROVIDER

Define um novo provedor de identidades. Somente um superusuário pode criar um provedor de identidades.

Sintaxe

```
CREATE IDENTITY PROVIDER identity_provider_name TYPE type_name
NAMESPACE namespace_name
[PARAMETERS parameter_string]
[APPLICATION_ARN arn]
[IAM_ROLE iam_role]
[AUTO_CREATE_ROLES
    [ TRUE [ { INCLUDE | EXCLUDE } GROUPS LIKE filter_pattern] |
      FALSE
    ]
  ];
```

Parâmetros

identity_provider_name

O nome do provedor de identidades. Para obter mais informações sobre nomes válidos, consulte Nomes e identificadores.

type_name

O provedor de identidades com o qual interagir. No momento, o AWSIDC é o único provedor de identidades compatível.

namespace_name

O namespace. Esse é um identificador exclusivo abreviado para o diretório do provedor de identidades.

parameter_string

Uma string com um objeto JSON formatado corretamente que contém os parâmetros e valores necessários para o provedor de identidades.

arn

O nome do recurso da Amazon (ARN) para uma aplicação gerenciada pelo Centro de Identidade do IAM. Esse parâmetro é aplicável somente quando o tipo de provedor de identidades é AWSIDC.

iam_role

O perfil do IAM que fornece permissões para fazer a conexão com o Centro de Identidade do IAM. Esse parâmetro é aplicável somente quando o tipo de provedor de identidades é AWSIDC.

auto_create_roles

Habilita ou desabilita o atributo de criação automática de perfil. Se o valor for TRUE, o Amazon Redshift habilitará o recurso de criação automática de perfil. Se o valor for FALSE, o Amazon Redshift desabilitará o recurso de criação automática de perfil. Se o valor desse parâmetro não for especificado, o Amazon Redshift o determinará usando a seguinte lógica:

- Se `AUTO_CREATE_ROLES` for habilitado, mas o valor não for especificado, o valor será definido como TRUE.
- Se `AUTO_CREATE_ROLES` não for habilitado e o provedor de identidades for AWSIDC, o valor será definido como FALSE.
- Se `AUTO_CREATE_ROLES` não for habilitado e o provedor de identidades for o Azure, o valor será definido como TRUE.

Para incluir grupos, especifique `INCLUDE`. O padrão é vazio, o que significa incluir todos os grupos se `AUTO_CREATE_ROLES` estiver ativado.

Para excluir grupos, especifique `EXCLUDE`. O padrão é vazio, o que significa não excluir nenhum grupo se `AUTO_CREATE_ROLES` estiver ativado.

filter_pattern

Uma expressão de caractere UTF-8 válida com o padrão para estabelecer correspondência com os nomes de grupo. A opção LIKE executa uma correspondência com distinção entre letras maiúsculas e minúsculas compatível com os seguintes metacaracteres de correspondência de padrões:

Metacaractere	Descrição
%	Corresponde a qualquer sequência de zero ou mais caracteres.

Metacaractere	Descrição
_	Corresponde a qualquer caractere único.

Se filter_pattern não contiver metacaracteres, o padrão representará somente a própria string. Nesse caso, LIKE age da mesma forma que o operador de igualdade.

filter_pattern aceita os seguintes caracteres:

- Caracteres alfabéticos maiúsculos e minúsculos (A-Z e a-z)
- Números (0-9)
- Os seguintes caracteres especiais:

```
_ % ^ * + ? { } , $
```

Exemplos

O exemplo a seguir cria um provedor de identidades chamado oauth_standard, com um TYPE Azure, para estabelecer comunicação com o Microsoft Azure Active Directory (AD).

```
CREATE IDENTITY PROVIDER oauth_standard TYPE azure
NAMESPACE 'aad'
PARAMETERS '{"issuer":"https://sts.windows.net/2sdfdsf-d475-420d-b5ac-667adad7c702/",
"client_id":"87f4aa26-78b7-410e-bf29-57b39929ef9a",
"client_secret":"BUAH~ewrqewrqwerUUY^%tHe1oNZShoiU7",
"audience":["https://analysis.windows.net/powerbi/connector/AmazonRedshift"]
}'
```

Você pode conectar uma aplicação gerenciada pelo Centro de Identidade do IAM a um cluster provisionado ou um grupo de trabalho existente do Amazon Redshift sem servidor. Isso permite gerenciar o acesso a um banco de dados do Redshift por meio do Centro de Identidade do IAM. Para fazer isso, execute um comando SQL de acordo com o exemplo a seguir. Você deve ser administrador de banco de dados.

```
CREATE IDENTITY PROVIDER "redshift-idc-app" TYPE AWSIDC
NAMESPACE 'awsidc'
APPLICATION_ARN 'arn:aws:sso::123456789012:application/ssoins-12345f67fe123d4/apl-
a0b0a12dc123b1a4'
```

```
IAM_ROLE 'arn:aws:iam::123456789012:role/MyRedshiftRole';
```

O ARN da aplicação, nesse caso, identifica a aplicação gerenciada à qual se conectar. Você pode encontrá-la executando `SELECT * FROM SVV_IDENTITY_PROVIDERS;`.

Para obter mais informações sobre como usar CREATE IDENTITY PROVIDER, incluindo exemplos adicionais, consulte Federação de um provedor de identidades (IdP) nativo para o Amazon Redshift. Para obter mais informações sobre a configuração de uma conexão do Redshift ao Centro de Identidade do IAM, consulte Conectar o Redshift ao IAM Identity Center para proporcionar aos usuários uma experiência de logon único.

CREATE LIBRARY

Instala uma biblioteca Python, que fica disponível para os usuários incorporarem ao criar uma função definida pelo usuário (UDF) com o comando CREATE FUNCTION. O tamanho total das bibliotecas instaladas pelo usuário não pode exceder 100 MB.

CREATE LIBRARY não pode ser executado em um bloco de transação (BEGIN … END). Para obter mais informações sobre transações, consulte Níveis de isolamento no Amazon Redshift.

O Amazon Redshift é compatível com Python versão 2.7. Para obter mais informações, acesse www.python.org.

Para obter mais informações, consulte Exemplo: importar módulos personalizados da biblioteca Python.

Privilégios obrigatórios

A seguir estão os privilégios obrigatórios para CREATE LIBRARY:

- Superusuário
- Usuários com o privilégio CREATE LIBRARY ou com o privilégio da linguagem especificada

Sintaxe

```
CREATE [ OR REPLACE ] LIBRARY library_name LANGUAGE plpythonu
FROM
{ 'https://file_url'
| 's3://bucketname/file_name'
authorization
```

```
  [ REGION [AS] 'aws_region']
  IAM_ROLE { default | 'arn:aws:iam::<Conta da AWS-id>:role/<role-name>' }
}
```

Parâmetros

OR REPLACE

Especifica que, se uma biblioteca com o mesmo nome que a atual já existe, a biblioteca existente será substituída. REPLACE é confirmado imediatamente. Se uma UDF que depende da biblioteca estiver em execução simultaneamente, a UDF poderá falhar ou retornar resultados inesperados, mesmo se a UDF estiver sendo executada em uma transação. É necessário ser o proprietário ou um superusuário para substituir uma biblioteca.

nome_biblioteca

Nome da biblioteca a ser instalada. Você não pode criar uma biblioteca que contenha um módulo com o mesmo nome que um módulo da biblioteca padrão Python ou de um módulo do Python pré-instalado pelo Amazon Redshift. Se uma biblioteca instalada pelo usuário existente usa o mesmo pacote Python que a biblioteca a ser instalada, é necessário remover a biblioteca existente antes de instalar a nova biblioteca. Para obter mais informações, consulte Suporte da linguagem Python para UDFs.

LANGUAGE plpythonu

A linguagem a ser usada. O Python (plpythonu) é a única linguagem compatível. O Amazon Redshift é compatível com Python versão 2.7. Para obter mais informações, acesse www.python.org.

FROM

Localização do arquivo da biblioteca. É possível especificar um nome de bucket e de objeto do Amazon S3, ou um URL para baixar o arquivo de um site público. A biblioteca deve ser compactada na forma de um arquivo `.zip`. Para obter mais informações, acesse Criar e instalar módulos Python na documentação do Python.

https://file_url

URL para baixar o arquivo de um site público. O URL pode conter até três redirecionamentos. Veja a seguir um exemplo de URL de arquivo.

```
'https://www.example.com/pylib.zip'
```

s3://bucket_name/file_name

O caminho para um único objeto do Amazon S3 que contém o arquivo de biblioteca. Veja a seguir um exemplo de um caminho de objeto do Amazon S3.

```
's3://amzn-s3-demo-bucket/my-pylib.zip'
```

Se você especificar um bucket do Amazon S3, também deve fornecer credenciais de um usuário da AWS com permissão para baixar o arquivo.

> **Important**
>
> Se o bucket do Amazon S3 não residir na mesma região da AWS que seu cluster do Amazon Redshift, use a opção REGION para especificar a região da AWS na qual os dados estão localizados. O valor da aws_region deve corresponder a uma região da AWS listada na tabela na descrição do parâmetro REGION para o comando COPY.

authorization

Cláusula que indica o método que o cluster usa na autenticação e autorização para acessar o bucket do Amazon S3 que contém o arquivo da biblioteca. Seu cluster deve ter permissão para acessar o Amazon S3 com as ações LIST e GET.

A sintaxe para autorização é a mesma para a autorização do comando COPY. Para obter mais informações, consulte Parâmetros de autorização.

```
IAM_ROLE { default | ‘arn:aws:iam::<Conta da AWS-id>:role/<role-name>’
```

Use a palavra-chave padrão para que o Amazon Redshift use a função do IAM definida como padrão e associada ao cluster quando o comando CREATE LIBRARY for executado.

Use o nome do recurso da Amazon (ARN) de uma função do IAM que seu cluster usa para autenticação e autorização. Se especificar IAM_ROLE, você não poderá usar ACCESS_KEY_ID e SECRET_ACCESS_KEY, SESSION_TOKEN ou CREDENTIALS.

Como opção, se o bucket do Amazon S3 usar criptografia no lado do servidor, forneça a chave de criptografia na string credentials-args. Se você usar credenciais de segurança temporárias, forneça o token temporário na string credentials-args.

Para obter mais informações, consulte Credenciais de segurança temporárias.

REGION [AS] aws_region

A região da AWS em que o bucket do Amazon S3 está localizado. REGION é necessário quando o bucket do Amazon S3 não está na mesma região da AWS que o cluster do Amazon Redshift. O valor da aws_region deve corresponder a uma região da AWS listada na tabela na descrição do parâmetro REGION para o comando COPY.

Por padrão, CREATE LIBRARY pressupõe que o bucket do Amazon S3 esteja localizado na mesma região da AWSque o cluster do Amazon Redshift.

Exemplos

Os dois exemplos a seguir instalam o módulo Python urlparse, que é compactado em um arquivo chamado `urlparse3-1.0.3.zip`.

O comando a seguir instala uma biblioteca de UDF chamada `f_urlparse` a partir de um arquivo compactado carregado no bucket do Amazon S3 localizado na região leste dos EUA.

```
create library f_urlparse
language plpythonu
from 's3://amzn-s3-demo-bucket/urlparse3-1.0.3.zip'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
region as 'us-east-1';
```

O exemplo a seguir instala uma biblioteca chamada `f_urlparse` a partir de um arquivo de biblioteca em um site.

```
create library f_urlparse
language plpythonu
from 'https://example.com/packages/urlparse3-1.0.3.zip';
```

CREATE MASKING POLICY

Cria uma política de mascaramento dinâmico de dados para ofuscar dados em um determinado formato. Para obter mais informações sobre mascaramento dinâmico de dados, consulte Mascaramento dinâmico de dados.

Superusuários e usuários ou perfis que têm o perfil sys:secadmin podem criar uma política de mascaramento.

Sintaxe

```
CREATE MASKING POLICY
   { policy_name | database_name.policy_name } [IF NOT EXISTS]
   WITH (input_columns)
   USING (masking_expression);
```

Parâmetros

policy_name

O nome da política de mascaramento. A política de mascaramento não pode ter o mesmo nome que outra política de mascaramento que já existe no banco de dados.

database_name

O nome do banco de dados de onde a política será criada. A política pode ser criada no banco de dados conectado ou no Catálogo de Permissões Federadas do Amazon Redshift.

input_columns

Uma tupla de nomes de colunas no formato (col1 type, col2 type ...).

Os nomes das colunas são usados como entrada para a expressão de mascaramento. Os nomes das colunas não precisam corresponder aos nomes das colunas que estão sendo mascaradas, mas os tipos de dados de entrada e saída devem corresponder.

masking_expression

A expressão SQL usada para transformar as colunas de destino. Ela pode ser escrita usando funções de manipulação de dados, como funções de manipulação de strings, ou em conjunto com funções definidas pelo usuário escritas em SQL, Python ou com o AWS Lambda. Você pode incluir uma tupla de expressões de coluna para políticas de mascaramento que tenham várias saídas. Se você usar uma constante como sua expressão de mascaramento, deverá convertê-la explicitamente em um tipo que corresponda ao tipo de entrada.

Você deve ter a permissão USAGE em todas as funções definidas pelo usuário que você usa na expressão de mascaramento.

Para o uso da CREATE MASKING POLICY no Catálogo de Permissões Federadas do Amazon Redshift, consulte Gerenciar o controle de acesso com permissões federadas do Amazon Redshift.

CREATE MATERIALIZED VIEW

Cria uma visão materializada com base em uma ou mais tabelas do Amazon Redshift. Você também pode basear visões materializadas em tabelas externas criadas usando Spectrum ou consulta federada. Para obter informações sobre o Spectrum, consulte Amazon Redshift Spectrum. Para obter informações sobre consulta federada, consulte Consultar dados com consultas federadas no Amazon Redshift.

Sintaxe

```
CREATE MATERIALIZED VIEW mv_name
[ BACKUP { YES | NO } ]
[ table_attributes ]
[ AUTO REFRESH { YES | NO } ]
AS query
```

A sintaxe a seguir cria uma visualização materializada que armazena seus dados como uma tabela do Apache Iceberg. Para acessar mais informações, consulte Visualizações materializadas armazenadas como tabelas do Apache Iceberg.

```
CREATE MATERIALIZED VIEW mv_name
USING ICEBERG
[LOCATION 's3://bucket/path/']
[PARTITIONED BY (partition_transform [, ...])]
[TABLE PROPERTIES ('property_name' = 'property_value' [, ...])]
AS query
```

em que partition_transform é:

```
    column_name |
    YEAR( column_name ) |
    MONTH( column_name ) |
    DAY( column_name ) |
    HOUR( column_name ) |
    BUCKET( integer, column_name ) |
    TRUNCATE( integer, column_name )
```

Parâmetros

BACKUP

Uma cláusula que especifica se uma visão materializada deve ser incluída em snapshots de cluster manuais ou automáticos.

Para visões materializadas que não contêm dados críticos, especifique BACKUP NO para economizar tempo de processamento ao criar snapshosts, restaurar de snapshots e reduzir o uso de espaço de armazenamento no Amazon Simple Storage Service. A configuração BACKUP NO não tem efeito na replicação automática de dados para outros nós no cluster. Portanto, visões materializadas com BACKUP NO especificado são restauradas em caso de falha de nó. O padrão é BACKUP YES.

table_attributes

Uma cláusula que especifica como os dados na visão materializada são distribuídos, incluindo o seguinte:

- O estilo de distribuição da visão materializada, no formato `DISTSTYLE { EVEN | ALL | KEY }`. Se você omitir essa cláusula, o estilo da distribuição será `EVEN`. Para obter mais informações, consulte Estilos de distribuição.
- O código de distribuição da visão materializada, no formato `DISTKEY ( distkey_identifier )`. Para obter mais informações, consulte Designação de estilos de distribuição.
- A chave de classificação para a visão materializada, no formato `SORTKEY ( column_name [, ...] )`. Para obter mais informações, consulte Chaves de classificação.

AS query

Uma instrução `SELECT` válida que define a visão materializada e seu conteúdo. O conjunto de resultados da consulta define as colunas e as linhas da visão materializada. Para obter informações sobre limitações ao criar visões materializadas, consulte Limitações.

Além disso, os constructos de linguagem SQL específicos usados na consulta determinam se a visão materializada pode ser atualizada de maneira incremental ou total. Para obter informações sobre o método de atualização, consulte REFRESH MATERIALIZED VIEW. Para obter informações sobre as limitações da atualização incremental, consulte Limitações para atualização incremental.

Se a consulta contiver um comando SQL incompatível com a atualização incremental, o Amazon Redshift exibirá uma mensagem indicando que a visão materializada usará uma atualização total. A mensagem poderá ou não ser exibida dependendo da aplicação cliente SQL. Verifique a coluna `state` do STV_MV_INFO para ver o tipo de atualização usado por uma visão materializada.

AUTO REFRESH

Uma cláusula que define se a visão materializada deve ser atualizada automaticamente com as alterações mais recentes de suas tabelas base. O valor padrão é `NO`. Para obter mais informações, consulte Atualizar uma visualização materializada.

USING ICEBERG

Especifica que a visualização materializada armazene seus dados como uma tabela do Apache Iceberg no Amazon S3. Quando você especifica USING ICEBERG, os dados da visualização materializada são gravados como arquivos Parquet no formato Iceberg e registrados no AWS Glue Data Catalog. Para acessar mais informações, consulte Visualizações materializadas armazenadas como tabelas do Apache Iceberg.

mv_name

Quando você usa USING ICEBERG, o nome da visualização materializada deve usar a notação qualificada pelo catálogo. Você pode usar a notação de três partes (`awsdatacatalog.`*`database`*`.`*`mv_name`*), a notação de Tabelas do Amazon S3 (`"`*`bucket`*`@s3tablescatalog".`*`database`*`.`*`mv_name`*) ou a notação de duas partes com um esquema externo (*`external_schema`*`.`*`mv_name`*). O banco de dados de destino não deve existir no AWS Glue Data Catalog.

LOCALIZAÇÃO

O caminho do Amazon S3 no qual os arquivos de dados e metadados do Iceberg são armazenados. O caminho deve apontar para um local vazio. O bucket da Amazon S3 deve estar na mesma região da Região da AWS que o cluster ou grupo de trabalho do Amazon Redshift. Omita esse parâmetro ao usar o Buckets de Tabelas do Amazon S3, porque as Tabelas do Amazon S3 gerenciam o local de armazenamento automaticamente. Para tabelas criadas usando esquemas externos ou o catálogo raiz `awsdatacatalog`, LOCATION é obrigatório.

PARTITIONED BY (partition_transform [, ...])

Especifica uma ou mais transformações de partição do Iceberg para a visualização materializada. O Amazon Redshift comporta todas as transformações de partição do Iceberg v2, exceto `void`. Uma única coluna não pode aparecer em várias transformações.

TABLE PROPERTIES ('property_name' = 'property_value' [, ...])

Especifica as propriedades da tabela do Iceberg. Por exemplo, use `'write.parquet.compression-codec' = 'zstd'` para definir o codec de compactação Parquet.

Observações de uso

Para criar uma visão materializada, você deve ter os seguintes privilégios:

- Privilégios CREATE para um esquema.
- Privilégio SELECT por tabela ou coluna para as tabelas de base a fim de criar uma visão materializada. Se você tiver privilégios por coluna para colunas específicas, poderá criar uma visão materializada somente para essas colunas.

É possível criar uma visão materializada em um cluster remoto de unidade de compartilhamento de dados fornecendo o nome do banco de dados externo no `mv_name`.

Visualizações materializadas do Iceberg

Para criar uma visualização materializada do Iceberg, você deve ter a permissão CREATE TABLE no banco de dados AWS Glue Data Catalog de destino. O perfil do IAM associado ao esquema externo (o perfil definidor de MV) deve ter a permissão SELECT em todas as tabelas de origem referidas na consulta.

Todos os identificadores na definição da visualização materializada (nomes de tabelas, nomes de colunas, aliases) devem estar em minúsculas. O AWS Glue Data Catalog armazena identificadores em letras minúsculas para compatibilidade com o Hive. O Amazon Redshift rejeita as instruções CREATE MATERIALIZED VIEW com USING ICEBERG que contêm identificadores em maiúsculas.

A criação ou atualização de visualizações materializadas do Iceberg não é aceita quando `enable_case_sensitive_identifier` está definida como `true`. Essa restrição existe porque o AWS Glue Data Catalog não comporta identificadores com distinção entre maiúsculas e minúsculas. Se seu cluster ou grupo de trabalho tiver `enable_case_sensitive_identifier` habilitado, configure-o como `false` para a sessão antes de criar ou atualizar as visualizações materializadas do Iceberg:

```
SET enable_case_sensitive_identifier TO false;
```

As tabelas de origem devem estar no formato Apache Iceberg, versão 2 ou inferior. Tabelas que não sejam do Iceberg não podem ser usadas como tabelas de origem para visualizações materializadas do Iceberg.

Você não pode usar o seguinte com visualizações materializadas do Iceberg:

- Cláusula BACKUP
- Cláusulas DISTSTYLE, DISTKEY ou SORTKEY
- Referências a tabelas nativas, tabelas temporárias ou tabelas do sistema do Amazon Redshift
- AUTO REFRESH (a atualização automática não é aceita; atualize manualmente as visualizações materializadas do Iceberg)

Para acessar mais informações sobre os recursos e limitações da visualização materializada do Iceberg, consulte Visualizações materializadas armazenadas como tabelas do Apache Iceberg.

Atualização incremental para visões materializadas em uma unidade de compartilhamento de dados

O Amazon Redshift oferece suporte à atualização automática e incremental de visões materializadas em uma unidade de compartilhamento de dados do consumidor quando as tabelas base são compartilhadas. A atualização incremental é uma operação em que o Amazon Redshift identifica alterações em uma ou mais tabelas base que ocorreram após a atualização anterior e atualiza somente os registros correspondentes na visão materializada. Ela é mais rápida do que uma atualização completa e melhora o desempenho da workload. Você não precisa alterar a definição de visão materializada para usar a atualização incremental.

Há algumas limitações a serem observadas para usar a atualização incremental com uma visão materializada:

- A visão materializada deve fazer referência a um banco de dados apenas, seja local ou remoto.
- A atualização incremental está disponível somente em novas visões materializadas. Portanto, você deve descartar as visões materializadas existentes e recriá-las para que ocorra a atualização incremental.

Para obter mais informações sobre a criação de visões materializadas em uma unidade de compartilhamento de dados, consulte Trabalhar com visualizações no compartilhamento de dados do Amazon Redshift, que contém vários exemplos de consulta.

Atualizações em DDL para visões materializadas ou tabelas base

Ao usar visões materializadas no Amazon Redshift, siga estas observações de uso para atualizações em linguagem de definição de dados (DDL) para visões materializadas ou tabelas base.

- É possível adicionar colunas a uma tabela base sem afetar as visões materializadas que fazem referência à tabela-base.
- Algumas operações podem deixar a visão materializada em um estado que não pode ser atualizado de forma alguma. Entre os exemplos estão operações como renomeação ou descarte de uma coluna, alteração do tipo de uma coluna e alteração do nome de um esquema. Essas visões materializadas podem ser consultadas, mas não podem ser atualizadas. Nesse caso, você deve descartar e recriar a visão materializada.
- Em geral, não é possível alterar a definição de uma visão materializada (instrução SQL desta).
- Não é possível renomear uma visão materializada.

Limitações

Você não pode definir uma visão materializada que faça referência ou inclua o seguinte:

- Exibições padrão ou tabelas e exibições do sistema.
- Tabelas temporárias.
- Funções definidas pelo usuário.
- A cláusula ORDER BY, LIMIT ou OFFSET.
- Referências de vinculação tardia a tabelas base. Ou seja, todas as tabelas base ou colunas relacionadas referenciadas na definição da consulta SQL da visão materializada devem existir e serem válidas.
- Funções somente de nó líder: CURRENT_SCHEMA, CURRENT_SCHEMAS, HAS_DATABASE_PRIVILEGE, HAS_SCHEMA_PRIVILEGE, HAS_TABLE_PRIVILEGE.
- Você não pode usar a opção AUTO REFRESH YES quando a definição de visão materializada incluir funções mutáveis ou esquemas externos. Você também não pode usá-lo ao definir uma visão materializada em outra visão materializada.
- Não é necessário executar manualmente o ANALYZE com visões materializadas. Atualmente, isso ocorre somente por meio do AUTO ANALYZE. Para obter mais informações, consulte Análise de tabelas.
- Tabelas protegidas por RLS ou DDM.

- As visões materializadas criadas em clusters remotos de unidades de compartilhamento de dados não são compatíveis com referências em outras visões materializadas, tabelas do Spectrum, tabelas definidas em um cluster diferente do Redshift e UDFs. Isso é possível quando as visões materializadas são criadas no cluster local (produtor).

Para visualizações materializadas do Iceberg criadas com USING ICEBERG, as seguintes limitações adicionais se aplicam:

- As tabelas de origem devem estar no formato Apache Iceberg.
- Todos os identificadores devem estar em minúscula.
- Funções definidas pelo usuário e funções mutáveis não são permitidas.
- As tabelas de origem devem residir na mesma Região da AWS e conta que a visualização materializada.
- Tabelas AWS Lake Formation filtradas (FGAC) não podem ser usadas como tabelas de origem.

Exemplos

O exemplo a seguir cria uma visão materializada de três tabelas base que são unidas e agregadas. Cada linha representa uma categoria com o número de bilhetes vendidos. Ao consultar a visão materializada tickets_mv, os dados pré-calculados da visão materializada tickets_mv são acessados diretamente.

```
CREATE MATERIALIZED VIEW tickets_mv AS
    select   catgroup,
    sum(qtysold) as sold
    from     category c, event e, sales s
    where    c.catid = e.catid
    and      e.eventid = s.eventid
    group by catgroup;
```

O exemplo a seguir cria uma visão materializada semelhante ao exemplo anterior e usa a função agregada MAX().

```
CREATE MATERIALIZED VIEW tickets_mv_max AS
    select   catgroup,
    max(qtysold) as sold
    from     category c, event e, sales s
```

```
    where    c.catid = e.catid
    and      e.eventid = s.eventid
    group by catgroup;

SELECT name, state FROM STV_MV_INFO;
```

O exemplo a seguir usa uma cláusula UNION ALL para unir a tabela `public_sales` do Amazon Redshift e a tabela `spectrum.sales` do Redshift Spectrum para criar uma visão materializada `mv_sales_vw`. Para obter informações sobre o comando CREATE EXTERNAL TABLE para Amazon Redshift Spectrum, consulte CREATE EXTERNAL TABLE. A tabela externa do Redshift Spectrum referencia os dados no Amazon S3.

```
CREATE MATERIALIZED VIEW mv_sales_vw as
select salesid, qtysold, pricepaid, commission, saletime from public.sales
union all
select salesid, qtysold, pricepaid, commission, saletime from spectrum.sales
```

O exemplo a seguir cria uma visão materializada `mv_fq` com base em uma tabela externa de consulta federada. Para obter informações sobre consulta federada, consulte CREATE EXTERNAL SCHEMA.

```
CREATE MATERIALIZED VIEW mv_fq as select firstname, lastname from apg.mv_fq_example;

select firstname, lastname from mv_fq;
 firstname | lastname
-----------+----------
 John      | Day
 Jane      | Doe
(2 rows)
```

O exemplo a seguir mostra a definição de uma visão materializada.

```
SELECT pg_catalog.pg_get_viewdef('mv_sales_vw'::regclass::oid, true);

pg_get_viewdef
-------------------------------------------------
create materialized view mv_sales_vw as select a from t;
```

O exemplo a seguir mostra como definir AUTO REFRESH na definição de visão materializada e também especifica um DISTSTYLE. Primeiro, crie uma tabela base simples.

```
CREATE TABLE baseball_table (ball int, bat int);
```

Depois, crie uma visão materializada.

```
CREATE MATERIALIZED VIEW mv_baseball DISTSTYLE ALL AUTO REFRESH YES AS SELECT ball AS
 baseball FROM baseball_table;
```

Agora você pode consultar a visão materializada de mv_baseball. Para verificar se a opção AUTO REFRESH está ativada para uma visão materializada, consulte STV_MV_INFO.

O exemplo a seguir cria uma visão materializada que faz referência a uma tabela de origem em outro banco de dados. Ele assume que o banco de dados que contém a tabela de origem, database_A, está no mesmo cluster ou grupo de trabalho da visão materializada, que é criada em database_B. (Você pode usar seus próprios bancos de dados no exemplo.) Primeiro, crie uma tabela em database_A chamada cities, com uma coluna cityname. Defina o tipo de dado da coluna como VARCHAR. Depois de criar a tabela de origem, execute o seguinte comando em database_B para criar uma visão materializada cuja origem seja sua tabela cities. Especifique o banco de dados e o esquema da tabela de origem na cláusula FROM:

```
CREATE MATERIALIZED VIEW cities_mv AS
SELECT  cityname
FROM    database_A.public.cities;
```

Consulte a visão materializada que você criou. A consulta recupera registros cuja fonte original é a tabela cities em database_A:

```
select * from cities_mv;
```

Quando você executa a instrução SELECT, cities_mv retorna os registros. Os registros são atualizados da tabela de origem somente quando uma instrução REFRESH é executada. Além disso, observe que não é possível atualizar registros diretamente na visão materializada. Para obter informações sobre como atualizar os dados em uma visão materializada, consulte REFRESH MATERIALIZED VIEW.

Para obter detalhes sobre a visão geral da visão materializada e os comandos SQL usados para atualizar e descartar visões materializadas, consulte os seguintes tópicos:

- Visões materializadas no Amazon Redshift
- REFRESH MATERIALIZED VIEW

- DROP MATERIALIZED VIEW

O exemplo a seguir cria uma visualização materializada do Iceberg em um bucket de Tabelas do Amazon S3 com particionamento.

```
CREATE MATERIALIZED VIEW "my-bucket@s3tablescatalog".analytics.sales_by_region
USING ICEBERG
PARTITIONED BY (region)
AS SELECT region, count(*) as order_count, sum(amount) as total_sales
   FROM "my-bucket@s3tablescatalog".production.orders
   GROUP BY region;
```

O exemplo a seguir cria uma visualização materializada do Iceberg em um bucket do Amazon S3 de uso geral com uma transformação e compactação de partição especificadas.

```
CREATE MATERIALIZED VIEW awsdatacatalog.mydb.daily_revenue
USING ICEBERG
LOCATION 's3://my-analytics-bucket/mvs/daily_revenue/'
PARTITIONED BY (day(order_date))
TABLE PROPERTIES ('write.parquet.compression-codec' = 'zstd')
AS SELECT order_date, sum(amount) as revenue, count(*) as num_orders
   FROM awsdatacatalog.mydb.orders
   GROUP BY order_date;
```

O exemplo a seguir cria uma visualização materializada do Iceberg usando um esquema externo com notação em duas partes.

```
CREATE MATERIALIZED VIEW my_ext_schema.customer_totals
USING ICEBERG
LOCATION 's3://my-analytics-bucket/mvs/customer_totals/'
AS SELECT customer_id, count(*) as order_count, sum(amount) as total_spent
   FROM my_ext_schema.orders
   GROUP BY customer_id;
```

CREATE MODEL

Tópicos

- Pré-requisitos
- Privilégios obrigatórios

- Controle de custos
- CREATE MODEL completo
- Parâmetros
- Observações de uso
- Casos de uso

Pré-requisitos

Antes de usar a instrução CREATE MODEL, conclua os pré-requisitos em Configuração de cluster para usar o Amazon Redshift ML. A seguir está um resumo de alto nível dos pré-requisitos.

- Crie um cluster do Amazon Redshift com o Console de Gerenciamento da AWS ou a Interface de linha de comando da AWS (AWS CLI).
- Anexe a política de Identity and Access Management (IAM) da AWS ao criar o cluster.
- Para permitir que o Amazon Redshift e o SageMaker AI assumam o perfil a fim de interagir com outros serviços, adicione a política de confiança apropriada ao perfil do IAM.

Para obter detalhes sobre a função do IAM, a política de confiança e outros pré-requisitos, consulte Configuração de cluster para usar o Amazon Redshift ML.

A seguir, você pode encontrar diferentes casos de uso para a instrução CREATE MODEL.

- CREATE MODEL simples
- CREATE MODEL com orientação do usuário
- Modelos CREATE XGBoost com AUTO OFF
- Traga seu próprio modelo (BYOM): inferência local
- Traga seu próprio modelo (BYOM): inferência remota
- CREATE MODEL com K-MEANS
- CREATE MODEL completo

Privilégios obrigatórios

A seguir estão os privilégios obrigatórios para CREATE MODEL:

- Superusuário

- Usuários com o privilégio CREATE MODEL
- Funções com privilégio GRANT CREATE MODEL

Controle de custos

O Amazon Redshift ML usa os recursos de cluster existentes para criar modelos de previsão. Portanato, você não precisa pagar a mais. No entanto, você pode ter custos adicionais se precisar redimensionar o cluster ou quiser treinar seus modelos. O Amazon Redshift ML usa o Amazon SageMaker AI para treinar modelos, o que tem um custo adicional associado. Há maneiras de controlar os custos adicionais, como limitar a quantidade máxima de tempo que o treinamento pode levar ou limitar o número de exemplos de treinamento usados para treinar o modelo. Para obter mais informações, consulte Custos de uso do Amazon Redshift ML.

CREATE MODEL completo

A seguir, resume as opções básicas da sintaxe CREATE MODEL completa.

Sintaxe completa do CREATE MODEL

A seguir está a sintaxe completa da instrução CREATE MODEL.

> **Important**
>
> Ao criar um modelo usando a instrução CREATE MODEL, siga a ordem das palavras-chave na sintaxe a seguir.

```
CREATE MODEL model_name
FROM { table_name | ( select_statement )  | 'job_name' }
[ TARGET column_name ]
FUNCTION function_name [ ( data_type [, ...] ) ]
[ RETURNS data_type ]
  -- supported only for BYOM
[ SAGEMAKER 'endpoint_name'[:'model_name']]
  -- supported only for BYOM remote inference
IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>' }
[ AUTO ON / OFF ]
  -- default is AUTO ON
[ MODEL_TYPE { XGBOOST | MLP | LINEAR_LEARNER | KMEANS | FORECAST } ]
  -- not required for non AUTO OFF case, default is the list of all supported types
```

```
  -- required for AUTO OFF
[ PROBLEM_TYPE ( REGRESSION | BINARY_CLASSIFICATION | MULTICLASS_CLASSIFICATION ) ]
  -- not supported when AUTO OFF
[ OBJECTIVE ( 'MSE' | 'Accuracy' | 'F1' | 'F1_Macro' | 'AUC' |
              'reg:squarederror' | 'reg:squaredlogerror'| 'reg:logistic'|
              'reg:pseudohubererror' | 'reg:tweedie' | 'binary:logistic' |
 'binary:hinge',
              'multi:softmax' | 'RMSE' | 'WAPE' | 'MAPE' | 'MASE' |
 'AverageWeightedQuantileLoss' ) ]
  -- for AUTO ON: first 5 are valid
  -- for AUTO OFF: 6-13 are valid
  -- for FORECAST: 14-18 are valid
[ PREPROCESSORS 'string' ]
  -- required for AUTO OFF, when it has to be 'none'
  -- optional for AUTO ON
[ HYPERPARAMETERS { DEFAULT | DEFAULT EXCEPT ( Key 'value' (,...) ) } ]
  -- support XGBoost hyperparameters, except OBJECTIVE
  -- required and only allowed for AUTO OFF
  -- default NUM_ROUND is 100
  -- NUM_CLASS is required if objective is multi:softmax (only possible for AUTO OFF)
 [ SETTINGS (
   S3_BUCKET 'amzn-s3-demo-bucket',  |
     -- required
  TAGS 'string', |
     -- optional
  KMS_KEY_ID 'kms_string', |
     -- optional
  S3_GARBAGE_COLLECT on / off, |
     -- optional, defualt is on.
  MAX_CELLS integer, |
     -- optional, default is 1,000,000
  MAX_RUNTIME integer (, ...) |
     -- optional, default is 5400 (1.5 hours)
  HORIZON integer, |
     -- required if creating a forecast model
  FREQUENCY integer, |
     -- required if creating a forecast model
  PERCENTILES string, |
     -- optional if creating a forecast model
  MAX_BATCH_ROWS integer -- optional for BYOM remote inference
     ) ]
```

Parâmetros

model_name

O nome do modelo. O nome do modelo em um esquema deve ser exclusivo.

FROM { table_name | (select_query) | 'job_name'}

O table_name ou a consulta que especifica os dados de treinamento. Eles podem ser uma tabela existente no sistema ou uma consulta SELECT compatível com o Amazon RedShift entre parênteses, ou seja, (). Deve haver pelo menos duas colunas no resultado da consulta.

TARGET column_name

O nome da coluna que se torna o alvo da previsão. A coluna deve existir na cláusula FROM.

FUNCTION function_name (data_type [, ...])

O nome da função a ser criada e os tipos de dados dos argumentos de entrada. É possível fornecer o nome de um esquema no banco de dados em vez de um nome de função.

RETURNS tipo_dados

O tipo de dados a ser retornado pela função do modelo. O tipo de dados SUPER retornado só se aplica a BYOM com inferência remota.

SAGEMAKER 'endpoint_name'[:'model_name']

O nome do endpoint do Amazon SageMaker AI. Se o nome do endpoint apontar para um endpoint multimodelo, adicione o nome do modelo a ser usado. O endpoint deve estar hospedado na mesma Região da AWS que o cluster do Amazon Redshift.

IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>'}

Use a palavra-chave padrão para que o Amazon Redshift use a função do IAM definida como padrão e associada ao cluster quando o comando CREATE MODEL for executado. Como alternativa, você pode especificar o ARN de um perfil do IAM para usar esse perfil.

[AUTO ON / OFF]

Ativa ou desativa a descoberta automática da seleção de hiperparâmetros, algoritmo e pré-processador de CREATE MODEL. Especificar “on” ao criar um modelo do Forecast indica o uso de um AutoPredictor, em que o Amazon Forecast aplica as combinações ideais de algoritmos a cada série temporal em seu conjunto de dados.

MODEL_TYPE { XGBOOST | MLP | LINEAR_LEARNER | KMEANS | FORECAST }

(Opcional) Especifica o tipo de modelo. Você pode especificar se deseja treinar um modelo de um tipo específico, como XGBoost, multilayer perceptron (MLP), KMEANS ou Linear Learner, que são todos algoritmos compatíveis com o Amazon SageMaker AI Autopilot. Se você não especificar o parâmetro, todos os tipos de modelo aceitos serão pesquisados durante o treinamento para obter o melhor modelo. Também é possível gerar um modelo de previsão no Redshift ML para criar previsões precisas de séries temporais.

PROBLEM_TYPE (REGRESSION | BINARY_CLASSIFICATION | MULTICLASS_CLASSIFICATION)

(Opcional) Especifica o tipo de problema. Se você souber o tipo de problema, pode restringir o Amazon Redshift a apenas pesquisar o melhor modelo desse tipo de modelo específico. Se você não especificar esse parâmetro, um tipo de problema será descoberto durante o treinamento com base em seus dados.

OBJECTIVE ('MSE' | 'Accuracy' | 'F1' | 'F1Macro' | 'AUC' | 'reg:squarederror' | 'reg:squaredlogerror' | 'reg:logistic' | 'reg:pseudohubererror' | 'reg:tweedie' | 'binary:logistic' | 'binary:hinge' | 'multi:softmax' | 'RMSE' | 'WAPE' | 'MAPE' | 'MASE' | 'AverageWeightedQuantileLoss')

(Opcional) Especifica o nome da métrica objetiva usada para medir a qualidade preditiva de um sistema de Machine Learning. Essa métrica é otimizada durante o treinamento para fornecer a melhor estimativa dos valores dos parâmetros do modelo a partir dos dados. Se você não especificar uma métrica explicitamente, o comportamento padrão é usar automaticamente MSE: para regressão, F1: para classificação binária, Precisão: para classificação multiclass. Para obter informações sobre objetivos, consulte AutoMLJobObjective na Referência de API do Amazon SageMaker AI e Learning Task Parameters na documentação do XGBOOST. Os valores RMSE, WAPE, MAPE, MASE e AverageWeightedQuantileLoss são aplicáveis somente aos modelos de previsão. Para obter mais informações, consulte a operação de API CreateAutoPredictor.

PREPROCESSORS 'string'

(Opcional) Especifica certas combinações de pré-processadores para determinados conjuntos de colunas. O formato é uma lista de columnSets e as transformações apropriadas a serem aplicadas a cada conjunto de colunas. O Amazon Redshift aplica todos os transformadores em uma lista de transformadores específica a todas as colunas no ColumnSet correspondente. Por exemplo, para aplicar OneHotEncoder com Imputer às colunas t1 e t2, use o comando de exemplo a seguir.

```
CREATE MODEL customer_churn
```

```
FROM customer_data
TARGET 'Churn'
FUNCTION predict_churn
IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>' }
PROBLEM_TYPE BINARY_CLASSIFICATION
OBJECTIVE 'F1'
PREPROCESSORS '[
...
  {"ColumnSet": [
      "t1",
      "t2"
    ],
    "Transformers": [
      "OneHotEncoder",
      "Imputer"
    ]
  },
  {"ColumnSet": [
      "t3"
    ],
    "Transformers": [
      "OneHotEncoder"
    ]
  },
  {"ColumnSet": [
      "temp"
    ],
    "Transformers": [
      "Imputer",
      "NumericPassthrough"
    ]
  }
]'
SETTINGS (
  S3_BUCKET 'amzn-s3-demo-bucket'
)
```

HYPERPARAMETERS { DEFAULT | DEFAULT EXCEPT (key ‘value’ (,..)) }

Especifica se os parâmetros padrão do XGBoost são usados ou substituídos por valores especificados pelo usuário. Os valores devem ser colocados entre aspas simples. A seguir estão exemplos de parâmetros para XGBoost e seus padrões.

Nome do parâmetro	Valor do parâmetro	Valor padrão	Observações
num_class	Inteiro	Necessário para a classificação Multiclass.	N/D
num_round	Inteiro	100	N/D
tree_method	String	Auto	N/D
max_depth	Inteiro	6	[0, 10]
min_child_weight	Float	1	MinValue: 0, MaxValue: 120
subsample	Float	1	MinValue: 0.5, MaxValue: 1
gamma	Float	0	MinValue: 0, MaxValue: 5
alpha	Float	0	MinValue: 0, MaxValue: 1000
eta	Float	0.3	MinValue: 0.1, MaxValue: 0.5
colsample_byleve	Float	1	MinValue: 0.1, MaxValue: 1
colsample_bynode	Float	1	MinValue: 0.1, MaxValue: 1
colsample_bytree	Float	1	MinValue: 0.5, MaxValue: 1

Nome do parâmetro	Valor do parâmetro	Valor padrão	Observações
lambda	Float	1	MinValue: 0, MaxValue: 1000
max_delta_step	Inteiro	0	[0, 10]

SETTINGS (S3_BUCKET 'amzn-s3-demo-bucket', | TAGS 'string', | KMS_KEY_ID 'kms_string' , | S3_GARBAGE_COLLECT on / off, | MAX_CELLS integer , | MAX_RUNTIME (,...) , | HORIZON integer, | FREQUENCY forecast_frequency, | PERCENTILES array of strings)

A cláusula S3_BUCKET especifica o local do Amazon S3 usado para armazenar resultados intermediários.

(Opcional) O parâmetro TAGS é uma lista separada por vírgulas de pares de chave-valor que você pode usar para marcar recursos criados no Amazon SageMaker AI e no Amazon Forecast. As tags ajudam você a organizar recursos e alocar custos. Os valores no par são opcionais, então você pode criar tags usando o formato `key=value` ou simplesmente criando uma chave. Para obter mais informações sobre tags no Amazon Redshift, consulte Visão geral da marcação.

(Opcional) KMS_KEY_ID especifica se o Amazon Redshift usa criptografia do lado do servidor com uma chave do AWS KMS para proteger dados em repouso. Os dados em trânsito são protegidos com Secure Sockets Layer (SSL).

(Opcional) S3_GARBAGE_COLLECT { ON | OFF } especifica se o Amazon Redshift executa a coleta de resíduos nos conjuntos de dados resultantes usados nos modelos e para treiná-los. Se definido como OFF, os conjuntos de dados resultantes usados para treinar modelos e os modelos permanecem no Amazon S3 e podem ser usados para outros fins. Se definido como ON, o Amazon Redshift excluirá os artefatos no Amazon S3 após a conclusão do treinamento. O padrão é ON.

(Opcional) MAX_CELLS especifica o número de células nos dados de treinamento. Esse valor é o produto do número de registros (na consulta de treinamento ou tabela) vezes o número de colunas. O padrão é 1.000.000.

(Opcional) MAX_RUNTIME especifica a quantidade máxima de tempo para treinar. Os trabalhos de treinamento geralmente concluem mais cedo dependendo do tamanho do conjunto de dados.

Isso especifica a quantidade máxima de tempo em que o treinamento deve levar. The default is 5.400 (90 minutos).

HORIZON especifica o número máximo de previsões que o modelo pode retornar. Depois que o modelo é treinado, não é possível alterar esse número inteiro. Esse parâmetro é necessário para treinar um modelo de previsão.

FREQUENCY especifica o nível de detalhamento que você deseja que as previsões tenham, em unidades de tempo. As opções disponíveis são `Y | M | W | D | H | 30min | 15min | 10min | 5min | 1min`. Esse parâmetro é necessário para treinar um modelo de previsão.

(Opcional) PERCENTILES é uma string delimitada por vírgula que especifica os tipos de previsão usados para treinar um previsor. Os tipos de previsão podem ser quantis de 0,01 a 0,99, em incrementos de 0,01 ou mais. Você também pode especificar a previsão média com mean. É possível especificar até cinco tipos de previsão.

MAX_BATCH_ROWS inteiro

(Opcional) O número máximo de linhas que o Amazon Redshift envia em uma única solicitação em lote para uma única invocação do SageMaker AI. Ele é compatível somente com BYOM com inferência remota. O valor mínimo desse parâmetro é 1. O valor máximo é `INT_MAX` ou 2.147.483.647. Esse parâmetro é necessário somente quando os tipos de dados de entrada e retornados são SUPER. O valor padrão é `INT_MAX`, ou 2.147.483.647.

Observações de uso

Ao usar CREATE MODEL, considere o seguinte:

- A instrução CREATE MODEL opera em um modo assíncrono e retorna após a exportação de dados de treinamento para o Amazon S3. As etapas restantes do treinamento no Amazon SageMaker AI ocorrem em segundo plano. Enquanto o treinamento estiver em andamento, a função de inferência correspondente será visível, mas não pode ser executada. É possível consultar STV_ML_MODEL_INFO para ver o estado do treinamento.
- O treinamento pode ser executado por até 90 minutos em segundo plano, por padrão no modelo Automático e pode ser estendido. Para cancelar o treinamento, basta executar o comando DROP MODEL.
- O cluster do Amazon Redshift que você usa para criar o modelo e o bucket do Amazon S3 que é usado para preparar os dados de treinamento e os artefatos do modelo devem estar na mesma região da AWS.

- Durante o treinamento do modelo, o Amazon Redshift e o SageMaker AI armazenam artefatos intermediários no bucket do Amazon S3 fornecido por você. Por padrão, o Amazon Redshift executa a coleta de resíduos no final da operação CREATE MODEL. O Amazon Redshift remove esses objetos do Amazon S3. Para reter esses artefatos no Amazon S3, defina a opção S3_GARBAGE COLLECT OFF.
- Você deve usar pelo menos 500 linhas nos dados de treinamento fornecidos na cláusula FROM.
- Você só pode especificar até 256 colunas de recursos (entrada) na cláusula FROM { table_name | (select_query) } ao usar a instrução CREATE MODEL.
- Para AUTO ON, os tipos de coluna que você pode usar como conjunto de treinamento são SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE, BOOLEAN, CHAR, VARCHAR, DATE, TIME, TIMETZ, TIMESTAMP, e TIMESTAMPTZ. Para AUTO ON, os tipos de coluna que você pode usar como conjunto de treinamento são SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE, e BOOLEAN.
- Não é possível usar DECIMAL, DATE, TIME, TIMETZ, TIMESTAMP, TIMESTAMPTZ, GEOMETRY, GEOGRAPHY, HLLSKETCH, SUPER ou VARBYTE como o tipo de coluna de destino.
- Para melhorar a precisão do modelo, execute um dos seguintes procedimentos:
  - Adicione tantas colunas relevantes no comando CREATE MODEL quanto possível quando você especificar os dados de treinamento na cláusula FROM.
  - Use um valor maior para MAX_RUNTOME e MAX_CELLS. Valores maiores para este parâmetro aumentam o custo do treinamento de um modelo.
- A execução da instrução CREATE MODEL retorna assim que os dados de treinamento são calculados e exportados para o bucket do Amazon S3. Após esse ponto, você pode verificar o status do treinamento usando o comando SHOW MODEL. Quando um modelo que está sendo treinado em segundo plano falhar, você pode verificar o erro usando SHOW MODEL. Não é possível repetir um modelo com falha. Use DROP MODEL para remover um modelo com falha e recriar um novo modelo. Para obter mais informações sobre SHOW MODEL, consulte SHOW MODEL.
- O BYOM local oferece suporte ao mesmo tipo de modelos que o Amazon Redshift ML oferece suporte para casos não BYOM. O Amazon Redshift é compatível com o XGBoost simples (usando o XGBoost versão 1.0 ou posterior), modelos KMEANS sem pré-processadores e modelos XGBOOST/MLP/Linear Learner treinados pelo Amazon SageMaker AI Autopilot. Ele é compatível com este último com pré-processadores especificados pelo Autopilot e que são compatíveis com o Amazon SageMaker AI Neo.

- Se o cluster do Amazon Redshift tiver o roteamento aprimorado habilitado para a nuvem privada virtual (VPC), crie um endpoint da VPC do Amazon S3 e um endpoint da VPC do SageMaker AI para a VPC em que o cluster está. Isso permite que o tráfego seja executado através de sua VPC entre esses serviços durante CREATE MODEL. Para obter mais informações, consulte Sub-redes e grupos de segurança da Amazon VPC dos trabalhos do SageMaker Clarify.

Casos de uso

Os casos de uso a seguir demonstram como usar CREATE MODEL para atender às suas necessidades.

CREATE MODEL simples

A seguir o resumo das opções básicas da sintaxe CREATE MODEL.

Sintaxe simples CREATE MODEL

```
CREATE MODEL model_name
FROM { table_name | ( select_query ) }
TARGET column_name
FUNCTION prediction_function_name
IAM_ROLE { default }
SETTINGS (
  S3_BUCKET 'amzn-s3-demo-bucket',
  [ MAX_CELLS integer ]
)
```

Parâmetros simples CREATE MODEL

model_name

O nome do modelo. O nome do modelo em um esquema deve ser exclusivo.

FROM { table_name | (select_query) }

O table_name ou a consulta que especifica os dados de treinamento. Eles podem ser uma tabela existente no sistema ou uma consulta SELECT compatível com o Amazon RedShift entre parênteses, ou seja, (). Deve haver pelo menos duas colunas no resultado da consulta.

TARGET column_name

O nome da coluna que se torna o alvo da previsão. A coluna deve existir na cláusula FROM.

FUNCTION prediction_function_name

Um valor que especifica o nome da função de Machine Learning do Amazon Redshift a ser gerada pelo CREATE MODEL e usada para fazer previsões usando esse modelo. A função é criada no mesmo esquema que o objeto modelo e pode ser sobrecarregada.

O Machine Learning do Amazon Redshift oferece suporte a modelos, como modelos Xtreme Gradient Boosted tree (XGBoost) para regressão e classificação.

IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>'}

Use a palavra-chave padrão para que o Amazon Redshift use a função do IAM definida como padrão e associada ao cluster quando o comando CREAT MODEL for executado. Como alternativa, você pode especificar o ARN de um perfil do IAM para usar esse perfil.

S3_BUCKET 'amzn-s3-demo-bucket'

O nome do bucket do Amazon S3 que você criou anteriormente era usado para compartilhar dados e artefatos de treinamento entre o Amazon Redshift e o SageMaker AI. O Amazon Redshift cria uma subpasta neste bucket antes do descarregamento dos dados de treinamento. Quando o treinamento for concluído, o Amazon Redshift exclui a subpasta criada e seu conteúdo.

Inteiro MAX_CELLS

O número máximo de células a serem exportadas da cláusula FROM. O padrão é 1.000.000.

O número de células é o produto do número de linhas nos dados de treinamento (produzidos pela tabela ou consulta da cláusula FROM) vezes o número de colunas. Se o número de células nos dados de treinamento for maior do que o especificado pelo parâmetro max_cells, CREATE MODEL baixará os dados de treinamento da cláusula FROM para reduzir o tamanho do conjunto de treinamento abaixo de MAX_CELLS. Permitir conjuntos de dados de treinamento maiores pode produzir maior precisão, mas também pode significar que o modelo leva mais tempo para treinar e custa mais.

Para obter informações sobre os custos de uso do Amazon Redshift, consulte Custos para usar o Amazon Redshift ML.

Para obter mais informações sobre os custos associados a vários números de celular e detalhes de avaliação gratuita, consulte Preços do Amazon Redshift.

CREATE MODEL com orientação do usuário

A seguir, você pode encontrar uma descrição das opções para CREATE MODEL, além das opções descritas em CREATE MODEL simples.

Por padrão, CREATE MODEL procura a melhor combinação de pré-processamento e modelo para seu conjunto de dados específico. Você pode querer controle adicional ou introduzir conhecimento de domínio adicional (como tipo de problema ou objetivo) sobre seu modelo. Em um cenário de rotatividade do cliente, se o resultado “cliente não está ativo” for raro, o objetivo de F1 geralmente é preferido ao objetivo de precisão. Como os modelos de alta precisão podem prever “o cliente está ativo” o tempo todo, isso resulta em alta precisão, mas pouco valor empresarial. Para obter informações sobre o objetivo F1, consulte AutoMLJobObjective na Referência de API do Amazon SageMaker AI.

Em seguida, o CREATE MODEL segue suas sugestões sobre os aspectos especificados, como o objetivo. Ao mesmo tempo, o CREATE MODEL descobre automaticamente os melhores pré-processadores e os melhores hiperparâmetros.

CREATE MODEL com sintaxe de orientação do usuário

CREATE MODEL oferece mais flexibilidade nos aspectos que você pode especificar e nos aspectos que o Amazon Redshift detecta automaticamente.

```
CREATE MODEL model_name
FROM { table_name | ( select_statement ) }
TARGET column_name
FUNCTION function_name
IAM_ROLE { default }
[ MODEL_TYPE { XGBOOST | MLP | LINEAR_LEARNER} ]
[ PROBLEM_TYPE ( REGRESSION | BINARY_CLASSIFICATION | MULTICLASS_CLASSIFICATION ) ]
[ OBJECTIVE ( 'MSE' | 'Accuracy' | 'F1' | 'F1Macro' | 'AUC') ]
SETTINGS (
  S3_BUCKET 'amzn-s3-demo-bucket', |
  S3_GARBAGE_COLLECT { ON | OFF }, |
  KMS_KEY_ID 'kms_key_id', |
  MAX_CELLS integer, |
  MAX_RUNTIME integer (, ...)
)
```

CREATE MODEL com parâmetros de orientação do usuário

MODEL_TYPE { XGBOOST | MLP | LINEAR_LEARNER }

(Opcional) Especifica o tipo de modelo. Você pode especificar se deseja treinar um modelo de um tipo específico, como XGBoost, perceptron multicamada (MLP) ou Linear Learner, que são todos algoritmos compatíveis com o Amazon SageMaker AI Autopilot. Se você não especificar o parâmetro, todos os tipos de modelo aceitos serão pesquisados durante o treinamento para obter o melhor modelo.

PROBLEM_TYPE (REGRESSION | BINARY_CLASSIFICATION | MULTICLASS_CLASSIFICATION)

(Opcional) Especifica o tipo de problema. Se você souber o tipo de problema, pode restringir o Amazon Redshift a apenas pesquisar o melhor modelo desse tipo de modelo específico. Se você não especificar esse parâmetro, um tipo de problema será descoberto durante o treinamento com base em seus dados.

OBJECTIVE ('MSE' | 'Accuracy' | 'F1' | 'F1Macro' | 'AUC')

(Opcional) Especifica o nome da métrica objetiva usada para medir a qualidade preditiva de um sistema de Machine Learning. Essa métrica é otimizada durante o treinamento para fornecer a melhor estimativa dos valores dos parâmetros do modelo a partir dos dados. Se você não especificar uma métrica explicitamente, o comportamento padrão é usar automaticamente MSE: para regressão, F1: para classificação binária, Precisão: para classificação multiclass. Para obter informações sobre os objetivos, consulte AutoMLJobObjective na Referência de API do Amazon SageMaker AI.

Inteiro MAX_CELLS

(Opcional) Especifica o número de células nos dados de treinamento. Esse valor é o produto do número de registros (na consulta de treinamento ou tabela) vezes o número de colunas. O padrão é 1.000.000.

Inteiro MAX_RUNTIME

(Opcional) Especifica a quantidade máxima de tempo para treinar. Os trabalhos de treinamento geralmente concluem mais cedo dependendo do tamanho do conjunto de dados. Isso especifica a quantidade máxima de tempo em que o treinamento deve levar. The default is 5.400 (90 minutos).

S3_GARBAGE_COLLECT { ON | OFF }

(Opcional) Especifica se o Amazon Redshift executa a coleta de lixo nos conjuntos de dados resultantes usados nos modelos e para treiná-los. Se definido como OFF, os conjuntos de dados resultantes usados para treinar modelos e os modelos permanecem no Amazon S3 e podem ser usados para outros fins. Se definido como ON, o Amazon Redshift excluirá os artefatos no Amazon S3 após a conclusão do treinamento. O padrão é ON.

KMS_KEY_ID 'kms_key_id'

(Opcional) Especifica se o Amazon Redshift usa criptografia do lado do servidor com uma chave AWS KMS para proteger dados em repouso. Os dados em trânsito são protegidos com Secure Sockets Layer (SSL).

PREPROCESSORS 'string'

(Opcional) Especifica certas combinações de pré-processadores para determinados conjuntos de colunas. O formato é uma lista de columnSets e as transformações apropriadas a serem aplicadas a cada conjunto de colunas. O Amazon Redshift aplica todos os transformadores em uma lista de transformadores específica a todas as colunas no ColumnSet correspondente. Por exemplo, para aplicar OneHotEncoder com Imputer às colunas t1 e t2, use o comando de exemplo a seguir.

```
CREATE MODEL customer_churn
FROM customer_data
TARGET 'Churn'
FUNCTION predict_churn
IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>' }
PROBLEM_TYPE BINARY_CLASSIFICATION
OBJECTIVE 'F1'
PREPROCESSORS '[
...
{"ColumnSet": [
    "t1",
    "t2"
  ],
  "Transformers": [
    "OneHotEncoder",
    "Imputer"
  ]
},
{"ColumnSet": [
    "t3"
```

```
  ],
  "Transformers": [
    "OneHotEncoder"
  ]
},
{"ColumnSet": [
    "temp"
  ],
  "Transformers": [
    "Imputer",
    "NumericPassthrough"
  ]
}
]'
SETTINGS (
S3_BUCKET 'amzn-s3-demo-bucket'
)
```

O Amazon Redshift oferece suporte aos seguintes transformadores:

- OneHotEncoder — Normalmente usado para codificar um valor discreto em um vetor binário com um valor diferente de zero. Este transformador é adequado para muitos modelos de machine learning.
- OrdinalEncoder — Codifica valores discretos em um único inteiro. Esse transformador é adequado para determinados modelos de machine learning, como MLP e Linear Learner.
- NumericPassThrough — Passa a entrada como está no modelo.
- Imputer — Preenche valores ausentes e valores não numéricos (NaN).
- ImputerWithIndicator — Preenche valores ausentes e valores NaN. Este transformador também cria um indicador de se algum valor estava faltando e preenchido.
- Normalizer — Normaliza valores, o que pode melhorar a performance de muitos algoritmos de Machine Learning.
- DateTimeVectorizer — Cria uma incorporação vetorial, representando uma coluna do tipo de dados datetime que pode ser usada em modelos de Machine Learning.
- PCA — Projeta os dados em um espaço dimensional mais baixo para reduzir o número de recursos e, ao mesmo tempo, manter o máximo de informações possível.
- StandardScaler: padroniza os recursos removendo a média e a escalabilidade para a variância da unidade.

- minMax: transforma recursos escalando cada recurso para um determinado intervalo.

O Amazon Redshift ML armazena os transformadores treinados e os aplica automaticamente como parte da consulta de previsão. Você não precisa especificá-los ao gerar previsões do seu modelo.

Modelos CREATE XGBoost com AUTO OFF

O AUTO OFF CREATE MODEL tem geralmente objetivos diferentes do padrão CREATE MODEL.

Como um usuário avançado que já conhece o tipo de modelo que você deseja e hiperparâmetros para usar ao treinar esses modelos, você pode usar CREATE MODEL com AUTO OFF para desativar a descoberta automática CREATE MODEL de pré-processadores e hiperparâmetros. Para fazer isso, especifique explicitamente o tipo de modelo. O XGBoost é atualmente o único tipo de modelo compatível quando AUTO é definido como OFF. Você pode especificar hiperparâmetros. O Amazon Redshift usa valores padrão para quaisquer hiperparâmetros especificados.

Modelos CREATE XGBoost com sintaxe AUTO OFF

```
CREATE MODEL model_name
FROM { table_name | (select_statement ) }
TARGET column_name
FUNCTION function_name
IAM_ROLE { default }
AUTO OFF
MODEL_TYPE XGBOOST
OBJECTIVE { 'reg:squarederror' | 'reg:squaredlogerror' | 'reg:logistic' |
            'reg:pseudohubererror' | 'reg:tweedie' | 'binary:logistic' | 'binary:hinge'
 |
            'multi:softmax' | 'rank:pairwise' | 'rank:ndcg' }
HYPERPARAMETERS DEFAULT EXCEPT (
    NUM_ROUND '10',
    ETA '0.2',
    NUM_CLASS '10',
    (, ...)
)
PREPROCESSORS 'none'
SETTINGS (
  S3_BUCKET 'amzn-s3-demo-bucket', |
  S3_GARBAGE_COLLECT { ON | OFF }, |
  KMS_KEY_ID 'kms_key_id', |
  MAX_CELLS integer, |
  MAX_RUNTIME integer (, ...)
```

```
)
```

Modelos CREATE XGBoost com parâmetros AUTO OFF

AUTO OFF

Desativa a descoberta automática de pré-processador, algoritmo e seleção de hiper-parâmetros do CREATE MODEL.

MODEL_TYPE XGBOOST

Especifica o uso do XGBOOST para treinar o modelo.

OBJECTIVE str

Especifica um objetivo reconhecido pelo algoritmo. O Amazon Redshift oferece suporte a reg:squarederror, reg:squaredlogerror, reg:logistic, reg:pseudohubererror, reg:tweedie, binary:logistic, binary:hinge e multi:softmax. Para obter mais informações sobre esses objetivos, consulte Aprendizado de parâmetros de tarefa na documentação do XGBoost.

HYPERPARAMETERS { DEFAULT | DEFAULT EXCEPT (key ‘value’ (,..)) }

Especifica se os parâmetros padrão do XGBoost são usados ou substituídos por valores especificados pelo usuário. Os valores devem ser colocados entre aspas simples. A seguir estão exemplos de parâmetros para XGBoost e seus padrões.

Nome do parâmetro	Valor do parâmetro	Valor padrão	Observações
num_class	Inteiro	Necess o para a classifi ação Multicl s.	N/D
num_round	Inteiro	100	N/D
tree_meth od	String	Auto	N/D

Nome do parâmetro	Valor do parâmetro	Valor padrão	Observações
max_depth	Inteiro	6	[0, 10]
min_child_weight	Float	1	MinValue: 0, MaxValue: 120
subsample	Float	1	MinValue: 0.5, MaxValue: 1
gamma	Float	0	MinValue: 0, MaxValue: 5
alpha	Float	0	MinValue: 0, MaxValue: 1000
eta	Float	0.3	MinValue: 0.1, MaxValue: 0.5
colsample_byleve	Float	1	MinValue: 0.1, MaxValue: 1
colsample_bynode	Float	1	MinValue: 0.1, MaxValue: 1
colsample_bytree	Float	1	MinValue: 0.5, MaxValue: 1
lambda	Float	1	MinValue: 0, MaxValue: 1000
max_delta_step	Inteiro	0	[0, 10]

O exemplo a seguir prepara dados para XGBoost.

```
DROP TABLE IF EXISTS abalone_xgb;

CREATE TABLE abalone_xgb (
length_val float,
diameter float,
height float,
whole_weight float,
shucked_weight float,
```

```
viscera_weight float,
shell_weight float,
rings int,
record_number int);

COPY abalone_xgb
FROM 's3://redshift-downloads/redshift-ml/abalone_xg/'
REGION 'us-east-1'
IAM_ROLE default
IGNOREHEADER 1 CSV;
```

O exemplo a seguir cria um modelo XGBoost com opções avançadas especificadas, como MODEL_TYPE, OBJETIVE e PREPROCESSORS.

```
DROP MODEL abalone_xgboost_multi_predict_age;

CREATE MODEL abalone_xgboost_multi_predict_age
FROM ( SELECT length_val,
              diameter,
              height,
              whole_weight,
              shucked_weight,
              viscera_weight,
              shell_weight,
              rings
   FROM abalone_xgb WHERE record_number < 2500 )
TARGET rings FUNCTION ml_fn_abalone_xgboost_multi_predict_age
IAM_ROLE default
AUTO OFF
MODEL_TYPE XGBOOST
OBJECTIVE 'multi:softmax'
PREPROCESSORS 'none'
HYPERPARAMETERS DEFAULT EXCEPT (NUM_ROUND '100', NUM_CLASS '30')
SETTINGS (S3_BUCKET 'amzn-s3-demo-bucket');
```

O exemplo a seguir usa uma consulta de inferência para prever a idade do peixe com um número de registro maior que 2.500. Ele usa a função ml_fn_abalone_xgboost_multi_predict_age criada a partir do comando acima.

```
select ml_fn_abalone_xgboost_multi_predict_age(length_val,
                                               diameter,
                                               height,
```

```
                                                    whole_weight,
                                                    shucked_weight,
                                                    viscera_weight,
                                                    shell_weight)+1.5 as age
from abalone_xgb where record_number > 2500;
```

Traga seu próprio modelo (BYOM): inferência local

O Amazon Redshift ML é compatível com o uso de BYOM (traga seu próprio modelo) para inferência local.

A seguir o resumo das opções básicas da sintaxe CREATE MODEL para BYOM. Você pode usar um modelo treinado fora do Amazon Redshift com o Amazon SageMaker AI para inferência no banco de dados localmente no Amazon Redshift.

Sintaxe CREATE MODEL para inferência local

A seguir é descrito a sintaxe CREATE MODEL para inferência local.

```
CREATE MODEL model_name
FROM ('job_name' | 's3_path' )
FUNCTION function_name ( data_type [, ...] )
RETURNS data_type
IAM_ROLE { default }
[ SETTINGS (
  S3_BUCKET 'amzn-s3-demo-bucket', | --required
  KMS_KEY_ID 'kms_string') --optional
];
```

Atualmente, o Amazon Redshift comporta apenas modelos XGBoost, MLP e Linear Learner pré-treinados para BYOM. Você pode importar o SageMaker AI Autopilot e modelos treinados diretamente no Amazon SageMaker AI para inferência local usando esse caminho.

Parâmetros CREATE MODEL para inferência local

model_name

O nome do modelo. O nome do modelo em um esquema deve ser exclusivo.

FROM ('job_name' | 's3_path')

O job_name usa um nome de trabalho do Amazon SageMaker AI como entrada. O nome do trabalho pode ser um nome de trabalho de treinamento do Amazon SageMaker AI ou um nome

de trabalho do Amazon SageMaker AI Autopilot. O trabalho deve ser criado na mesma conta da AWS que possui o cluster do Amazon Redshift. Para encontrar o nome do trabalho, inicie o Amazon SageMaker AI. No menu suspenso Training (Treinamento), escolha Training jobs (Trabalhos de treinamento).

O 's3_path' especifica o local S3 do arquivo de artefatos de modelo .tar.gz que deve ser usado ao criar o modelo.

FUNCTION function_name (data_type [, ...])

O nome da função a ser criada e os tipos de dados dos argumentos de entrada. Você pode fornecer um nome de esquema.

RETURNS tipo_dados

Tipo de dados do valor retornado pela função.

IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>'}

Use a palavra-chave padrão para que o Amazon Redshift use a função do IAM definida como padrão e associada ao cluster quando o comando CREATE MODEL for executado.

Use o nome do recurso da Amazon (ARN) de uma função do IAM que seu cluster usa para autenticação e autorização.

SETTINGS (S3_BUCKET 'amzn-s3-demo-bucket', | KMS_KEY_ID 'kms_string')

A cláusula S3_BUCKET especifica o local do Amazon S3 usado para armazenar resultados intermediários.

(Opcional) A cláusula KMS_KEY_ID especifica se o Amazon Redshift usa criptografia do lado do servidor com uma chave do AWS KMS para proteger dados em repouso. Os dados em trânsito são protegidos com Secure Sockets Layer (SSL).

Para obter mais informações, consulte CREATE MODEL com orientação do usuário.

CREATE MODEL para exemplo de inferências locais

O exemplo a seguir cria um modelo que foi previamente treinado no Amazon SageMaker AI, fora do Amazon Redshift. Como o tipo de modelo é compatível com o Amazon Redshift ML para inferência local, o seguinte CREATE MODEL cria uma função que pode ser usada localmente no Amazon Redshift. Você pode fornecer um nome de tarefa de treinamento do SageMaker AI.

```
CREATE MODEL customer_churn
FROM 'training-job-customer-churn-v4'
FUNCTION customer_churn_predict (varchar, int, float, float)
RETURNS int
IAM_ROLE default
SETTINGS (S3_BUCKET 'amzn-s3-demo-bucket');
```

Depois que o modelo é criado, você pode usar a função customer_churn_predict com os tipos de argumento especificados para fazer previsões.

Traga seu próprio modelo (BYOM): inferência remota

O Amazon Redshift ML também é compatível com o uso de BYOM (traga seu próprio modelo) para inferência remota.

A seguir o resumo das opções básicas da sintaxe CREATE MODEL para BYOM.

Sintaxe CREATE MODEL para inferência remota

A seguir é descrito a sintaxe CREATE MODEL para inferência remota.

```
CREATE MODEL model_name
FUNCTION function_name ( data_type [, ...] )
RETURNS data_type
SAGEMAKER 'endpoint_name'[:'model_name']
IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>' }
[SETTINGS (MAX_BATCH_ROWS integer)];
```

Parâmetros CREATE MODEL para inferência remota

model_name

O nome do modelo. O nome do modelo em um esquema deve ser exclusivo.

FUNCTION fn_name ([data_type] [, ...])

O nome da função e os tipos de dados dos argumentos de entrada. Consulte Tipos de dados para obter todos os tipos de dados compatíveis. `Geography`,`geometry` e `hllsketch` não são compatíveis.

Você também pode fornecer um nome de função dentro de um esquema usando a notação em duas partes, como `myschema.myfunction`.

RETURNS tipo_dados

Tipo de dados do valor retornado pela função. Consulte Tipos de dados para obter todos os tipos de dados compatíveis. `Geography`,`geometry` e `hllsketch` não são compatíveis.

SAGEMAKER 'endpoint_name'[:'model_name']

O nome do endpoint do Amazon SageMaker AI. Se o nome do endpoint apontar para um endpoint multimodelo, adicione o nome do modelo a ser usado. O endpoint deve ser hospedado na mesma região da AWS e Conta da AWS que o cluster do Amazon Redshift. Para encontrar o endpoint, inicie o Amazon SageMaker AI. No menu suspenso Inference (Inferência), escolha Endpoints.

IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>'}

Use a palavra-chave padrão para que o Amazon Redshift use a função do IAM definida como padrão e associada ao cluster quando o comando CREATE MODEL for executado. Como alternativa, você pode especificar o ARN de um perfil do IAM para usar esse perfil.

MAX_BATCH_ROWS inteiro

O número máximo de linhas que o Amazon Redshift envia em uma única solicitação em lote para uma única invocação do SageMaker AI. Ele é compatível somente com BYOM com inferência remota. O número real de linhas em um lote também depende do tamanho da entrada, mas é menor ou igual a esse valor. O valor mínimo desse parâmetro é 1. O valor máximo é `INT_MAX` ou 2.147.483.647. Esse parâmetro é necessário somente quando os tipos de dados de entrada e retornados são `SUPER`. O valor padrão é `INT_MAX`, ou 2.147.483.647.

Quando o modelo é implantado em um endpoint do SageMaker AI, o SageMaker AI cria as informações do modelo no Amazon Redshift. Em seguida, executa a inferência através da função externa. Você pode usar o comando SHOW MODEL para exibir as informações do modelo no cluster do Amazon Redshift.

CREATE MODEL para observações de uso da inferência remota

Antes de usar CREATE MODEL para inferência remota, considere o seguinte:

- O endpoint deve ser hospedado na mesma conta da AWS que possui o cluster do Amazon Redshift.

- Verifique se o endpoint do Amazon SageMaker AI tem recursos suficientes para atender a chamadas de inferência do Amazon Redshift ou se o endpoint do Amazon SageMaker AI pode ser escalado automaticamente.
- Se você não estiver usando o tipo de dados SUPER como entrada, o modelo só aceitará entradas no formato de valores separados por vírgula (CSV), o que corresponde ao tipo de conteúdo de `text/CSV` no SageMaker AI.
- Se você não estiver usando o tipo de dados SUPER como entrada, a saída dos modelos será um valor único do tipo especificado ao criar a função. A saída está no formato de valores separados por vírgula (CSV) por meio de um tipo de conteúdo de `text/CSV` no SageMaker AI. Os tipos de dados VARCHAR não podem estar entre aspas e não podem conter novas linhas, e cada saída precisa estar em uma nova linha.
- Os modelos aceitam nulos como strings vazias.
- Quando o tipo de dados de entrada é SUPER, somente um argumento de entrada é permitido.
- Quando o tipo de dados de entrada é SUPER, o tipo de dados retornado também deve ser SUPER.
- MAX_BATCH_ROWS é necessário quando os tipos de dados de entrada e retornados são SUPER.
- Quando o tipo de dados de entrada é SUPER o tipo de conteúdo da invocação do endpoint é `application/json` quando MAX_BATCH_ROWS é 1 ou `application/jsonlines` em todos os outros casos.
- Quando o tipo de dados retornados é SUPER o tipo de aceitação da invocação do endpoint é `application/json` quando MAX_BATCH_ROWS é 1 ou `application/jsonlines` em todos os outros casos.

CREATE MODEL para exemplo de inferências remotas

O exemplo a seguir cria um modelo que usa o endpoint do SageMaker AI para fazer previsões. Certifique-se de que o endpoint está em execução para fazer previsões e especifique seu nome no comando CREATE MODEL.

```
CREATE MODEL remote_customer_churn
FUNCTION remote_fn_customer_churn_predict (varchar, int, float, float)
RETURNS int
SAGEMAKER 'customer-churn-endpoint'
IAM_ROLE default;
```

O exemplo a seguir cria um BYOM com inferência remota com um grande modelo de linguagem (LLM). Os LLMs hospedados no Amazon SageMaker AI Jumpstart aceitam e retornam o tipo de conteúdo `application/json` e permitem um único JSON por invocação. Os tipos de dados de entrada e retornados devem ser SUPER e MAX_BATCH_ROWS deve ser definido como 1.

```
CREATE MODEL sample_super_data_model
FUNCTION sample_super_data_model_predict(super)
RETURNS super
SAGEMAKER 'sample_super_data_model_endpoint'
IAM_ROLE default
SETTINGS (MAX_BATCH_ROWS 1);
```

CREATE MODEL com K-MEANS

O Amazon Redshift oferece suporte ao algoritmo K-Means que agrupa dados que não são rotulados. Esse algoritmo resolve problemas de clusterização em que você deseja detectar agrupamentos nos dados. Os dados não classificados são agrupados e particionados de acordo com suas semelhanças e diferenças.

CREATE MODEL com sintaxe K-MEANS

```
CREATE MODEL model_name
FROM { table_name | ( select_statement ) }
FUNCTION function_name
IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>' }
AUTO OFF
MODEL_TYPE KMEANS
PREPROCESSORS 'string'
HYPERPARAMETERS DEFAULT EXCEPT ( K 'val' [, ...] )
SETTINGS (
  S3_BUCKET 'amzn-s3-demo-bucket',
  KMS_KEY_ID 'kms_string', |
    -- optional
  S3_GARBAGE_COLLECT on / off, |
    -- optional
  MAX_CELLS integer, |
    -- optional
  MAX_RUNTIME integer
    -- optional);
```

CREATE MODEL com parâmetros K-MEANS

AUTO OFF

Desativa a descoberta automática de pré-processador, algoritmo e seleção de hiper-parâmetros do CREATE MODEL.

MODEL_TYPE KMEANS

Especifica o uso do KMEANS para treinar o modelo.

PREPROCESSORS 'string'

Especifica certas combinações de pré-processadores para determinados conjuntos de colunas. O formato é uma lista de columnSets e as transformações apropriadas a serem aplicadas a cada conjunto de colunas. O Amazon Redshift oferece suporte a três pré-processadores K-Means: StandardScaler, MinMax e NumericPassThrough. Caso não queira aplicar nenhum pré-processamento para K-means, escolha NumericPassthrough explicitamente como um transformador. Para obter mais informações sobre os transformadores compatíveis, consulte CREATE MODEL com parâmetros de orientação do usuário.

O algoritmo K-Means usa a distância euclidiana para calcular a semelhança. O pré-processamento dos dados garante que os recursos do modelo permaneçam na mesma escala e produzam resultados confiáveis.

HYPERPARAMETERS DEFAULT EXCEPT (K 'val' [, ...])

Especifica se os parâmetros K-Means são usados. É necessário especificar o parâmetro K ao usar o algoritmo K-Means. Para obter mais informações, consulte Hiperparâmetros do k-means no Guia do desenvolvedor do Amazon SageMaker AI

O exemplo a seguir prepara dados para K-Means.

```
CREATE MODEL customers_clusters
FROM customers
FUNCTION customers_cluster
IAM_ROLE default
AUTO OFF
MODEL_TYPE KMEANS
PREPROCESSORS '[
{
  "ColumnSet": [ "*" ],
  "Transformers": [ "NumericPassthrough" ]
```

```
}
]'
HYPERPARAMETERS DEFAULT EXCEPT ( K '5' )
SETTINGS (S3_BUCKET 'amzn-s3-demo-bucket');

select customer_id, customers_cluster(...) from customers;
customer_id | customers_cluster
--------------------
12345            1
12346            2
12347            4
12348
```

CREATE MODEL com o Forecast

Os modelos de previsão no Redshift ML usam o Amazon Forecast para criar previsões precisas de séries temporais. Dessa forma, é possível usar dados históricos de um determinado período para fazer previsões sobre eventos futuros. Os casos de uso comuns do Amazon Forecast incluem o uso de dados de produtos de varejo para decidir como definir os preços dos itens, dados de quantidade de fabricação para prever quanto de um item pedir e dados de tráfego da web para prever quanto tráfego um servidor da web pode receber.

Os limites de cota do Amazon Forecast são aplicados nos modelos de previsão do Amazon Redshift. Por exemplo, o número máximo de previsões é 100, mas esse valor é ajustável. A eliminação de um modelo de previsão não exclui automaticamente os recursos associados no Amazon Forecast. Se você excluir um cluster do Redshift, todos os modelos associados também serão descartados.

Observe que, no momento, os modelos do Forecast só estão disponíveis nas seguintes regiões:

- Leste dos EUA (Ohio) (us-east-2)
- Leste dos EUA (Norte da Virgínia) (us-east-1)
- Oeste dos EUA (Oregon) (us-west-2)
- Ásia-Pacífico (Mumbai) (ap-south-1)
- Ásia-Pacífico (Seul) (ap-northeast-2)
- Ásia-Pacífico (Singapura) (ap-southeast-1)
- Ásia-Pacífico (Sydney) (ap-southeast-2)
- Ásia Pacific (Tóquio) (ap-northeast-1)
- Europa (Frankfurt) (eu-central-1)

- Europa (Irlanda) (eu-west-1)

Sintaxe de CREATE MODEL com o Forecast

```
CREATE [ OR REPLACE ] MODEL forecast_model_name
FROM { table_name | ( select_query ) }
TARGET column_name
IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>'}
AUTO ON
MODEL_TYPE FORECAST
SETTINGS (
  S3_BUCKET 'amzn-s3-demo-bucket',
  HORIZON integer,
  FREQUENCY forecast_frequency
  [PERCENTILES '0.1', '0.5', '0.9']
  )
```

Parâmetros de CREATE MODEL com o Forecast

forecast_model_name

O nome do modelo. O nome do modelo deve ser exclusivo.

FROM { table_name | (select_query) }

O table_name ou a consulta que especifica os dados de treinamento. Isso pode ser uma tabela existente no sistema ou uma consulta SELECT compatível com o Amazon Redshift entre parênteses. O resultado da tabela ou consulta deve ter pelo menos três colunas: (1) uma coluna varchar que especifica o nome da série temporal (cada conjunto de dados pode ter várias séries temporais); (2) uma coluna de data e hora; e (3) a coluna de destino das previsões. Essa coluna de destino deve ser do tipo int ou float. Se você fornecer um conjunto de dados com mais de três colunas, o Amazon Redshift vai assumir que todas as colunas adicionais fazem parte de uma série temporal relacionada. Observe que as séries temporais relacionadas devem ser do tipo int ou float. Para obter mais informações sobre séries temporais relacionadas, consulte Usar conjuntos de dados de séries temporais relacionadas.

TARGET column_name

O nome da coluna que se torna o alvo da previsão. A coluna deve existir na cláusula FROM.

IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>'}

Use a palavra-chave padrão para que o Amazon Redshift use a função do IAM definida como padrão e associada ao cluster quando o comando CREAT MODEL for executado. Como alternativa, você pode especificar o ARN de um perfil do IAM para usar esse perfil.

AUTO ON

Ativa a descoberta automática da seleção de hiperparâmetros e algoritmo de CREATE MODEL. Especificar “on” ao criar um modelo do Forecast indica o uso de um Forecast AutoPredictor, em que o Amazon Forecast aplica as combinações ideais de algoritmos a cada série temporal em seu conjunto de dados.

MODEL_TYPE FORECAST

Especifica o uso de FORECAST para treinar o modelo.

S3_BUCKET 'amzn-s3-demo-bucket'

O nome do bucket do Amazon Simple Storage Service que você criou anteriormente e que é usado para compartilhar artefatos e dados de treinamento entre o Amazon Redshift e o Amazon Forecast. O Amazon Redshift cria uma subpasta neste bucket antes de descarregar os dados de treinamento. Quando o treinamento for concluído, o Amazon Redshift exclui a subpasta criada e seu conteúdo.

HORIZON integer

O número máximo de previsões que o modelo pode retornar. Depois que o modelo é treinado, não é possível alterar esse número inteiro.

FREQUENCY forecast_frequency

Especifica o nível de detalhamento que você deseja que as previsões tenham. As opções disponíveis são `Y | M | W | D | H | 30min | 15min | 10min | 5min | 1min`. Obrigatório se você for treinar um modelo de previsão.

PERCENTILES string

Uma string delimitada por vírgula que especifica os tipos de previsão usados para treinar um previsor. Os tipos de previsão podem ser quantis de 0,01 a 0,99, em incrementos de 0,01 ou mais. Você também pode especificar a previsão média com mean. É possível especificar até cinco tipos de previsão.

O exemplo a seguir demonstra como criar um modelo de previsão simples.

```
CREATE MODEL forecast_example
FROM forecast_electricity_
TARGET target
IAM_ROLE 'arn:aws:iam::<account-id>:role/<role-name>'
AUTO ON
MODEL_TYPE FORECAST
SETTINGS (S3_BUCKET 'amzn-s3-demo-bucket',
          HORIZON 24,
          FREQUENCY 'H',
          PERCENTILES '0.25,0.50,0.75,mean',
          S3_GARBAGE_COLLECT OFF);
```

Depois de criar o modelo de previsão, você pode criar uma tabela com os dados da previsão.

```
CREATE TABLE forecast_model_results as SELECT Forecast(forecast_example)
```

Em seguida, você pode consultar a nova tabela para obter previsões.

```
SELECT * FROM forecast_model_results
```

CREATE PROCEDURE

Cria um novo procedimento armazenado ou substitui um procedimento existente para o banco de dados atual.

Para ter mais informações e exemplos, consulte Criar procedimentos armazenados no Amazon Redshift.

Privilégios obrigatórios

Você deve ter permissão de uma das seguintes formas para executar CREATE OR REPLACE PROCEDURE:

- Para CREATE PROCEDURE:
  - Superusuário
  - Usuários com privilégios CREATE e USAGE no esquema no qual o procedimento armazenado é criado.
- Para REPLACE PROCEDURE:

- Superusuário
- Proprietário do procedimento

Sintaxe

```
CREATE [ OR REPLACE ] PROCEDURE sp_procedure_name
  ( [ [ argname ] [ argmode ] argtype [, ...] ] )
[ NONATOMIC ]
AS $$
  procedure_body
$$ LANGUAGE plpgsql
[ { SECURITY INVOKER | SECURITY DEFINER } ]
[ SET configuration_parameter { TO value | = value } ]
```

Parâmetros

OR REPLACE

Uma cláusula que especifica que, se um procedimento com o mesmo nome e tipos de dados do argumento de entrada, ou assinatura, como esse já existir, o procedimento existente será substituído. Só é possível substituir um procedimento por um novo procedimento que defina um conjunto idêntico de tipos de dados.

Se você definir um procedimento com o mesmo nome de um procedimento existente, mas com uma assinatura diferente, um novo procedimento será criado. Em outras palavras, o nome do procedimento é sobrecarregado. Para obter mais informações, consulte Sobrecarga de nomes de procedimento.

sp_procedure_name

O nome do procedimento. Se você especificar um nome de esquema (como **`myschema.myprocedure`**), o procedimento será criado no esquema especificado. Caso contrário, o procedimento será criado no esquema atual. Para obter mais informações sobre nomes válidos, consulte Nomes e identificadores.

Recomendamos que você prefixe os nomes de todos os UDF com `sp_`. O Amazon Redshift reserva o prefixo `sp_` para nomes de procedimentos armazenados. Ao usar o prefixo `sp_`, você garante que o nome do procedimento armazenado não entre em conflito com nenhum nome de procedimento armazenado ou função integrada existente ou futuro do Amazon Redshift. Para obter mais informações, consulte Nomeação de procedimentos armazenados.

É possível definir mais de um procedimento com o mesmo nome se os tipos de dados dos argumentos de entrada, ou assinaturas, forem diferentes. Em outras palavras, nesse caso, o nome do procedimento é sobrecarregado. Para obter mais informações, consulte . Sobrecarga de nomes de procedimento

[argname] [argmode] argtype

Uma lista de nomes de argumentos, modos de argumentos e tipos de dados. Somente o tipo de dados é obrigatório. O nome e o modo são opcionais e suas posições podem ser alternadas.

O modo do argumento pode ser IN, OUT ou INOUT. O padrão é IN.

Use argumentos OUT e INOUT para retornar um ou mais valores de uma chamada de procedimento. Quando houver argumentos OUT ou INOUT, a chamada de procedimento retorna uma linha de resultados que contém n colunas, onde n é o número total de argumentos OUT ou INOUT.

Argumentos INOUT são argumentos de entrada e saída ao mesmo tempo. Argumentos de entrada incluem argumentos IN e INOUT, e argumentos de saída incluem argumentos OUT e INOUT.

Argumentos OUT não são especificados como parte da instrução CALL. Especifique argumentos INOUT na instrução CALL do procedimento armazenado. Argumentos INOUT podem ser úteis ao enviar e retornar valores de uma chamada aninhada, bem como ao retornar um `refcursor`. Para obter mais informações sobre os tipos `refcursor`, consulte Cursores.

Os tipos de dados de argumento podem ser qualquer tipo de dados padrão do Amazon Redshift. Além disso, um tipo de dados de argumento pode ser `refcursor`.

Você pode especificar no máximo 32 argumentos de entrada e 32 argumentos de saída.

AS $$ procedure_body $$

Uma construção que delimita o procedimento a ser executado. As palavras-chave literais AS $$ e $$ são obrigatórias.

O Amazon Redshift exige que você inclua a instrução em seu procedimento usando um formato chamado cotação de dólar. Qualquer item incluído na instrução é aprovado exatamente como está. Não é preciso inserir caracteres de escape antes dos caracteres especiais, pois o conteúdo da string é gravado literalmente.

Com a cotação de dólar, um par de símbolos de cifrão ($$) deve ser usado no início e no final da instrução para que ela seja executada, conforme exibido no exemplo a seguir.

```
$$ my statement $$
```

Como opção, entre cada par de símbolos de cifrão, você pode especificar uma string para ajudar a identificar a instrução. A string que você usa deve ser a mesma no início e no final dos pares de sinais. A sequência diferencia maiúsculas de minúsculas e segue as mesmas restrições de um identificador sem aspas, com a exceção de que não pode conter cifrões. O exemplo a seguir usa a string test.

```
$test$ my statement $test$
```

Essa sintaxe também é útil para cotação de dólar aninhada. Para obter mais informações sobre o uso de símbolos de cifrão, consulte “Constantes de string entre símbolos de cifrão” em Estrutura lexical na documentação do PostgreSQL.

procedure_body

Um conjunto de instruções válidas da PL/pgSQL. As instruções da PL/pgSQL aumentam comandos SQL com construções processuais, incluindo loops e expressões condicionais, a fim de controlar o fluxo lógico. A maioria dos comandos SQL podem ser usados no corpo do procedimento, incluindo linguagem de modificação de dados (DML), como COPY, UNLOAD e INSERT, e linguagem de definição de dados (DDL), como CREATE TABLE. Para obter mais informações, consulte Referência da linguagem PL/pgSQL.

LANGUAGE plpgsql

Um valor de linguagem. Especifique `plpgsql`. É necessário ter permissão de uso na linguagem para usar `plpgsql`. Para obter mais informações, consulte GRANT.

NONATOMIC

Cria o procedimento armazenado em um modo de transação não atômico. O modo NONATOMIC confirma automaticamente as instruções dentro do procedimento. Além disso, quando ocorre um erro dentro do procedimento NONATOMIC, o erro não será relançado se for tratado por um bloco de exceções. Para obter mais informações, consulte Gerenciamento de transações e RAISE.

Ao definir um procedimento armazenado como `NONATOMIC`, considere o seguinte:

- Quando você aninha chamadas de procedimentos armazenados, todos os procedimentos devem ser criados no mesmo modo de transação.
- A opção `SECURITY DEFINER` e a opção `SET configuration_parameter` não são compatíveis ao criar um procedimento no modo NONATOMIC.

- Qualquer cursor aberto (explícita ou implicitamente) será fechado automaticamente quando uma confirmação implícita for processada. Portanto, é necessário iniciar uma transação explícita antes de iniciar um loop de cursor para garantir que qualquer SQL dentro da iteração do loop não seja confirmado implicitamente.

SECURITY INVOKER | SECURITY DEFINER

A opção `SECURITY DEFINER` não é compatível quando `NONATOMIC` é especificado.

O modo de segurança para o procedimento determina os privilégios de acesso do procedimento em tempo de execução. O procedimento deve ter permissão para acessar os objetos de banco de dados subjacentes.

Para o modo SECURITY INVOKER, o procedimento usa os privilégios do usuário que chama o procedimento. O usuário deve ter permissões explícitas para os objetos de banco de dados subjacentes. O padrão é SECURITY INVOKER.

Para o modo SECURITY DEFINER, o procedimento usa os privilégios do proprietário do procedimento. O proprietário do procedimento é definido como o usuário que possui o procedimento no tempo de execução, não necessariamente o usuário que definiu o procedimento inicialmente. O usuário que chama o procedimento precisa ter privilégio de execução no procedimento, mas não precisa de qualquer privilégio nos objetos subjacentes.

SET configuration_parameter { TO value | = value }

Essas opções não são compatíveis quando `NONATOMIC` é especificado.

A cláusula SET faz com que o `configuration_parameter` especificado seja definido como o valor especificado quando o procedimento for inserido. Depois, essa cláusula restaurará `configuration_parameter` para seu valor anterior quando o procedimento sair.

Observações de uso

Se um procedimento armazenado tiver sido criado usando a opção SECURITY DEFINER, ao invocar a função CURRENT_USER de dentro do procedimento armazenado, o Amazon Redshift retornará o nome de usuário do proprietário do procedimento armazenado.

Exemplos

> **Note**
>
> Se você encontrar um erro na execução desses exemplos parecido com:

```
ERROR: 42601: [Amazon](500310) unterminated dollar-quoted string at or near "$$
```

Consulte Visão geral dos procedimentos armazenados no Amazon Redshift.

O exemplo a seguir cria um procedimento com dois parâmetros de entrada.

```
CREATE OR REPLACE PROCEDURE test_sp1(f1 int, f2 varchar(20))
AS $$
DECLARE
  min_val int;
BEGIN
  DROP TABLE IF EXISTS tmp_tbl;
  CREATE TEMP TABLE tmp_tbl(id int);
  INSERT INTO tmp_tbl values (f1),(10001),(10002);
  SELECT min_val MIN(id) FROM tmp_tbl;
  RAISE INFO 'min_val = %, f2 = %', min_val, f2;
END;
$$ LANGUAGE plpgsql;
```

Note

Quando você escreve procedimentos armazenados, recomendamos a prática de proteger valores confidenciais:
Não codifique nenhuma informação confidencial na lógica do procedimento armazenado. Por exemplo, não atribua uma senha de usuário em uma instrução CREATE USER no corpo de um procedimento armazenado. Isso representa um risco de segurança, pois valores codificados podem ser registrados como metadados de esquema nas tabelas do catálogo. Em vez disso, transmita valores confidenciais, como senhas, como argumentos ao procedimento armazenado por meio de parâmetros.
Para obter mais informações sobre os procedimentos armazenados, consulte CREATE PROCEDURE e "Criar procedimentos armazenados no Amazon Redshift". Para obter mais informações sobre tabelas de catálogo, consulte "Tabelas de catálogo do sistema".

O exemplo a seguir cria um procedimento com um parâmetro IN, um parâmetro OUT e um parâmetro INOUT.

```
CREATE OR REPLACE PROCEDURE test_sp2(f1 IN int, f2 INOUT varchar(256), out_var OUT
 varchar(256))
AS $$
DECLARE
  loop_var int;
BEGIN
  IF f1 is null OR f2 is null THEN
    RAISE EXCEPTION 'input cannot be null';
  END IF;
  DROP TABLE if exists my_etl;
  CREATE TEMP TABLE my_etl(a int, b varchar);
    FOR loop_var IN 1..f1 LOOP
        insert into my_etl values (loop_var, f2);
        f2 := f2 || '+' || f2;
    END LOOP;
  SELECT INTO out_var count(*) from my_etl;
END;
$$ LANGUAGE plpgsql;
```

O exemplo a seguir cria um procedimento que usa o parâmetro `SECURITY DEFINER`. Esse procedimento é executado usando os privilégios do usuário proprietário do procedimento.

```
CREATE OR REPLACE PROCEDURE sp_get_current_user_definer()
AS $$
DECLARE curr_user varchar(250);
BEGIN
  SELECT current_user INTO curr_user;
  RAISE INFO '%', curr_user;
END;
$$ LANGUAGE plpgsql
SECURITY DEFINER;
```

O exemplo a seguir cria um procedimento que usa o parâmetro `SECURITY INVOKER`. Esse procedimento é executado usando os privilégios do usuário que executa o procedimento.

```
CREATE OR REPLACE PROCEDURE sp_get_current_user_invoker()
AS $$
DECLARE curr_user varchar(250);
BEGIN
  SELECT current_user INTO curr_user;
  RAISE INFO '%', curr_user;
```

```
END;
$$ LANGUAGE plpgsql
SECURITY INVOKER;
```

CREATE RLS POLICY

Cria uma nova política de segurança no nível da linha para fornecer acesso granular aos objetos do banco de dados.

Superusuários e usuários ou funções que têm a função sys:secadmin podem criar uma política.

Sintaxe

```
CREATE RLS POLICY { policy_name | database_name.policy_name }
[ WITH (column_name data_type [, ...]) [ [AS] relation_alias ] ]
USING ( using_predicate_exp )
```

Parâmetros

policy_name

O nome da política.

database_name

O nome do banco de dados de onde a política será criada. A política pode ser criada no banco de dados conectado ou em um banco de dados que aceite as permissões federadas do Amazon Redshift.

WITH (column_name data_type [, ...])

Especifica o column_name e data_type referenciada às colunas de tabelas às quais a política está anexada.

É possível omitir a cláusula WITH somente quando a política de RLS não fizer referência a nenhuma coluna de tabelas às quais a política está anexada.

AS relation_alias

Especifica um alias opcional para a tabela à qual a política de RLS será anexada.

USING (using_predicate_exp)

Especifica um filtro que é aplicado à cláusula WHERE da consulta. O Amazon Redshift aplica um predicado de política antes dos predicados do usuário no nível da consulta. Por exemplo, **`current_user = 'joe' and price > 10`** limita Joe a ver apenas registros com o preço superior a US$ 10.

Para o uso da CREATE RLS POLICY no Catálogo de Permissões Federadas do Amazon Redshift, consulte Gerenciar o controle de acesso com permissões federadas do Amazon Redshift.

Observações de uso

Ao trabalhar com a instrução CREATE RLS POLICY, observe o seguinte:

- O Amazon Redshift é compatível com filtros que podem fazer parte de uma cláusula WHERE de uma consulta.
- Todas as políticas anexadas a uma tabela devem ter sido criadas com o mesmo alias de tabela.
- Você deve usar as instruções GRANT e REVOKE para conceder e revogar explicitamente permissões SELECT para políticas de RLS que fazem referência a tabelas de pesquisa. Uma tabela de pesquisa é um objeto de tabela usado dentro de uma definição de política. Para obter mais informações, consulte the section called “GRANT” e the section called “REVOKE”.
- A segurança por linha do Amazon Redshift não é compatível com os seguintes tipos de objeto dentro de uma definição de política: tabelas de catálogo, relações entre bancos de dados, tabelas externas, visualizações regulares, visualizações de vinculação tardia, tabelas com políticas RLS ativadas e tabelas temporárias.

Exemplos

O exemplo a seguir cria uma política de RLS chamada policy_concerts. Essa política se aplica a uma coluna VARCHAR (10) chamada catgroup e define o filtro USING para exibir somente as linhas em que o valor de catgroup seja `'Concerts'`.

```
CREATE RLS POLICY policy_concerts
WITH (catgroup VARCHAR(10))
USING (catgroup = 'Concerts');
```

Para ver um exemplo completo de uso de políticas de RLS, consulte Exemplo completo de segurança em nível de linha.

CREATE ROLE

Cria um novo perfil personalizado que é um conjunto de permissões. Para conferir a lista de perfis definidos pelo sistema do Amazon Redshift, consulte the section called “Funções definidas pelo sistema do Amazon Redshift”. Consulte SVV_ROLES para exibir as funções criadas no cluster ou no grupo de trabalho no momento.

Há uma cota do número de funções que podem ser criadas. Para obter mais informações, consulte “Cotas e limites no Amazon Redshift” no Guia de gerenciamento de clusters do Amazon Redshift.

Permissões obrigatórias

A seguir estão os privilégios obrigatórios para CREATE ROLE.

- Superusuário
- Usuários com o privilégio CREATE ROLE

Sintaxe

```
CREATE ROLE role_name
[ EXTERNALID external_id ]
```

Parâmetros

role_name

O nome da função. O nome da função deve ser exclusivo e não pode ser o mesmo que nenhum nome de usuário. Um nome de função não pode ser uma palavra reservada.

Um superusuário ou usuário regular com o privilégio CREATE ROLE pode criar funções. Um usuário que não seja um superusuário, mas que tenha recebido USAGE para a função WITH GRANT OPTION e o privilégio ALTER pode conceder essa função a qualquer pessoa.

EXTERNALID external_id

O identificador para a função, que está associada a um provedor de identidades. Para obter mais informações, consulte Federação do provedor de identidades (IdP) nativo para o Amazon Redshift.

Exemplos

O exemplo a seguir cria uma função `sample_role1`.

```
CREATE ROLE sample_role1;
```

O exemplo a seguir cria uma função `sample_role1` com um ID externo associado a um provedor de identidade.

```
CREATE ROLE sample_role1 EXTERNALID "ABC123";
```

CREATE SCHEMA

Define um novo esquema para o banco de dados atual.

Privilégios obrigatórios

A seguir estão os privilégios obrigatórios para CREATE SCHEMA:

- Superusuário
- Usuários com o privilégio CREATE SCHEMA

Sintaxe

```
CREATE SCHEMA [ IF NOT EXISTS ] schema_name [ AUTHORIZATION username ]
           [ QUOTA {quota [MB | GB | TB] | UNLIMITED} ] [ schema_element [ ... ]

CREATE SCHEMA AUTHORIZATION username[ QUOTA {quota [MB | GB | TB] | UNLIMITED} ]
 [ schema_element [ ... ] ]
```

Parâmetros

IF NOT EXISTS

Cláusula que indica que, se o esquema especificado existe, o comando não deve fazer alterações e deve retornar uma mensagem informando que o esquema existe, em vez de encerrar com um erro.

Esta cláusula é útil para realizar scripting para que o script não falhe se o comando CREATE SCHEMA tentar criar um esquema que já existe.

schema_name

Nome do novo esquema. O nome do esquema não pode ser `PUBLIC`. Para obter mais informações sobre nomes válidos, consulte Nomes e identificadores.

> **Note**
>
> A lista de esquemas no parâmetro de configuração search_path determina a precedência de objetos com nomes idênticos quando forem mencionados sem os nomes dos esquemas.

AUTHORIZATION

Cláusula que dá a propriedade a um usuário especificado.

Nome de usuário do

Nome do proprietário do esquema.

elemento_esquema

Definição de um ou mais objetos a serem criados no esquema.

QUOTA

A quantidade máxima de espaço em disco que o esquema especificado pode usar. Esse espaço é o uso coletivo do disco. Ele inclui todas as tabelas permanentes, visões materializadas do esquema especificado e cópias duplicadas de todas as tabelas com distribuição ALL em cada nó de computação. A cota do esquema não contabiliza tabelas temporárias criadas como parte de um esquema ou namespace temporário.

Para exibir as cotas de esquema configuradas, consulte SVV_SCHEMA_QUOTA_STATE.

Para exibir os registros em que as cotas de esquema foram excedidas, consulte STL_SCHEMA_QUOTA_VIOLATIONS.

O Amazon Redshift converte o valor selecionado para megabytes. Gigabytes é a unidade de medida padrão quando um valor não é especificado.

Você deve ser um superusuário de banco de dados para definir e alterar uma cota de esquema. Um usuário que não seja um superusuário, mas que tenha permissão CREATE SCHEMA, poderá criar um esquema com uma cota definida. Quando você cria um esquema sem definir uma cota, o esquema assume uma cota ilimitada. Quando você define a cota abaixo do valor atual usado pelo esquema, o Amazon Redshift impede novas ingestões até liberar espaço em disco. Uma instrução DELETE exclui os dados de uma tabela e o espaço em disco é liberado somente quando VACUUM é executada.

O Amazon Redshift verifica cada transação em busca de violações de cota antes de confirmar a transação. O Amazon Redshift verifica o tamanho (o espaço em disco usado por todas as tabelas em um esquema) de cada esquema modificado em relação à cota definida. Como a verificação de violação de cota ocorre no final de uma transação, o limite de tamanho poderá exceder temporariamente a cota enquanto a transação não é confirmada. Quando uma transação excede a cota, o Amazon Redshift aborta a transação, proíbe ingestões subsequentes e reverte todas as alterações até você liberar espaço em disco. Devido à VACUUM em segundo plano e limpeza interna, é possível que um esquema não esteja cheio no momento da verificação após uma transação cancelada.

Como exceção, o Amazon Redshift desconsidera a violação da cota e confirma transações em determinados casos. O Amazon Redshift faz isso para transações que consistem apenas em uma ou mais das seguintes instruções, onde não houver uma instrução de consumo INSERT ou COPY na mesma transação:

- DELETE
- TRUNCATE
- VACUUM
- DESCARTAR TABELA
- ALTER TABLE APPEND somente ao mover dados do esquema cheio para outro esquema não cheio

UNLIMITED

O Amazon Redshift não impõe limite para o crescimento do tamanho total do esquema.

Limites

O Amazon Redshift aplica os limites a seguir a esquemas.

- O máximo de esquemas permitido por banco de dados é 9.900.

Exemplos

O exemplo a seguir cria um esquema chamado US_SALES e dá sua propriedade ao usuário DWUSER.

```
create schema us_sales authorization dwuser;
```

O exemplo a seguir cria um esquema chamado US_SALES, dá sua propriedade ao usuário DWUSER e define a cota para 50 GB.

```
create schema us_sales authorization dwuser QUOTA 50 GB;
```

Para exibir o novo esquema, consulte a tabela de catálogo PG_NAMESPACE, conforme mostrado a seguir.

```
select nspname as schema, usename as owner
from pg_namespace, pg_user
where pg_namespace.nspowner = pg_user.usesysid
and pg_user.usename ='dwuser';

   schema |  owner
----------+----------
 us_sales | dwuser
(1 row)
```

O exemplo a seguir cria o esquema US_SALES, ou não realiza ação nenhuma, e retorna uma mensagem se o esquema já existir.

```
create schema if not exists us_sales;
```

CRIAR TABELA

Cria uma nova tabela no banco de dados atual. Você define uma lista de colunas, cada uma contendo dados de um tipo distinto. O proprietário desta tabela é o emissor do comando CREATE TABLE.

Privilégios obrigatórios

A seguir estão os privilégios obrigatórios para CREATE TABLE:

- Superusuário
- Usuários com o privilégio CREATE TABLE

Sintaxe

```
CREATE [ [LOCAL ] { TEMPORARY | TEMP } ] TABLE
[ IF NOT EXISTS ] table_name
( { column_name data_type [column_attributes] [ column_constraints ]
  | table_constraints
  | LIKE parent_table [ { INCLUDING | EXCLUDING } DEFAULTS ] }
  [, ... ]  )
[ BACKUP { YES | NO } ]
[table_attributes]

where column_attributes are:
  [ DEFAULT default_expr ]
  [ IDENTITY ( seed, step ) ]
  [ GENERATED BY DEFAULT AS IDENTITY ( seed, step ) ]
  [ ENCODE encoding ]
  [ DISTKEY ]
  [ SORTKEY ]
  [ COLLATE { CASE_SENSITIVE | CS | CASE_INSENSITIVE | CI } ]

and column_constraints are:
  [ { NOT NULL | NULL } ]
  [ { UNIQUE  |  PRIMARY KEY } ]
  [ REFERENCES reftable [ ( refcolumn ) ] ]

and table_constraints  are:
  [ UNIQUE ( column_name [, ... ] ) ]
  [ PRIMARY KEY ( column_name [, ... ] )  ]
  [ FOREIGN KEY (column_name [, ... ] ) REFERENCES reftable [ ( refcolumn ) ]


and table_attributes are:
  [ DISTSTYLE { AUTO | EVEN | KEY | ALL } ]
  [ DISTKEY ( column_name ) ]
  [ [COMPOUND | INTERLEAVED ] SORTKEY ( column_name [,...]) |  [ SORTKEY AUTO ] ]
  [ ENCODE AUTO ]
```

Parâmetros

LOCAL

Opcional. Embora a instrução aceite esta palavra-chave, ela não tem efeito no Amazon Redshift.

TEMPORARY | TEMP

Palavra-chave que cria uma tabela temporária visível somente na sessão atual. A tabela é automaticamente descartada no final da sessão na qual é criada. A tabela temporária pode ter o mesmo nome de uma tabela permanente. A tabela temporária é criada em um esquema separado e específico da sessão. (Não é possível especificar um nome para este esquema.) Este esquema temporário se torna o primeiro no caminho de pesquisa. Dessa forma, a tabela temporária tem precedência sobre a tabela permanente a menos que você qualifique o nome da tabela com o nome do esquema para acessar a tabela permanente. Para obter mais informações sobre esquemas e precedência, consulte search_path.

> **Note**
>
> Por padrão, os usuários do banco de dados têm permissão para criar tabelas temporárias por associação automática ao grupo PUBLIC. Para negar esse privilégio a um usuário, revogue o privilégio TEMP do grupo PUBLIC, e explicitamente conceda o privilégio TEMP somente a usuários ou grupos específicos de usuários.

IF NOT EXISTS

Cláusula que indica que, se a tabela especificada já existir, o comando não deverá fazer alterações e deverá retornar uma mensagem informando que a tabela existe, em vez de parar com um erro. Observe que a tabela existente pode não parecer em nada com aquela que seria criada; somente o nome da tabela é comparado.

Esta cláusula é útil para realizar scripting para que o script não falhe se o comando CREATE TABLE tentar criar uma tabela que já existe.

table_name

Nome da tabela a ser criada.

Important

Se você especificar um nome de tabela que comece com “#”, a tabela será criada como temporária. Este é um exemplo:

```
create table #newtable (id int);
```

Você também faz referência à tabela com “# ”. Por exemplo:

```
select * from #newtable;
```

O tamanho máximo de um nome de tabela é 127 bytes; nomes mais longos são truncados para ter no máximo 127 bytes. É possível usar caracteres multibyte UFT-8 até um máximo de quatro bytes. O Amazon Redshift aplica uma cota do número de tabelas por cluster por tipo de nó, incluindo tabelas temporárias definidas pelo usuário e tabelas temporárias criadas pelo Amazon Redshift durante o processamento de consultas ou a manutenção do sistema. Como opção, é possível qualificar o nome da tabela com o nome do banco de dados e do esquema. No exemplo a seguir, o nome do banco de dados é `tickit`, o nome do esquema é `public` e o nome da tabela é `test`.

```
create table tickit.public.test (c1 int);
```

Se o banco de dados ou o esquema não existir, a tabela não será criada e a instrução retornará um erro. Você não pode criar tabelas ou exibições nos bancos de dados do sistema `template0`, `template1`, `padb_harvest` ou `sys:internal`.

Se um nome de esquema for fornecido, a nova tabela será criada naquele esquema (presumindo que o criador tenha acesso ao esquema). O nome da tabela deve ser exclusivo para aquele esquema. Se nenhum esquema for especificado, a tabela será criada usando o esquema atual do banco de dados. Se estiver criando uma tabela temporária, você não poderá especificar um nome de esquema, porque as tabelas temporárias existem em um esquema especial.

Podem existir várias tabelas temporárias com o mesmo nome ao mesmo tempo no mesmo banco de dados se elas forem criadas em sessões separadas, uma vez que as tabelas são atribuídas a esquemas diferentes. Para obter mais informações sobre nomes válidos, consulte Nomes e identificadores.

column_name

Nome de uma coluna a ser criada na nova tabela. O tamanho máximo de um nome de coluna é 127 bytes; nomes mais longos são truncados para ter no máximo 127 bytes. É possível usar caracteres multibyte UFT-8 até um máximo de quatro bytes. O número máximo de colunas que você pode definir em uma única tabela é 1.600. Para obter mais informações sobre nomes válidos, consulte Nomes e identificadores.

> **Note**
>
> Se você estiver criando uma “tabela ampla”, tome cuidado para que sua lista de colunas não exceda o limite de largura de linha para resultados intermediários durante cargas e processamento de consultas. Para obter mais informações, consulte Observações de uso.

data_type

Tipo de dados da coluna que está sendo criada. Para as colunas CHAR e VARCHAR, é possível usar a palavra-chave MAX em vez de declarar o tamanho máximo. MAX define o tamanho máximo de CHAR para 4.096 bytes ou de VARCHAR para 65.535 bytes. O tamanho máximo de um objeto GEOMETRY é 1.048.447 bytes.

Para obter mais informações sobre os tipos de dados que o Amazon Redshift aceita, consulte Tipos de dados.

DEFAULT default_expr

Cláusula que atribui um valor de dados padrão à coluna. O tipo de dados expr_padrão deve ser compatível com o tipo de dados da coluna. O valor DEFAULT deve ser uma expressão sem variáveis. Não são permitidas subconsultas, referências cruzadas de outras colunas na tabela atual e funções definidas pelo usuário.

A expressão default_expr é usada em qualquer operação INSERT que não especifique um valor para a coluna. Se não houver um valor padrão especificado, o valor padrão da coluna é nulo.

Se uma operação COPY com uma lista de colunas definida omitir uma coluna com um valor DEFAULT, o comando COPY deve inserir o valor default_expr.

IDENTITY(seed, step)

Cláusula que especifica que a coluna é uma coluna IDENTITY. Uma coluna IDENTITY contém valores exclusivos gerados automaticamente. O tipo de dados para uma coluna IDENTITY deve ser INT ou BIGINT.

Quando você adicionar linhas usando uma instrução `INSERT` ou `INSERT INTO [tablename] VALUES()`, esses valores são iniciados com o valor especificado como seed e o incremento com o número especificado como step.

Quando você carrega a tabela usando uma instrução `INSERT INTO [tablename] SELECT * FROM` ou `COPY`, os dados são carregados em paralelo e distribuídos para as fatias do nó. Para garantir que os valores de identidade sejam exclusivos, o Amazon Redshift ignora diversos valores ao criar valores de identidade. Os valores de identidade são exclusivos, mas a ordem pode não corresponder à ordem nos arquivos de origem.

GENERATED BY DEFAULT AS IDENTITY (seed, step)

Cláusula que especifica que a coluna é uma coluna IDENTITY padrão, permitindo atribuir automaticamente um valor exclusivo à coluna. O tipo de dados para uma coluna IDENTITY deve ser INT ou BIGINT. Quando você adicionar linhas sem valores, esses valores serão iniciados com o valor especificado como seed e o incremento com o número especificado como step. Para obter informações sobre como os valores são gerados, consulte IDENTITY .

Além disso, durante INSERT, UPDATE ou COPY é possível fornecer um valor sem EXPLICIT_IDS. O Amazon Redshift usa esse valor para inseri-lo na coluna de identidade em vez de usar o valor gerado pelo sistema. O valor pode ser uma duplicação, um valor inferior ao da semente ou um valor entre os valores das etapas. O Amazon Redshift não verifica a exclusividade dos valores na coluna. O fornecimento de um valor não afeta o próximo valor gerado pelo sistema.

> **Note**
>
> Se você precisar de exclusividade na coluna, não adicione um valor duplicado. Adicione um valor exclusivo menor do que a semente ou entre os valores das etapas.

Lembre-se do seguinte sobre colunas de identidade padrão:

- As colunas de identidade padrão são NOT NULL. NULL não pode ser inserido.

- Para inserir um valor gerado em uma coluna de identidade padrão, use a palavra-chave DEFAULT.

```
INSERT INTO tablename (identity-column-name) VALUES (DEFAULT);
```

- A substituição de valores de uma coluna de identidade padrão não afeta o próximo valor gerado.
- Não é possível associar uma coluna de identidade padrão com a instrução ALTER TABLE ADD COLUMN.
- É possível associar uma coluna de identidade padrão com a instrução ALTER TABLE APPEND.

ENCODE encoding

Codificação de compactação de uma coluna. ENCODE AUTO é o padrão para tabelas. O Amazon Redshift gerencia automaticamente a codificação de compactação para todas as colunas da tabela. Se você especificar a codificação de compactação para qualquer coluna da tabela, a tabela não será mais definida como ENCODE AUTO. O Amazon Redshift não gerencia mais automaticamente a codificação de compactação para todas as colunas da tabela. Você pode especificar a opção ENCODE AUTO para a tabela para permitir que o Amazon Redshift gerencie automaticamente a codificação de compactação para todas as colunas da tabela.

O Amazon Redshift atribui automaticamente uma codificação de compactação inicial a colunas para as quais você não especifica a codificação de compactação da seguinte maneira:

- Todas as colunas nas tabelas temporárias são atribuídas à compactação RAW como padrão.
- Colunas que são definidas como chaves de classificação são designadas a compactação RAW.
- Colunas que são definidas como tipos de dados BOOLEAN, REAL, DOUBLE PRECISION, GEOMETRY ou GEOGRAPHY recebem a compactação RAW.
- As colunas definidas como SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIME, TIMETZ, TIMESTAMP ou TIMESTAMPTZ recebem a compactação AZ64.
- As colunas definidas como CHAR, VARCHAR ou VARBYTE recebem a compactação LZO.

> **Note**
>
> Se você não quiser que uma coluna seja compactada, especifique explicitamente codificação RAW.

Os seguintes compression encodings (p. 81) são compatíveis:

- AZ64
- BYTEDICT
- DELTA
- DELTA32K
- LZO
- MOSTLY8
- MOSTLY16
- MOSTLY32
- RAW (sem compactação)
- RUNLENGTH
- TEXT255
- TEXT32K
- ZSTD

DISTKEY

Palavra-chave que especifica que a coluna é a chave de distribuição da tabela. Somente uma coluna da tabela pode ser a chave de distribuição. Você pode usar a palavra-chave DISTKEY depois do nome de uma coluna ou como parte da definição da tabela usando a sintaxe DISTKEY (nome_coluna). Qualquer dos métodos tem o mesmo efeito. Para obter mais informações, consulte o parâmetro DISTSTYLE mais adiante neste tópico.

O tipo dos dados de uma coluna de chave de distribuição pode ser: BOOLEAN, REAL, DOUBLE PRECISION, SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIME, TIMETZ, TIMESTAMP ou TIMESTAMPTZ e CHAR ou VARCHAR.

SORTKEY

Palavra-chave que especifica que a coluna é a chave de classificação da tabela. Quando carregados na tabela, os dados são classificados em uma ou mais colunas designadas como chaves de classificação. Você pode usar a palavra-chave SORTKEY depois de um nome de coluna para especificar uma chave de classificação única, ou especificar uma ou mais colunas como colunas de chave de classificação para a tabela usando a sintaxe SORTKEY (nome_coluna [, ...]). Somente chaves de classificação compostas são criadas com essa sintaxe.

Você pode definir um máximo de 400 colunas SORTKEY por tabela.

O tipo dos dados de uma coluna de chave de classificação pode ser: BOOLEAN, REAL, DOUBLE PRECISION, SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIME, TIMETZ, TIMESTAMP ou TIMESTAMPTZ e CHAR ou VARCHAR.

COLLATE { CASE_SENSITIVE | CS | CASE_INSENSITIVE | CI }

Uma cláusula que especifica se a pesquisa ou comparação de string na coluna distingue ou não maiúscula e minúsculas. O valor padrão é o mesmo da configuração atual de diferenciação de maiúsculas e minúsculas do banco de dados.

COLLATE é aceito somente em tipos de dados baseados em string, incluindo CHAR, VARCHAR e valores de string nas colunas SUPER. Para acessar detalhes sobre a consulta de dados SUPER sem distinção entre maiúsculas e minúsculas, consulte Consulta sem distinção entre maiúsculas e minúsculas.

Para verificar o agrupamento atual de um banco de dados, use a função DB_COLLATION.

CASE_SENSITIVE e CS são intercambiáveis e geram os mesmos resultados. Da mesma forma, CASE_INSENSITIVE e CI são intercambiáveis e geram os mesmos resultados.

NOT NULL | NULL

NOT NULL especifica que a coluna não deve conter valores nulos. NULL, o padrão, especifica que a coluna aceita valores nulos. As colunas IDENTITY são declaradas como NOT NULL por padrão.

UNIQUE

Palavra-chave que especifica que a coluna somente pode conter valores exclusivos. O comportamento da restrição exclusiva de tabela é o mesmo das restrições de coluna, com capacidade adicional para abranger diversas colunas. Para definir uma restrição exclusiva de tabela, use a sintaxe UNIQUE (nome_coluna [, …]).

> **Important**
>
> As restrições exclusivas são informativas e não são aplicadas pelo sistema.

PRIMARY KEY

Palavra-chave que especifica que a coluna é a chave primária para a tabela. Somente uma coluna pode ser definida como a chave primária usando uma definição de coluna. Para definir

uma restrição de tabela com uma chave primária de várias colunas, use a sintaxe PRIMARY KEY (nome_coluna [, ...]).

Identificar uma coluna como a chave primária fornece metadados sobre o design do esquema. Uma chave primária implica que outras tabelas podem se basear nesse conjunto de colunas como um identificador exclusivo de linhas. Uma chave primária pode ser especificada para uma tabela, como uma restrição de coluna ou de tabela. A restrição de chave primária deve denominar um conjunto de colunas diferente de outros conjuntos denominados por qualquer restrição exclusiva definida para a mesma tabela.

As colunas PRIMARY KEY também são definidas como NOT NULL.

> **Important**
>
> As restrições de chave primária são somente informativas. Elas não são impostas pelo sistema, mas são usadas pelo planejador.

References tabelaref [(colunaref)]

Cláusula que especifica uma restrição de chave externa, que significa que a coluna deve conter somente valores que correspondam a valores na referida coluna de alguma linha na referida tabela. As colunas referidas devem ser as colunas de uma restrição exclusiva ou de chave primária na tabela referida.

> **Important**
>
> As restrições de chave externa são somente informativas. Elas não são impostas pelo sistema, mas são usadas pelo planejador.

LIKE parent_table [{ INCLUDING | EXCLUDING } DEFAULTS]

Cláusula que especifica uma tabela existente da qual uma nova tabela copia automaticamente nomes de colunas, tipos de dados e restrições NOT NULL. A nova tabela e a tabela pai são desacopladas, e nenhuma alteração realizada na tabela pai é aplicada na nova tabela. As expressões padrão para as definições de colunas copiadas são copiadas somente se INCLUDING DEFAULTS for especificado. O comportamento padrão é excluir expressões padrão para que todas as colunas da nova tabela tenham padrões nulos.

As tabelas criadas com a opção LIKE não herdam restrições de chave primária e externa. O estilo de distribuição, as chaves de classificação e as propriedades BACKUP e NULL são herdados pelas tabelas LIKE, mas você não pode defini-los explicitamente na instrução CREATE TABLE... LIKE.

BACKUP { YES | NO }

Cláusula que especifica se uma tabela deve ser incluída em snapshots de cluster manuais ou automatizadas.

Para tabelas como as de preparação, que não contêm dados críticos, especifique BACKUP NO para economizar tempo de processamento ao criar snapshost, restaurar de snapshots e reduzir o uso de espaço de armazenamento no Amazon Simple Storage Service. A configuração BACKUP NO não tem efeito na replicação automática de dados para outros nós no cluster. Portanto, tabelas com BACKUP NO especificado são restauradas em caso de falha de nó. O padrão é BACKUP YES.

> **Note**
>
> Tabelas sem backup não são aceitas em clusters provisionados com RG ou RA3 e grupos de trabalho do Amazon Redshift sem servidor. Uma tabela marcada como sem backup em um cluster do RG ou RA3 ou um grupo de trabalho com tecnologia sem servidor será tratada como uma tabela permanente da qual sempre será feito backup durante a criação de um snapshot e sempre restaurada quando ocorrer a restauração por meio de um snapshot. Para evitar custos de snapshots em tabelas sem backup, trunque-as antes de tirar um snapshot.

DISTSTYLE { AUTO | EVEN | KEY | ALL }

Palavra-chave que define o estilo de distribuição de dados para toda a tabela. O Amazon Redshift distribui as linhas de uma tabela para cada um dos nós de computação de acordo com o estilo de distribuição especificado para a tabela. O padrão é AUTO.

O estilo de distribuição selecionado para as tabelas afeta a performance geral do seu banco de dados. Para obter mais informações, consulte Distribuição de dados para otimização de consultas. Estilos de distribuição possíveis são os seguintes:

- AUTO: o Amazon Redshift atribui um estilo de distribuição ideal com base nos dados da tabela. Por exemplo, se o estilo de distribuição AUTO for especificado, o Amazon Redshift inicialmente vai atribuir o estilo de distribuição ALL a uma tabela pequena. Quando a tabela

crescer, o Amazon Redshift poderá alterar o estilo de distribuição para KEY, escolhendo a chave primária (ou uma coluna da chave primária composta) como DISTKEY. Se a tabela crescer e nenhuma das colunas for adequada para ser DISTKEY, o Amazon Redshift vai alterar o estilo de distribuição para EVEN. A mudança no estilo de distribuição ocorre em segundo plano com impacto mínimo nas consultas do usuário.

Para visualizar o estilo de distribuição aplicado a uma tabela, consulte a tabela de catálogo de sistema PG_CLASS. Para obter mais informações, consulte Visualização dos estilos de distribuição.

- EVEN: Os dados na tabela são espalhados de maneira uniforme pelos nós em um cluster, em uma distribuição de ida e volta. Os IDs de linha são usados para determinar a distribuição e aproximadamente o mesmo número de linhas é distribuído para cada nó.
- KEY: Os dados são distribuídos pelos valores na coluna DISTKEY. Quando você define as colunas de junção das tabelas de junção como chaves de distribuição, as linhas de junção de ambas as tabelas são dispostas nos nós de computação. Quando os dados são dispostos, o otimizador pode realizar junções de maneira mais eficiente. Se você especificar DISTSTYLE KEY, deve denominar uma coluna DISTKEY, seja para a tabela ou como parte da definição de coluna. Para obter mais informações, consulte o parâmetro DISTKEY mencionado anteriormente neste tópico.
- ALL: Uma cópia de toda a tabela é distribuída para cada nó. Este estilo de distribuição garante que todas as linhas necessárias para qualquer junção estejam disponíveis em cada nó, mas multiplica os requisitos de armazenamento e aumenta a carga e os tempos de manutenção da tabela. A distribuição ALL pode melhorar o tempo de execução quando usada com determinadas tabelas de dimensão nas quais a distribuição KEY não é apropriada, mas as melhorias de performance devem ser comparadas aos custos de manutenção.

DISTKEY (nome_coluna)

Restrição que especifica a coluna a ser usada como a chave de distribuição da tabela. Você pode usar a palavra-chave DISTKEY depois do nome de uma coluna ou como parte da definição da tabela usando a sintaxe DISTKEY (nome_coluna). Qualquer dos métodos tem o mesmo efeito. Para obter mais informações, consulte o parâmetro DISTSTYLE mencionado anteriormente neste tópico.

[COMPOUND | INTERLEAVED] SORTKEY (column_name [,...]) | [SORTKEY AUTO]

Especifica uma ou mais chaves de classificação da tabela. Quando carregados na tabela, os dados são classificados pelas colunas designadas como chaves de classificação. Você pode usar a palavra-chave SORTKEY depois de um nome de coluna para especificar uma chave de

classificação única, ou especificar uma ou mais colunas como colunas de chave de classificação para a tabela usando a sintaxe `SORTKEY (column_name [ , ... ] )`.

Como opção, você pode especificar um estilo de classificação COMPOUND ou INTERLEAVED. Se você especificar SORTKEY com colunas, o padrão será COMPOUND. Para obter mais informações, consulte Chaves de classificação.

Se você não especificar nenhuma opção de chave de classificação, o padrão será AUTO.

Você pode definir um máximo de 400 colunas COMPOUND SORTKEY ou 8 colunas INTERLEAVED SORTKEY por tabela.

AUTO

Especifica que o Amazon Redshift atribui uma chave de classificação ideal com base nos dados da tabela. Por exemplo, se a chave de classificação AUTO for especificada, o Amazon Redshift inicialmente não atribuirá nenhuma chave de classificação a uma tabela. Se o Amazon Redshift determinar que uma nova chave de classificação melhorará a performance das consultas, o Amazon Redshift poderá alterar a chave de classificação da sua tabela. A classificação real da tabela é feita pela classificação automática da tabela. Para obter mais informações, consulte Classificação automática de tabela.

O Amazon Redshift não modifica tabelas que possuem chaves de classificação ou distribuição existentes. Com uma exceção, se uma tabela tiver uma chave de distribuição que nunca foi usada em um JOIN, a chave poderá ser alterada se o Amazon Redshift determinar que há uma chave melhor.

Para exibir a chave de classificação de uma tabela, consulte a visualização do catálogo do sistema SVV_TABLE_INFO. Para obter mais informações, consulte SVV_TABLE_INFO. Para exibir as recomendações do Amazon Redshift Advisor para tabelas, consulte a visualização do catálogo do sistema SVV_ALTER_TABLE_RECOMMENDATIONS. Para obter mais informações, consulte SVV_ALTER_TABLE_RECOMMENDATIONS. Para exibir as ações executadas pelo Amazon Redshift, consulte a visualização do catálogo do sistema SVL_AUTO_WORKER_ACTION. Para obter mais informações, consulte SVL_AUTO_WORKER_ACTION.

COMPOUND

Especifica que os dados sejam classificados usando uma chave composta por todas as colunas listadas, na ordem em que estão listadas. Uma chave de classificação composta é mais útil quando uma consulta faz a varredura de linhas de acordo com a ordem das colunas

classificadas. Os benefícios de performance da classificação com uma chave composta diminuem quando as consultas se baseiam em colunas de classificação secundárias. Você pode definir um máximo de 400 colunas COMPOUND SORTKEY por tabela.

INTERLEAVED

Especifica que os dados sejam classificados usando uma chave de classificação intercalada. Um máximo de oito colunas pode ser especificado para uma chave de classificação intercalada.

Uma classificação intercalada concede peso igual a todas as colunas, ou subconjuntos de colunas, na chave de classificação para que as consultas não dependam da ordem das colunas na chave de classificação. Quando uma consulta usa uma ou mais colunas de classificação secundárias, a classificação intercalada apresenta uma melhoria significativa de performance de consulta. A classificação intercalada carrega um pequeno custo de sobrecarga para operações de carregamento de dados e limpeza de operações.

> **Important**
>
> Não use uma chave de classificação intercalada em colunas com atributos que aumentam monotonicamente, como colunas de identidade, datas ou timestamps.

ENCODE AUTO

Permite que o Amazon Redshift ajuste automaticamente o tipo de codificação de todas as colunas da tabela para otimizar a performance da consulta. O ENCODE AUTO preserva os tipos de codificação iniciais especificados na criação da tabela. Em seguida, se o Amazon Redshift determinar que um novo tipo de codificação pode melhorar a performance da consulta, o Amazon Redshift poderá alterar o tipo de codificação das colunas da tabela. ENCODE AUTO é o padrão se você não especificar um tipo de codificação em qualquer coluna na tabela.

UNIQUE (nome_coluna [,...])

Restrição que especifica que um grupo contendo uma ou mais colunas de uma tabela pode conter somente valores exclusivos. O comportamento da restrição exclusiva de tabela é o mesmo das restrições de coluna, com capacidade adicional para abranger diversas colunas. No contexto de restrições exclusivas, valores nulos não são considerados iguais. Cada restrição de tabela exclusiva deve denominar um conjunto de colunas diferente do conjunto de colunas denominado por qualquer restrição exclusiva ou de chave primária definida para a tabela.

> **Important**
>
> As restrições exclusivas são informativas e não são aplicadas pelo sistema.

PRIMARY KEY (nome_coluna [,...])

Restrição que especifica que uma ou mais colunas de uma tabela pode conter somente valores exclusivos (não copiados) não nulos. Identificar um conjunto de colunas como a chave primária também fornece metadados sobre o design do esquema. Uma chave primária implica que outras tabelas podem se basear nesse conjunto de colunas como um identificador exclusivo de linhas. Uma chave primária pode ser especificada para uma tabela, como uma restrição única de coluna ou de tabela. A restrição de chave primária deve denominar um conjunto de colunas diferente de outros conjuntos denominados por qualquer restrição exclusiva definida para a mesma tabela.

> **Important**
>
> As restrições de chave primária são somente informativas. Elas não são impostas pelo sistema, mas são usadas pelo planejador.

FOREIGN KEY (nome_coluna [, ...]) REFERENCES tabelaref [(colunaref)]

Restrição que especifica uma restrição de chave externa, que exige que um grupo contendo uma ou mais colunas da nova tabela deve apresentar somente valores que correspondam aos valores na(s) coluna(s) mencionada(s) de alguma linha na tabela referida. Se colunaref for omitido, a chave primária de tabelaref será usada. As colunas mencionadas devem ser as colunas de uma restrição exclusiva ou de chave primária na tabela referida.

> **Important**
>
> As restrições de chave externa são somente informativas. Elas não são impostas pelo sistema, mas são usadas pelo planejador.

Observações de uso

As restrições de exclusividade, chave primária e chave estrangeira são apenas informativas; elas não são impostas pelo Amazon Redshift quando você preenche uma tabela. Por exemplo, se você

inserir dados em uma tabela com dependências, a inserção poderá ser bem-sucedida mesmo que viole a restrição. Apesar disso, chaves primárias e chaves estrangeiras são usadas como sugestões de planejamento e devem ser declaradas se seu processo de ETL ou algum outro processo em seu aplicativo reforçar sua integridade. Para obter informações sobre como descartar uma tabela com dependências, consulte DESCARTAR TABELA.

Limites e cotas

Considere os seguintes limites ao criar uma tabela.

- Há um limite para o número máximo de tabelas em um cluster por tipo de nó. Para obter mais informações, consulte "Limites" no Guia de gerenciamento de clusters do Amazon Redshift.
- O número máximo de caracteres para um nome de tabela é 127.
- O número máximo de colunas que você pode definir em uma única tabela é 1.600.
- O número máximo de colunas SORTKEY que você pode definir em uma única tabela é 400.

Resumo de configurações de nível de coluna e de nível de tabela

Diversos atributos e configurações podem ser definidos no nível da coluna ou no nível da tabela. Em alguns casos, configurar um atributo ou restrição no nível da coluna ou da tabela tem o mesmo efeito. Em outros casos, produz resultados diferentes.

A lista a seguir resume configurações nos níveis da coluna e da tabela::

DISTKEY

Não há diferenças no efeito, seja definido no nível da coluna ou no nível da tabela.

Se DISTKEY for definido, no nível da coluna ou no nível da tabela, DISTSTYLE deve ser definido como KEY ou não ser definido. DISTSTYLE pode ser definido somente no nível da tabela.

SORTKEY

Se definido no nível de coluna, SORTKEY deve ser uma coluna única. Se SORTKEY for definido no nível da tabela, uma ou mais colunas podem gerar uma chave de classificação composta ou intercalada.

COLLATE CASE_SENSITIVE | COLLATE CASE_INSENSITIVE

O Amazon Redshift não permite a alteração da configuração de diferenciação de maiúsculas e minúsculas para uma coluna. Quando você anexa uma nova coluna à tabela, o Amazon Redshift

usa o valor padrão para diferenciação de maiúsculas e minúsculas. O Amazon Redshift não oferece suporte à palavra-chave COLLATE ao anexar uma nova coluna.

Para obter informações sobre como criar banco de dados usando o agrupamento de banco de dados, consulte CREATE DATABASE.

Para obter mais informações sobre funções COLLATE, consulte Função COLLATE.

UNIQUE

No nível de coluna, uma ou mais chaves podem ser definidas como UNIQUE; a restrição UNIQUE se aplica a cada coluna individualmente. Se UNIQUE for definida no nível da tabela, uma ou mais colunas podem gerar uma restrição UNIQUE composta.

PRIMARY KEY

Se definida no nível de coluna, PRIMARY KEY deve ser uma coluna única. Se PRIMARY KEY for definida no nível da tabela, uma ou mais colunas podem gerar uma chave primária composta.

FOREIGN KEY

Não há diferenças no efeito seja a FOREIGN KEY definida no nível da coluna ou no nível da tabela. No nível da coluna, a sintaxe é simplesmente `REFERENCES` tabelaref [(colunaref)].

Distribuição de dados de entrada

Quando o esquema de distribuição de hash de dados de entrada corresponder aos dados da tabela de destino, nenhuma distribuição física de dados é realmente necessária quando os dados são carregados. Por exemplo, se uma chave de distribuição for definida para a nova tabela e os dados estiverem sendo introduzidos a partir de outra tabela que é distribuída na mesma coluna chave, os dados serão carregados no local, usando os mesmos nós e fatias. No entanto, se as tabelas de origem e de destino forem definidas com a distribuição EVEN, os dados serão redistribuídos em uma tabela de destino.

Tabelas largas

Você pode conseguir criar uma tabela muito larga, mas não conseguir realizar processamento de consultas, como as instruções INSERT ou SELECT, na tabela. A largura máxima de uma tabela com colunas de largura fixa, como CHAR, é de 64 KB - 1 (ou 65.535 bytes). Se a tabela incluir colunas VARCHAR, ela poderá ter uma largura declarada maior sem retornar um erro, pois as colunas VARCHARS não contribuem com sua largura total declarada para o limite de processamento de

consulta calculado. O limite de processamento de consultas efetivo em colunas VARCHAR varia com base em diversos fatores.

Se uma tabela for muito larga para ser inserida ou selecionada, você receberá o erro a seguir.

```
ERROR:  8001
DETAIL:  The combined length of columns processed in the SQL statement
exceeded the query-processing limit of 65535 characters (pid:7627)
```

Exemplos

Para exemplos que mostram como usar o comando CREATE TABLE, confira o tópico Exemplos.

Exemplos

Os exemplos a seguir demonstram vários atributos de coluna e tabela nas instruções CREATE TABLE do Amazon Redshift. Para obter mais informações sobre CREATE TABLE, incluindo definições de parâmetros, consulte CRIAR TABELA.

Muitos dos exemplos usam tabelas e dados do conjunto de dados de amostra TICKIT. Para obter mais informações, consulte Banco de dados de amostra.

Você pode prefixar o nome da tabela com o nome do banco de dados e o nome do esquema em um comando CREATE TABLE. Por exemplo, `dev_database.public.sales`. O nome do banco de dados deve ser o banco de dados ao qual você se conectou. Qualquer tentativa de criar objetos de banco de dados em outro banco de dados falha com um erro de operação inválida.

Criar uma tabela com uma chave de distribuição, uma chave composta de classificação e compactação

O exemplo a seguir cria uma tabela SALES no banco de dados TICKIT com a compactação definida para várias colunas. O LISTID é declarado como a chave de distribuição, e o LISTID e o SELLERID são declarados como chave de classificação composta com várias colunas. As restrições de chave primária e de chave externa também são definidas para a tabela. Antes de criar a tabela no exemplo, talvez seja necessário adicionar uma restrição UNIQUE a cada coluna referenciada por uma chave estrangeira, caso não existam restrições.

```
create table sales(
salesid integer not null,
```

```
listid integer not null,
sellerid integer not null,
buyerid integer not null,
eventid integer not null encode mostly16,
dateid smallint not null,
qtysold smallint not null encode mostly8,
pricepaid decimal(8,2) encode delta32k,
commission decimal(8,2) encode delta32k,
saletime timestamp,
primary key(salesid),
foreign key(listid) references listing(listid),
foreign key(sellerid) references users(userid),
foreign key(buyerid) references users(userid),
foreign key(dateid) references date(dateid))
distkey(listid)
compound sortkey(listid,sellerid);
```

Os resultados são os seguintes:

```
schemaname | tablename | column     | type                          | encoding | distkey
 | sortkey | notnull
-----------+-----------+------------+-------------------------------+----------+---------
+---------+--------
public     | sales     | salesid    | integer                       | lzo      | false
 |       0 | true
public     | sales     | listid     | integer                       | none     | true
 |       1 | true
public     | sales     | sellerid   | integer                       | none     | false
 |       2 | true
public     | sales     | buyerid    | integer                       | lzo      | false
 |       0 | true
public     | sales     | eventid    | integer                       | mostly16 | false
 |       0 | true
public     | sales     | dateid     | smallint                      | lzo      | false
 |       0 | true
public     | sales     | qtysold    | smallint                      | mostly8  | false
 |       0 | true
public     | sales     | pricepaid  | numeric(8,2)                  | delta32k | false
 |       0 | false
public     | sales     | commission | numeric(8,2)                  | delta32k | false
 |       0 | false
public     | sales     | saletime   | timestamp without time zone | lzo      | false
 |       0 | false
```

O exemplo a seguir cria a tabela t1 com uma coluna col1 que não diferencia maiúsculas e minúsculas.

```
create table T1 (
  col1 Varchar(20) collate case_insensitive
 );

insert into T1 values ('bob'), ('john'), ('Tom'), ('JOHN'), ('Bob');
```

Consulte a tabela:

```
select * from T1 where col1 = 'John';

col1
------
 john
 JOHN
(2 rows)
```

Criar uma tabela usando uma chave de classificação intercalada

O exemplo a seguir cria a tabela CUSTOMER com uma chave de classificação intercalada.

```
create table customer_interleaved (
  c_custkey      integer        not null,
  c_name         varchar(25)    not null,
  c_address      varchar(25)    not null,
  c_city         varchar(10)    not null,
  c_nation       varchar(15)    not null,
  c_region       varchar(12)    not null,
  c_phone        varchar(15)    not null,
  c_mktsegment      varchar(10)    not null)
diststyle all
interleaved sortkey (c_custkey, c_city, c_mktsegment);
```

Criar uma tabela usando IF NOT EXISTS

O exemplo a seguir cria a tabela CITIES ou não realiza ação nenhuma e retorna uma mensagem se a tabela já existir:

```
create table if not exists cities(
```

```
cityid integer not null,
city varchar(100) not null,
state char(2) not null);
```

Criar uma tabela com a distribuição ALL

O exemplo a seguir cria a tabela VENUE com a distribuição ALL.

```
create table venue(
venueid smallint not null,
venuename varchar(100),
venuecity varchar(30),
venuestate char(2),
venueseats integer,
primary key(venueid))
diststyle all;
```

Criar uma tabela com a distribuição EVEN

O exemplo a seguir cria uma tabela denominada MYEVENT com três colunas.

```
create table myevent(
eventid int,
eventname varchar(200),
eventcity varchar(30))
diststyle even;
```

A tabela é distribuída de maneira uniforme e não é classificada. A tabela não tem qualquer coluna DISTKEY ou SORTKEY declarada.

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'myevent';

  column   |          type          | encoding | distkey | sortkey
-----------+------------------------+----------+---------+---------
 eventid   | integer                | lzo      | f       |       0
 eventname | character varying(200) | lzo      | f       |       0
 eventcity | character varying(30)  | lzo      | f       |       0
(3 rows)
```

Criar uma tabela temporária que se seja LIKE em relação a outra tabela

O exemplo a seguir cria uma tabela temporária denominada TEMPEVENT, que herda as colunas da tabela EVENT.

```
create temp table tempevent(like event);
```

Essa tabela também herda os atributos DISTKEY e SORTKEY da tabela pai:

```
select "column", type, encoding, distkey, sortkey
 from pg_table_def where tablename = 'tempevent';

  column   |            type             | encoding | distkey | sortkey
-----------+-----------------------------+----------+---------+---------
 eventid   | integer                     | none     | t       |       1
 venueid   | smallint                    | none     | f       |       0
 catid     | smallint                    | none     | f       |       0
 dateid    | smallint                    | none     | f       |       0
 eventname | character varying(200)      | lzo      | f       |       0
 starttime | timestamp without time zone | bytedict | f       |       0
(6 rows)
```

Criar uma tabela com uma coluna IDENTITY

O exemplo a seguir cria uma tabela denominada VENUE_IDENT, com uma coluna IDENTITY denominada VENUEID. A coluna inicia com 0 e é incrementada por 1 para cada registro. VENUEID é também declarada como a chave primária da tabela.

```
create table venue_ident(venueid bigint identity(0, 1),
venuename varchar(100),
venuecity varchar(30),
venuestate char(2),
venueseats integer,
primary key(venueid));
```

Criar uma tabela com uma coluna IDENTITY padrão

O exemplo a seguir cria uma tabela chamada `t1`. Essa tabela tem uma coluna IDENTITY denominada `hist_id` e uma coluna IDENTITY padrão denominada `base_id`.

```
CREATE TABLE t1(
```

```
  hist_id BIGINT IDENTITY NOT NULL, /* Cannot be overridden */
  base_id BIGINT GENERATED BY DEFAULT AS IDENTITY NOT NULL, /* Can be overridden */
  business_key varchar(10) ,
  some_field varchar(10)
);
```

A inserção de uma linha na tabela mostra que os valores `hist_id` e `base_id` foram gerados.

```
INSERT INTO T1 (business_key, some_field) values ('A','MM');
```

```
SELECT * FROM t1;

 hist_id | base_id | business_key | some_field
---------+---------+--------------+------------
       1 |       1 | A            | MM
```

A inserção de uma segunda linha mostra que o valor padrão de `base_id` foi gerado.

```
INSERT INTO T1 (base_id, business_key, some_field) values (DEFAULT, 'B','MNOP');
```

```
SELECT * FROM t1;

 hist_id | base_id | business_key | some_field
---------+---------+--------------+------------
       1 |       1 | A            | MM
       2 |       2 | B            | MNOP
```

A inserção de uma terceira linha mostra que o valor de `base_id` não precisa ser exclusivo.

```
INSERT INTO T1 (base_id, business_key, some_field) values (2,'B','MNNN');
```

```
SELECT * FROM t1;

 hist_id | base_id | business_key | some_field
---------+---------+--------------+------------
       1 |       1 | A            | MM
       2 |       2 | B            | MNOP
       3 |       2 | B            | MNNN
```

Criar uma tabela com valores DEFAULT de coluna

O exemplo a seguir cria uma tabela CATEGORYDEF que declara valores padrão para cada coluna:

```
create table categorydef(
catid smallint not null default 0,
catgroup varchar(10) default 'Special',
catname varchar(10) default 'Other',
catdesc varchar(50) default 'Special events',
primary key(catid));

insert into categorydef values(default,default,default,default);
```

```
select * from categorydef;

 catid | catgroup | catname |    catdesc
-------+----------+---------+----------------
     0 | Special  | Other   | Special events
(1 row)
```

Opções DISTSTYLE, DISTKEY e SORTKEY

O exemplo a seguir mostra como as opções DISTKEY, SORTKEY e DISTSTYLE funcionam. Neste exemplo, COL1 é a chave de distribuição. Portanto, o estilo de distribuição deve ser definido como KEY ou não deve ser definido. Por padrão, a tabela não tem chave de classificação e, portanto, não é classificada:

```
create table t1(col1 int distkey, col2 int) diststyle key;
```

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 't1';

column |  type   | encoding | distkey | sortkey
-------+---------+----------+---------+---------
col1   | integer | az64     | t       | 0
col2   | integer | az64     | f       | 0
```

No exemplo, a mesma coluna é definida como a chave de distribuição e a chave de classificação. Além disso, o estilo de distribuição deve ser definido como KEY ou não deve ser definido.

```
create table t2(col1 int distkey sortkey, col2 int);
```

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 't2';

column |  type   | encoding | distkey | sortkey
-------+---------+----------+---------+---------
col1   | integer | none     | t       | 1
col2   | integer | az64     | f       | 0
```

No exemplo, nenhuma coluna é definida como chave de distribuição, COL2 é definida como chave de classificação e o estilo de distribuição é definido como ALL:

```
create table t3(col1 int, col2 int sortkey) diststyle all;
```

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 't3';

Column |  Type   | Encoding | DistKey | SortKey
-------+---------+----------+---------+--------
col1   | integer | az64     | f       | 0
col2   | integer | none     | f       | 1
```

No exemplo, o estilo de distribuição é definido como EVEN e nenhuma chave de classificação é explicitamente definida. Portanto, a tabela é distribuída de maneira uniforme, mas não é classificada.

```
create table t4(col1 int, col2 int) diststyle even;
```

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 't4';

             column |  type   |encoding | distkey | sortkey
--------+---------+---------+---------+--------
col1    | integer | az64    | f       | 0
col2    | integer | az64    | f       | 0
```

Criar uma tabela com a opção ENCODE AUTO

O exemplo a seguir cria a tabela `t1` com codificação de compactação automática. ENCODE AUTO é o padrão para tabelas quando você não especifica um tipo de codificação para qualquer coluna.

```
create table t1(c0 int, c1 varchar);
```

O exemplo a seguir cria a tabela `t2` com codificação de compactação automática especificando ENCODE AUTO.

```
create table t2(c0 int, c1 varchar) encode auto;
```

O exemplo a seguir cria a tabela `t3` com codificação de compactação automática especificando ENCODE AUTO. A coluna `c0` é definida com um tipo de codificação inicial de DELTA. O Amazon Redshift pode alterar a codificação se outra codificação fornecer melhor performance de consulta.

```
create table t3(c0 int encode delta, c1 varchar) encode auto;
```

O exemplo a seguir cria a tabela `t4` com codificação de compactação automática especificando ENCODE AUTO. A coluna `c0` é definida com uma codificação inicial de DELTA, e coluna `c1` é definida com uma codificação inicial de LZO. O Amazon Redshift pode alterar a codificação se outra codificação fornecer melhor performance de consulta.

```
create table t4(c0 int encode delta, c1 varchar encode lzo) encode auto;
```

CREATE TABLE AS

Tópicos

- Sintaxe
- Parâmetros
- Observações de uso CTAS
- Exemplos de CTAS

Cria uma nova tabela com base em uma consulta. O proprietário desta tabela é o usuário que emite o comando.

A nova tabela é carregada com dados definidos pela consulta no comando. As colunas da tabela têm nomes e tipos de dados associados às colunas de saída da consulta. O comando CREATE TABLE AS (CTAS) cria uma nova tabela e avalia a consulta para carregar a nova tabela.

Sintaxe

```
CREATE [ [ LOCAL ] { TEMPORARY | TEMP } ]
TABLE table_name
[ ( column_name [, ... ] ) ]
[ BACKUP { YES | NO } ]
[ table_attributes ]
AS query

where table_attributes are:
[ DISTSTYLE { AUTO | EVEN | ALL | KEY } ]
[ DISTKEY( distkey_identifier ) ]
[ [ COMPOUND | INTERLEAVED ] SORTKEY( column_name [, ...] ) ]
```

Parâmetros

LOCAL

Embora a instrução aceite esta palavra-chave, ela não tem efeito no Amazon Redshift.

TEMPORARY | TEMP

Cria uma tabela temporária. Uma tabela temporária é automaticamente descartada no final da sessão na qual ela foi criada.

table_name

Nome da tabela a ser criada.

> **Important**
>
> Se você especificar um nome de tabela que comece com "#", a tabela será criada como temporária. Por exemplo:
>
> ```
> create table #newtable (id) as select * from oldtable;
> ```

O tamanho máximo do nome da tabela é 127 bytes; nomes mais longos são truncados para ter no máximo 127 bytes. O Amazon Redshift impõe uma cota do número de tabelas por cluster por tipo de nó. É possível qualificar o nome da tabela com o nome do banco de dados e do esquema, conforme mostra a tabela a seguir.

```
create table tickit.public.test (c1) as select * from oldtable;
```

Neste exemplo, `tickit` é o nome do banco de dados e `public` é o nome do esquema. Se o banco de dados ou o esquema não existir, a instrução retornará um erro.

Se um nome de esquema for fornecido, a nova tabela será criada naquele esquema (presumindo que o criador tenha acesso ao esquema). O nome da tabela deve ser exclusivo para aquele esquema. Se nenhum esquema for especificado, a tabela será criada usando o esquema atual do banco de dados. Se estiver criando uma tabela temporária, você não poderá especificar um nome de esquema, uma vez que as tabelas temporárias existem em um esquema especial.

Podem existir várias tabelas temporárias com o mesmo nome ao mesmo tempo no mesmo banco de dados, se elas forem criadas em sessões separadas. Essas tabelas são atribuídas a esquemas diferentes.

column_name

Nome de uma coluna na nova tabela. Se nenhum nome de coluna for fornecido, os nomes de colunas podem ser obtidos com base nos nomes de coluna de saída da consulta. Os nomes de coluna padrão são usados para expressões. Para obter mais informações sobre nomes válidos, consulte Nomes e identificadores.

BACKUP { YES | NO }

Cláusula que especifica se uma tabela deve ser incluída em snapshots de cluster manuais ou automatizadas.

Para tabelas como as de preparação, que não contêm dados críticos, especifique BACKUP NO para economizar tempo de processamento ao criar snapshost, restaurar de snapshots e reduzir o uso de espaço de armazenamento no Amazon Simple Storage Service. A configuração BACKUP NO não tem efeito em replicação automática de dados para outros nós no cluster. Portanto, tabelas com BACKUP NO especificado são restauradas em caso de erro de nó. O padrão é BACKUP YES.

> **Note**
>
> Tabelas sem backup não são aceitas em clusters provisionados com RG ou RA3 e grupos de trabalho do Amazon Redshift sem servidor. Uma tabela marcada como sem backup em um cluster do RG ou RA3 ou um grupo de trabalho com tecnologia sem servidor será tratada como uma tabela permanente da qual sempre será feito backup

durante a criação de um snapshot e sempre restaurada quando ocorrer a restauração por meio de um snapshot. Para evitar custos de snapshots em tabelas sem backup, trunque-as antes de tirar um snapshot.

DISTSTYLE { AUTO | EVEN | KEY | ALL }

Palavra-chave que define o estilo de distribuição de dados para toda a tabela. O Amazon Redshift distribui as linhas de uma tabela para cada um dos nós de computação de acordo com o estilo de distribuição especificado para a tabela. O padrão é DISTSTYLE AUTO.

O estilo de distribuição selecionado para as tabelas afeta a performance geral do seu banco de dados. Para obter mais informações, consulte Distribuição de dados para otimização de consultas.

- AUTO: o Amazon Redshift atribui um estilo de distribuição ideal com base nos dados da tabela. Para visualizar o estilo de distribuição aplicado a uma tabela, consulte a tabela de catálogo de sistema PG_CLASS. Para obter mais informações, consulte Visualização dos estilos de distribuição.
- EVEN: Os dados na tabela são espalhados de maneira uniforme pelos nós em um cluster, em uma distribuição de ida e volta. Os IDs de linha são usados para determinar a distribuição e aproximadamente o mesmo número de linhas é distribuído para cada nó. Este método de distribuição é o padrão.
- KEY: Os dados são distribuídos pelos valores na coluna DISTKEY. Quando você define as colunas de junção das tabelas de junção como chaves de distribuição, as linhas de junção de ambas as tabelas são dispostas nos nós de computação. Quando os dados são dispostos, o otimizador pode realizar junções de maneira mais eficiente. Se você especificar a DISTSTYLE KEY, é preciso nomear uma coluna DISTKEY.
- ALL: Uma cópia de toda a tabela é distribuída para cada nó. Este estilo de distribuição garante que todas as linhas necessárias para qualquer junção estejam disponíveis em cada nó, mas multiplica os requisitos de armazenamento e aumenta a carga e os tempos de manutenção da tabela. A distribuição ALL pode melhorar o tempo de execução quando usada com determinadas tabelas de dimensão nas quais a distribuição KEY não é apropriada, mas as melhorias de performance devem ser comparadas aos custos de manutenção.

DISTKEY (coluna)

Especifica um nome de coluna ou um número de posição para a chave de distribuição. Use o nome especificado na lista opcional de coluna para a tabela ou na lista selecionada da consulta.

Como opção, você pode usar um número de posição, onde a primeira coluna selecionada é 1, a segunda é 2 e assim por diante. Somente uma coluna da tabela pode ser a chave de distribuição:

- Se você declarar uma coluna como DISTKEY, DISTSTYLE deve ser definida como KEY ou não deve ser definida.
- Se você não declarar uma coluna DISTKEY, é possível definir DISTSTYLE como EVEN.
- Se você não especificar DISTKEY ou DISTSTYLE, o CTAS determina o estilo de distribuição para a nova tabela com base no plano de consulta para a cláusula SELECT. Para obter mais informações, consulte Herança de atributos de coluna e tabela.

Você pode definir a mesma coluna como chave de distribuição e chave de classificação. Esta abordagem tende a acelerar a junção quando a coluna em questão é uma coluna de junção na consulta.

[COMPOUND | INTERLEAVED] SORTKEY (column_name [, ...])

Especifica uma ou mais chaves de classificação da tabela. Quando carregados na tabela, os dados são classificados pelas colunas designadas como chaves de classificação.

Como opção, você pode especificar um estilo de classificação COMPOUND ou INTERLEAVED. O padrão é COMPOUND. Para obter mais informações, consulte Chaves de classificação.

Você pode definir um máximo de 400 colunas COMPOUND SORTKEY ou 8 colunas INTERLEAVED SORTKEY por tabela.

Se você não especificar SORTKEY, o CTAS determina as chaves de classificação para a nova tabela com base no plano de consulta para a cláusula SELECT. Para obter mais informações, consulte Herança de atributos de coluna e tabela.

COMPOUND

Especifica que os dados sejam classificados usando uma chave composta por todas as colunas listadas, na ordem em que estão listadas. Uma chave de classificação composta é mais útil quando uma consulta faz a varredura de linhas de acordo com a ordem das colunas classificadas. Os benefícios de performance da classificação com uma chave composta diminuem quando as consultas se baseiam em colunas de classificação secundárias. Você pode definir um máximo de 400 colunas COMPOUND SORTKEY por tabela.

INTERLEAVED

Especifica que os dados sejam classificados usando uma chave de classificação intercalada. Um máximo de oito colunas pode ser especificado para uma chave de classificação intercalada.

Uma classificação intercalada concede peso igual a todas as colunas, ou subconjuntos de colunas, na chave de classificação para que as consultas não dependam da ordem das colunas na chave de classificação. Quando uma consulta usa uma ou mais colunas de classificação secundárias, a classificação intercalada apresenta uma melhoria significativa de performance de consulta. A classificação intercalada carrega um pequeno custo de sobrecarga para operações de carregamento de dados e limpeza de operações.

AS query

Qualquer consulta (instrução SELECT) compatível com o Amazon Redshift.

Observações de uso CTAS

Limites

O Amazon Redshift impõe uma cota do número de tabelas por cluster por tipo de nó.

O número máximo de caracteres para um nome de tabela é 127.

O número máximo de colunas que você pode definir em uma única tabela é 1.600.

Herança de atributos de coluna e tabela

Tabelas CREATE TABLE AS (CTAS) não herdam restrições, colunas de identidade, valores de coluna padrão ou a chave primária da tabela com base na qual foram criadas.

Você não pode especificar codificações de compactação de coluna para tabelas CTAS. O Amazon Redshift atribui automaticamente a codificação de compactação da seguinte maneira:

- Colunas que são definidas como chaves de classificação são designadas a compactação RAW.
- Colunas que são definidas como tipos de dados BOOLEAN, REAL, DOUBLE PRECISION, GEOMETRY ou GEOGRAPHY recebem a compactação RAW.
- As colunas definidas como SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIME, TIMETZ, TIMESTAMP ou TIMESTAMPTZ recebem a compactação AZ64.
- As colunas definidas como CHAR, VARCHAR ou VARBYTE recebem a compactação LZO.

Para obter mais informações, consulte Codificações de compactação e Tipos de dados.

Para atribuir explicitamente codificações de coluna, use CRIAR TABELA.

O CTAS determina o estilo de distribuição e a chave de classificação para a nova tabela com base no plano de consulta para a cláusula SELECT.

Para consultas complexas, como consultas incluindo junções, agregações, classificação por cláusula ou uma cláusula de limite, o CTAS se esforça para escolher o melhor estilo de distribuição e chave de classificação com base no plano de consulta.

> **Note**
>
> Para uma melhor performance com grandes conjuntos de dados ou consultas complexas, recomendamos testar usando conjuntos de dados típicos.

Muitas vezes você pode prever qual chave de distribuição e chave de classificação o CTAS escolhe examinando o plano de consulta para ver quais colunas, se houver, o otimizador de consulta escolherá para classificação e distribuição de dados. Se o nó superior do plano de consulta for uma varredura sequencial simples de uma tabela única (XN Seq Scan), o CTAS usará o estilo de distribuição e a chave de classificação da tabela de origem. Se o nó superior do plano de consulta for qualquer outra varredura sequencial (como XN Limit, XN Sort, XN HashAggregate, etc.), o CTAS se esforça para escolher o melhor estilo de distribuição e chave de classificação com base no plano de consulta.

Por exemplo, suponhamos que você crie cinco tabelas usando os seguintes tipos de cláusulas SELECT:

- Um instrução simples de seleção
- Uma cláusula de limitação
- Um classificação por cláusula usando LISTID
- Um classificação por cláusula usando QTYSOLD
- Uma função agregada SUM com um agrupamento por cláusula.

Os exemplos a seguir mostram o plano de consulta para cada comando CTAS.

```
explain create table sales1_simple as select listid, dateid, qtysold from sales;
                           QUERY PLAN
-----------------------------------------------------------------
 XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=8)
(1 row)
```

```
explain create table sales2_limit as select listid, dateid, qtysold from sales limit
 100;
                              QUERY PLAN
----------------------------------------------------------------------
 XN Limit  (cost=0.00..1.00 rows=100 width=8)
   ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=8)
(2 rows)


explain create table sales3_orderbylistid as select listid, dateid, qtysold from sales
 order by listid;
                               QUERY PLAN
-----------------------------------------------------------------------
 XN Sort  (cost=1000000016724.67..1000000017155.81 rows=172456 width=8)
   Sort Key: listid
   ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=8)
(3 rows)


explain create table sales4_orderbyqty as select listid, dateid, qtysold from sales
 order by qtysold;
                               QUERY PLAN
-----------------------------------------------------------------------
 XN Sort  (cost=1000000016724.67..1000000017155.81 rows=172456 width=8)
   Sort Key: qtysold
   ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=8)
(3 rows)


explain create table sales5_groupby as select listid, dateid, sum(qtysold) from sales
 group by listid, dateid;
                              QUERY PLAN
----------------------------------------------------------------------
 XN HashAggregate  (cost=3017.98..3226.75 rows=83509 width=8)
   ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=8)
(2 rows)
```

Para exibir a chave de distribuição e a chave de classificação para cada tabela, consulte a tabela de catálogo de sistema PG_TABLE_DEF, como mostrado a seguir.

```
select * from pg_table_def where tablename like 'sales%';
```

```
      tablename       |   column   | distkey | sortkey
----------------------+------------+---------+---------
 sales                | salesid    | f       |       0
 sales                | listid     | t       |       0
 sales                | sellerid   | f       |       0
 sales                | buyerid    | f       |       0
 sales                | eventid    | f       |       0
 sales                | dateid     | f       |       1
 sales                | qtysold    | f       |       0
 sales                | pricepaid  | f       |       0
 sales                | commission | f       |       0
 sales                | saletime   | f       |       0
 sales1_simple        | listid     | t       |       0
 sales1_simple        | dateid     | f       |       1
 sales1_simple        | qtysold    | f       |       0
 sales2_limit         | listid     | f       |       0
 sales2_limit         | dateid     | f       |       0
 sales2_limit         | qtysold    | f       |       0
 sales3_orderbylistid | listid     | t       |       1
 sales3_orderbylistid | dateid     | f       |       0
 sales3_orderbylistid | qtysold    | f       |       0
 sales4_orderbyqty    | listid     | t       |       0
 sales4_orderbyqty    | dateid     | f       |       0
 sales4_orderbyqty    | qtysold    | f       |       1
 sales5_groupby       | listid     | f       |       0
 sales5_groupby       | dateid     | f       |       0
 sales5_groupby       | sum        | f       |       0
```

A tabela a seguir resume os resultados. Para simplificar, esta tabela omite os detalhes de custo, linhas e largura do plano EXPLAIN.

Tabela	Instrução CTAS SELECT	Nó superior do plano de explicação	Chave de distribuição	Chave de classificação
S1_SIMPLE	`select listid, dateid, qtysold from sales`	`XN Seq Scan on sales ...`	LISTID	DATEID

Tabela	Instrução CTAS SELECT	Nó superior do plano de explicação	Chave de distribuição	Chave de classificação
S2_LIMIT	`select listid, dateid, qtysold from sales limit 100`	`XN Limit ...`	None (EVEN)	Nenhum
S3_ORDER_BY_LISTID	`select listid, dateid, qtysold from sales order by listid`	`XN Sort ...` `Sort Key: listid`	LISTID	LISTID
S4_ORDER_BY_QTY	`select listid, dateid, qtysold from sales order by qtysold`	`XN Sort ...` `Sort Key: qtysold`	LISTID	QTYSOLD
S5_GROUP_BY	`select listid, dateid, sum(qtysold) from sales group by listid, dateid`	`XN HashAggregate ...`	None (EVEN)	Nenhum

Você pode especificar explicitamente o estilo de distribuição e a chave de classificação da instrução CTAS. Por exemplo, a instrução a seguir cria uma tabela usando a distribuição EVEN e especifica SALESID como a chave de classificação.

```
create table sales_diseteven
diststyle even
sortkey (salesid)
as
select eventid, venueid, dateid, eventname
from event;
```

Codificação de compactação

ENCODE AUTO é usado como o padrão para tabelas. O Amazon Redshift gerencia automaticamente a codificação de compactação para todas as colunas da tabela.

Distribuição de dados de entrada

Quando o esquema de distribuição de hash de dados de entrada corresponder aos dados da tabela de destino, nenhuma distribuição física de dados é realmente necessária quando os dados são carregados. Por exemplo, se uma chave de distribuição for definida para a nova tabela e os dados estiverem sendo introduzidos a partir de outra tabela que é distribuída na mesma coluna chave, os dados serão carregados no local, usando os mesmos nós e fatias. No entanto, se as tabelas de origem e de destino forem definidas com a distribuição EVEN, os dados serão redistribuídos em uma tabela de destino.

Operações ANALYZE automáticas

O Amazon Redshift analisa automaticamente tabelas criadas com comandos CTAS. Não precisar executar o comando ANALYZE nessas tabelas quando elas são inicialmente criadas. Se você as modificar, você deve analisá-las da mesma forma que outras tabelas.

Exemplos de CTAS

O exemplo a seguir cria uma tabela denominada EVENT_BACKUP para a tabela EVENT:

```
create table event_backup as select * from event;
```

A tabela resultante herda as chaves de distribuição e de classificação da tabela EVENT.

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'event_backup';

column    | type                        | encoding | distkey | sortkey
----------+-----------------------------+----------+---------+--------
catid     | smallint                    | none     | false   |       0
dateid    | smallint                    | none     | false   |       1
eventid   | integer                     | none     | true    |       0
eventname | character varying(200)      | none     | false   |       0
starttime | timestamp without time zone | none     | false   |       0
venueid   | smallint                    | none     | false   |       0
```

O comando a seguir cria uma nova tabela denominada EVENTDISTSORT selecionando quatro colunas da tabela EVENT. A nova tabela é distribuída por EVENTID e classificada por EVENTID e DATEID:

```
create table eventdistsort
distkey (1)
sortkey (1,3)
as
select eventid, venueid, dateid, eventname
from event;
```

O resultado é o seguinte:

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'eventdistsort';

column   |          type          | encoding | distkey | sortkey
---------+------------------------+----------+---------+-------
eventid   | integer                | none     | t       | 1
venueid   | smallint               | none     | f       | 0
dateid    | smallint               | none     | f       | 2
eventname | character varying(200)| none     | f       | 0
```

Você pode criar exatamente a mesma tabela usando nomes de coluna para as chaves de distribuição e de classificação. Por exemplo:

```
create table eventdistsort1
distkey (eventid)
sortkey (eventid, dateid)
as
select eventid, venueid, dateid, eventname
from event;
```

A instrução a seguir aplica distribuição uniforme à tabela, mas não define uma chave de classificação explícita.

```
create table eventdisteven
diststyle even
as
select eventid, venueid, dateid, eventname
```

```
from event;
```

A tabela não herda a chave de classificação da tabela EVENT (EVENTID), pois a distribuição EVEN é especificada para a nova tabela. A nova tabela não tem chave de classificação e de distribuição.

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'eventdisteven';

column    |          type          | encoding | distkey | sortkey
----------+------------------------+----------+---------+---------
eventid   | integer                | none     | f       | 0
venueid   | smallint               | none     | f       | 0
dateid    | smallint               | none     | f       | 0
eventname | character varying(200) | none     | f       | 0
```

A instrução a seguir aplica a distribuição uniforme e define uma chave de classificação:

```
create table eventdistevensort diststyle even sortkey (venueid)
as select eventid, venueid, dateid, eventname from event;
```

A tabela resultante tem uma chave de classificação, mas não uma de distribuição.

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'eventdistevensort';

column    |          type          | encoding | distkey | sortkey
----------+------------------------+----------+---------+-------
eventid   | integer                | none     | f       | 0
venueid   | smallint               | none     | f       | 1
dateid    | smallint               | none     | f       | 0
eventname | character varying(200) | none     | f       | 0
```

A instrução a seguir redistribui a tabela EVENT em uma coluna chave diferente dos dados de entrada, que são classificados na coluna EVENTID, e não define nenhuma coluna SORTKEY. Portanto, a tabela não é classificada.

```
create table venuedistevent distkey(venueid)
as select * from event;
```

O resultado é o seguinte:

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'venuedistevent';

 column   |            type             | encoding | distkey | sortkey
----------+-----------------------------+----------+---------+-------
eventid   | integer                     | none     | f       | 0
venueid   | smallint                    | none     | t       | 0
catid     | smallint                    | none     | f       | 0
dateid    | smallint                    | none     | f       | 0
eventname | character varying(200)      | none     | f       | 0
starttime | timestamp without time zone | none     | f       | 0
```

CREATE TEMPLATE

Cria modelos reutilizáveis para comandos do Amazon Redshift, como COPY. Os modelos armazenam parâmetros comumente usados que podem ser referidos em várias execuções de comando, melhorando a consistência e reduzindo a especificação manual de parâmetros.

Os modelos eliminam a necessidade de especificar repetidamente os mesmos parâmetros de formatação em várias operações, enquanto os caminhos de origem, as tabelas de destino e a autorização podem variar entre as operações.

Privilégios obrigatórios

Para criar um modelo, é necessário ter uma das seguintes opções:

- Privilégios de superusuário.
- Permissão CREATE no esquema em que você deseja criar o modelo ou permissão CREATE com escopo nos esquemas no banco de dados em que você deseja criar o modelo

Sintaxe

```
CREATE [ OR REPLACE ] TEMPLATE [database_name.][schema_name.]template_name
FOR COPY [ AS ]
[ [ FORMAT ] [ AS ] data_format ]
[ parameter [ argument ] [ , ... ] ];
```

Parâmetros

OR REPLACE

Se um modelo com o mesmo nome já existir no banco de dados e no esquema especificados, o modelo existente será substituído. Só é possível substituir um modelo por um novo modelo que defina o mesmo tipo de operação; por exemplo, COPY. Você deve ter os privilégios necessários para substituir um modelo.

database_name

(Opcional) O nome do banco de dados onde o modelo será criado. Se não especificado, o modelo será criado no banco de dados atual.

Se o banco de dados ou o esquema não existir, o modelo não será criado e a instrução exibirá um erro. Não é possível criar modelos nos bancos de dados `template0`,`template1`, `padb_harvest` ou `sys:internal` do sistema.

schema_name

(Opcional) O nome do esquema onde o modelo será criado. Se não especificado, o modelo será criado no esquema atual.

Se um nome de esquema for fornecido, o novo modelo será criado no esquema em questão (presumindo que o criador tenha acesso ao esquema). O nome do modelo deve ser exclusivo para o esquema em questão.

template_name

O nome do modelo a ser criado. Como opção, é possível qualificar o nome do modelo com o nome do banco de dados e do esquema. No exemplo a seguir, o nome do banco de dados é `demo_database`, o nome do esquema é `demo_schema` e o nome do modelo é `test`. Para obter mais informações sobre nomes válidos, consulte Nomes e identificadores.

```
CREATE TEMPLATE demo_database.demo_schema.test FOR COPY AS CSV;
```

COPY

Especifica o tipo de comando do Redshift para o qual o modelo foi criado. No momento, é possível usar apenas o comando COPY.

[[FORMAT] [AS] data_format]

Esse parâmetro é opcional. Isso especifica o formato de dados para operações COPY.

[parameter [argument]]

Qualquer parâmetro válido para o comando do Redshift especificado.

Por exemplo, os modelos para o comando COPY podem incluir:

- Parâmetros de formato de dados
- Parâmetros de compactação de arquivo
- Parâmetros da conversão de dados
- Operações de carregamento de dados

Para ver uma lista completa de parâmetros compatíveis, consulte o comando COPY.

Observações de uso

- Por padrão, todos os usuários têm privilégios CREATE e USAGE no esquema PUBLIC. Para impedir que usuários criem objetos no esquema PUBLIC de um banco de dados, use o comando REVOKE para remover esse privilégio.
- Quando existe um parâmetro no modelo e no comando, o parâmetro do comando tem precedência.
- Os modelos são objetos de banco de dados e seguem as regras padrão de nomeação e permissão de objetos do Redshift. Para obter mais informações sobre nomes válidos, consulte Nomes e identificadores.
- Os modelos não podem conter especificações de arquivo de manifesto para o comando COPY.

Limitações

- Pelo menos um parâmetro deve ser especificado ao criar um modelo.
- Parâmetros excluídos: parâmetros específicos de comando, como caminhos de origem, tabelas de destino, credenciais de autorização e especificações de arquivos de manifesto, não podem ser incluídos nos modelos. Esses parâmetros devem ser especificados no comando real.
- Número máximo de modelos por cluster: é possível criar um máximo de 1.000 modelos por cluster. Esse limite se aplica ao número total de modelos em todos os bancos de dados e esquemas no cluster.
- Referências entre bancos de dados: não é possível referir modelos em bancos de dados.

- Compartilhamento de dados: não é possível incluir modelos em unidades de compartilhamentos de dados. Os modelos devem ser criados separadamente em cada cluster em que são necessários.

Exemplos

O exemplo a seguir cria um modelo para o comando COPY.

```
CREATE TEMPLATE test_schema.demo_template
FOR COPY
AS
FORMAT JSON 'auto'
NULL AS ''
MAXERROR 100;
```

Use SHOW TEMPLATE para obter a definição do modelo:

```
SHOW TEMPLATE test_schema.demo_template;
CREATE OR REPLACE TEMPLATE dev.test_schema.demo_template FOR COPY AS FORMAT AS JSON
 'auto' NULL '' MAXERROR 100;
```

Consulte a visualização SYS_REDSHIFT_TEMPLATE do sistema para obter mais detalhes sobre um modelo.

```
SELECT * FROM SYS_REDSHIFT_TEMPLATE;

database_name | schema_name | template_name | template_type |        create_time
  |     last_modified_time     | owner_id | last_modified_by | template_parameters
--------------+-------------+---------------+---------------
+---------------------------+---------------------------+----------
+------------------+--------------------
 dev           | test_schema | demo_template |             1 | 2025-12-17
 20:06:01.944171 | 2025-12-17 20:06:01.944171 |        1 |                1 | {
    "JSON": "auto",
    "MAXERROR": 100,
    "NULL": ""
}
```

CREATE USER

Cria um usuário de banco de dados. Os usuários do banco de dados podem recuperar dados, executar comandos e realizar outras ações em um banco de dados, dependendo de seus privilégios e perfis. Você deve ser um superusuário do banco de dados para executar este comando.

Privilégios obrigatórios

A seguir estão os privilégios obrigatórios para CREATE USER:

- Superusuário
- Usuários com o privilégio CREATE USER

Sintaxe

```
CREATE USER name [ WITH ]
PASSWORD { 'password' | 'md5hash' | 'sha256hash' | DISABLE }
[ option [ ... ] ]

where option can be:

CREATEDB | NOCREATEDB
| CREATEUSER | NOCREATEUSER
| SYSLOG ACCESS { RESTRICTED | UNRESTRICTED }
| IN GROUP groupname [, ... ]
| VALID UNTIL 'abstime'
| CONNECTION LIMIT { limit | UNLIMITED }
| SESSION TIMEOUT limit
| EXTERNALID external_id
```

Parâmetros

name

O nome do usuário a ser criado. O nome do usuário não pode ser `PUBLIC`. Para obter mais informações sobre nomes válidos, consulte Nomes e identificadores.

WITH

Palavra-chave opcional. WITH é ignorado pelo Amazon Redshift

PASSWORD { 'password' | 'md5hash' | 'sha256hash' | DISABLE }

Define a senha do usuário.

Por padrão, os usuários podem alterar suas próprias senhas, a menos que a senha esteja desabilitada. Para desabilitar a senha de um usuário, especifique DISABLE. Quando a senha de um usuário for desabilitada, ela será excluída do sistema e o usuário poderá fazer logon apenas usando as credenciais temporárias do usuário do AWS Identity and Access Management (IAM). Para obter mais informações, consulte Uso da autenticação do IAM para gerar credenciais do usuário do banco de dados. Apenas um superusuário pode habilitar ou desabilitar senhas. Não é possível desabilitar a senha de um superusuário. Para habilitar uma senha, execute ALTER USER e especifique uma senha.

É possível especificar a senha em texto simples, como uma string de hash MD5 ou como uma string de hash SHA256.

> **Note**
>
> Ao executar um novo cluster usando o Console de gerenciamento da AWS, a AWS CLI, ou a API do Amazon Redshift,, você deverá fornecer uma senha de texto não criptografado para o usuário inicial do banco de dados. Você pode alterar a senha posteriormente usando ALTER USER.

Para texto simples, a senha deve atender às seguintes restrições:

- Deve ter de 8 a 64 caracteres.
- Deve conter pelo menos uma letra maiúscula, uma letra minúscula e um número.
- Pode conter qualquer caractere ASCII com códigos ASCII 33–126, exceto aspas simples ('), aspas duplas ("), \, / ou @.

É possível especificar uma sequência de hash MD5 que inclua a senha e o nome do usuário como alternativa mais segura para o parâmetro CREATE USER em texto simples.

> **Note**
>
> Quando você especifica uma sequência de hash MD5, o comando CREATE USER procura uma sequência de hash MD5 válida, mas não valida a parte da sequência

contendo a senha. Nesse caso, é possível criar uma senha, como uma sequência vazia, que você não pode usar para acessar o banco de dados.

Para especificar uma senha MD5, siga estas etapas:

1. Concatene a senha e o nome do usuário.

   Por exemplo, para senha `ez` e usuário `user1`, a sequência concatenada é `ezuser1`.
2. Converta a sequência concatenada em uma sequência de hash MD5 de 32 caracteres. É possível usar qualquer utilitário de MD5 para criar a sequência de hash. O exemplo a seguir usa o Função MD5 do Amazon Redshift e o operador de concatenação (||) para retornar uma string de hash MD5 de 32 caracteres.

   ```
   select md5('ez' || 'user1');

   md5
   ----------------------------------
   153c434b4b77c89e6b94f12c5393af5b
   ```
3. Concatene 'md5' na frente da sequência de hash MD5 e forneça a sequência concatenada como argumento md5hash.

   ```
   create user user1 password 'md5153c434b4b77c89e6b94f12c5393af5b';
   ```
4. Faça login no banco de dados usando as credenciais de login.

   Para este exemplo, faça logon como `user1` com a senha `ez`.

Uma alternativa segura é especificar o hash SHA-256 de uma string de senha; ou você pode fornecer seu próprio digest SHA-256 válido e salt de 256 bits que foi usado para criar o digest.

- Digest: a saída de uma função de hash.
- Salt: dados gerados aleatoriamente combinados com a senha para ajudar a reduzir padrões na saída da função de hash.

```
'sha256|Mypassword'
```

```
'sha256|digest|256-bit-salt'
```

No exemplo a seguir, o Amazon Redshift gera e gerencia o salt.

```
CREATE USER admin PASSWORD 'sha256|Mypassword1';
```

No exemplo a seguir, são fornecidos um resumo SHA-256 válido e salt de 256 bits que foi usado para criar o resumo.

Para especificar uma senha e aplicar hash com seu próprio sal, siga estas etapas:

1. Crie um sal de 256 bits. É possível obter um sal usando qualquer gerador de string hexadecimal para gerar uma string de 64 caracteres. Neste exemplo, o sal é `c721bff5d9042cf541ff7b9d48fa8a6e545c19a763e3710151f9513038b0f6c6`.
2. Use a função FROM_HEX para converter o sal em binário. Isso ocorre porque a função SHA2 requer a representação binária do sal. Veja a instrução a seguir.

   ```
   SELECT
    FROM_HEX('c721bff5d9042cf541ff7b9d48fa8a6e545c19a763e3710151f9513038b0f6c6');
   ```
3. Use a função CONCAT para acrescentar o sal à sua senha. Neste exemplo, a senha é `Mypassword1`. Veja a instrução a seguir.

   ```
   SELECT
    CONCAT('Mypassword1',FROM_HEX('c721bff5d9042cf541ff7b9d48fa8a6e545c19a763e3710151f951303
   ```
4. Use a função SHA2 para criar um resumo da combinação de senha e sal. Veja a instrução a seguir.

   ```
   SELECT
    SHA2(CONCAT('Mypassword1',FROM_HEX('c721bff5d9042cf541ff7b9d48fa8a6e545c19a763e3710151f9
    0);
   ```
5. Usando o resumo e o sal das etapas anteriores, crie o usuário. Veja a instrução a seguir.

   ```
   CREATE USER admin PASSWORD 'sha256|
   821708135fcc42eb3afda85286dee0ed15c2c461d000291609f77eb113073ec2|
   c721bff5d9042cf541ff7b9d48fa8a6e545c19a763e3710151f9513038b0f6c6';
   ```
6. Faça login no banco de dados usando as credenciais de login.

   Para este exemplo, faça logon como `admin` com a senha `Mypassword1`.

Se você definir uma senha em texto sem especificar a função de hash, um digest MD5 será gerado usando o nome de usuário como salt.

CREATEDB | NOCREATEDB

A opção CREATEDB permite que o novo usuário crie bancos de dados. NOCREATEDB é o padrão.

CREATEUSER | NOCREATEUSER

A opção CREATEUSER cria um superusuário com todos os privilégios de banco de dados, incluindo CREATE USER. NOCREATEUSER é o valor padrão. Para obter mais informações, consulte superuser.

SYSLOG ACCESS { RESTRICTED | UNRESTRICTED }

Uma cláusula que especifica o nível de acesso do usuário para as tabelas e as visualizações de sistema do Amazon Redshift.

Usuários regulares que têm a permissão SYSLOG ACCESS RESTRICTED podem ver somente as linhas geradas por esse usuário nas tabelas e visualizações do sistema visíveis ao usuário. O padrão é RESTRICTED.

Usuários regulares que têm a permissão UNRESTRICTED podem ver todas as linhas nas tabelas e visualizações de sistema visíveis ao usuário, incluindo as linhas geradas por outro usuário. UNRESTRICTED não fornece acesso para os usuários regulares às tabelas visíveis para superusuários. Somente superusuários podem visualizar tabelas visíveis para superusuários.

> **Note**
>
> Fornecer acesso ilimitado para um usuário às tabelas de sistema é o mesmo que dar ao usuário visibilidade para os dados gerados por outros usuários. Por exemplo, STL_QUERY e STL_QUERYTEXT contêm texto completo de instruções INSERT, UPDATE e DELETE, e podem conter dados confidenciais gerados pelos usuários.

Todas as linhas em SVV_TRANSACTIONS são visíveis a todos os usuários.

Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

IN GROUP groupname

Especifica o nome de um grupo existente ao qual o usuário pertence. Vários nomes de grupos podem ser listados.

VALID UNTIL abstime

A opção VALID UNTIL define um tempo absoluto após o qual a senha do usuário não é mais válida. Por padrão, a senha não tem um limite de tempo.

CONNECTION LIMIT { limite | UNLIMITED }

Número máximo de conexões de banco de dados que o usuário pode abrir simultaneamente. Não há aplicação de limite para superusuários. Use a palavra-chave UNLIMITED para permitir o número máximo de conexões simultâneas. Um limite no número de conexões para cada banco de dados pode ser aplicável. Para obter mais informações, consulte CREATE DATABASE. O valor padrão é UNLIMITED. Para visualizar as conexões atuais, consulte a exibição STV_SESSIONS do sistema.

> **Note**
>
> Se limites de usuário e de conexão de banco de dados forem aplicáveis, um slot de conexão não utilizado que esteja dentro de ambos os limites deve estar disponível quando um usuário tenta se conectar.

SESSION TIMEOUT limit

O tempo máximo em segundos em que uma sessão permanece inativa ou ociosa. O intervalo é de 60 segundos (um minuto) a 1.728.000 segundos (20 dias). Se nenhum tempo limite de sessão estiver definido para o usuário, a configuração de cluster será aplicada. Para obter mais informações, consulte “Cotas e limites no Amazon Redshift” no Guia de gerenciamento de clusters do Amazon Redshift.

Quando você define o tempo limite da sessão, ele é aplicado somente a novas sessões.

Para exibir informações sobre sessões de usuário ativas, incluindo a hora de início, o nome de usuário e o tempo limite da sessão, consulte a visualização de sistema STV_SESSIONS. Para exibir informações sobre o histórico de sessões de usuário, consulte a visualização STL_SESSIONS. Para recuperar informações sobre usuários do banco de dados, incluindo valores de tempo limite de sessão, consulte a visualização SVL_USER_INFO.

EXTERNALID external_id

O identificador para o usuário, que está associado a um provedor de identidades. O usuário deve ter a senha desabilitada. Para obter mais informações, consulte Federação do provedor de identidades (IdP) nativo para o Amazon Redshift.

Observações de uso

Por padrão, todos os usuários têm privilégios CREATE e USAGE no esquema PUBLIC. Para impedir que usuários criem objetos no esquema PUBLIC de um banco de dados, use o comando REVOKE para remover esse privilégio.

Ao usar a autenticação do IAM para criar credenciais de usuário de banco de dados, convém criar um superusuário que possa fazer logon usando apenas credenciais temporárias. Você não pode desabilitar a senha de um superusuário, mas pode criar uma senha desconhecida usando uma string de hash MD5 gerada aleatoriamente.

```
create user iam_superuser password 'md5A1234567890123456780123456789012' createuser;
```

As maiúsculas e minnúsculas de um nome de usuário entre aspas duplas são sempre mantidas, independentemente da opção de configuração `enable_case_sensitive_identifier`. Para obter mais informações, consulte enable_case_sensitive_identifier.

Exemplos

O comando a seguir cria um usuário chamado dbuser, com a senha “abcD1234”, privilégios de criação de banco de dados e um limite de 30 conexões.

```
create user dbuser with password 'abcD1234' createdb connection limit 30;
```

Consulte a tabela de catálogo PG_USER_INFO para exibir detalhes sobre o usuário do bancos de dados.

```
select * from pg_user_info;

 usename   | usesysid | usecreatedb | usesuper | usecatupd | passwd   | valuntil |
 useconfig | useconnlimit
-----------+----------+-------------+----------+-----------+----------+----------
+-----------+-------------
```

```
 rdsdb     |        1 | true        | true     | true      | ******** | infinity |
      |
 adminuser |      100 | true        | true     | false     | ******** |          |
      | UNLIMITED
 dbuser    |      102 | true        | false    | false     | ******** |          |
      | 30
```

No exemplo, a senha da conta é válida até 10 de junho de 2017.

```
create user dbuser with password 'abcD1234' valid until '2017-06-10';
```

O exemplo a seguir cria um usuário com uma senha que diferencia maiúsculas de minúsculas e que contém caracteres especiais.

```
create user newman with password '@AbC4321!';
```

Para usar uma barra invertida (“\") na sua senha MD5, ignore a barra invertida com uma barra invertida na sua string de origem. O exemplo a seguir cria um usuário denominado `slashpass` com uma única barra invertida ("\") como senha.

```
select md5('\\'||'slashpass');

md5
--------------------------------
0c983d1a624280812631c5389e60d48c
```

Crie um usuário com a senha md5.

```
create user slashpass password 'md50c983d1a624280812631c5389e60d48c';
```

O exemplo a seguir cria um usuário chamado `dbuser` com um tempo limite de sessão ociosa definido para 120 segundos.

```
CREATE USER dbuser password 'abcD1234' SESSION TIMEOUT 120;
```

O exemplo a seguir cria um usuário denominado bob. Esse namespace é `myco_aad`. Isso é apenas um exemplo. Para executar o comando com êxito, você deve ter um provedor de identidades registrado. Para obter mais informações, consulte Federação do provedor de identidades (IdP) nativo para o Amazon Redshift.

```
CREATE USER myco_aad:bob EXTERNALID "ABC123" PASSWORD DISABLE;
```

CREATE VIEW

Cria uma exibição em um banco de dados. A exibição não é materializada fisicamente. A consulta que define a exibição é executada sempre que ela é mencionada em uma consulta. Para criar uma exibição com uma tabela externa, inclua a cláusula WITH NO SCHEMA BINDING.

Para criar uma visualização padrão, você precisa acessar as tabelas subjacentes ou as visualizações subjacentes. Para consultar uma exibição padrão, você precisa selecionar permissões para a própria exibição, mas não precisa selecionar permissões para as tabelas subjacentes. Quando criar uma exibição que referencia uma tabela ou uma exibição em outro esquema ou se criar uma exibição que referencia uma visão materializada, você precisará de permissões de uso. Para consultar uma exibição de vinculação tardia, você precisa selecionar permissões para ela. Você também precisa verificar se o proprietário da visualização de vinculação tardia selecionou as permissões para os objetos referenciados (tabelas, visualizações ou funções definidas pelo usuário). Para obter mais informações sobre exibições de vinculação tardia, consulte Observações de uso.

Permissões obrigatórias

Para usar CREATE VIEW, uma das permissões a seguir é necessária.

- Para criar uma visualização usando CREATE [OR REPLACE] VIEW:
  - Superusuário
  - Usuários com a permissão CREATE [REPLACE] VIEW
- Para substituir uma visualização existente usando CREATE OR REPLACE VIEW:
  - Superusuário
  - Usuários com a permissão CREATE [OR REPLACE] VIEW
  - Proprietário da exibição

Se um usuário quiser acessar uma visualização que incorpora uma função definida pelo usuário, o usuário deverá ter a permissão EXECUTE nessa função.

Sintaxe

```
CREATE [ OR REPLACE ] VIEW name [ ( column_name [, ...] ) ] AS query
```

```
[ WITH NO SCHEMA BINDING ]
```

Parâmetros

OR REPLACE

Se uma exibição com o mesmo nome já existir, a exibição será substituída. Você somente pode substituir uma exibição com uma consulta nova que gere o conjunto idêntico de colunas, usando os mesmos nomes de coluna e tipos de dados. CREATE OR REPLACE VIEW bloqueia a exibição para leituras e gravações até que a operação seja concluída.

Quando uma exibição é substituída, suas outras propriedades, como propriedade e privilégios concedidos, são preservadas.

name

Nome da exibição. Se um nome de esquema (como `myschema.myview`) for fornecido, a exibição será criada usando o esquema especificado. Caso contrário, a exibição será criada no esquema atual. O nome da exibição deve ser diferente do nome de qualquer outra exibição ou tabela no mesmo esquema.

Se você especificar um nome para a exibição que comece com "#", a exibição é criada como temporária e será visível somente na sessão atual.

Para obter mais informações sobre nomes válidos, consulte Nomes e identificadores. Você não pode criar tabelas ou exibições nos bancos de dados do sistema template0, template1, padb_harvest ou sys:internal.

column_name

Lista opcional de nomes a serem usados para as colunas na exibição. Se nenhum nome de coluna for fornecido, os nomes de colunas serão derivados da consulta. O número máximo de colunas que você pode definir em uma única exibição é 1.600.

query

Consulta (na forma de instrução SELECT) que avalia uma tabela. Esta tabela define as colunas e linhas na exibição.

WITH NO SCHEMA BINDING

Cláusula que especifica que a exibição não está vinculada a objetos de banco de dados subjacentes, como tabelas e funções definidas pelo usuário. Como resultado, não há nenhuma

dependência entre a exibição e os objetos aos quais ela faz referência. Você pode criar uma exibição mesmo se não houver objetos referenciados. Como não há dependência, você pode descartar ou alterar um objeto referenciado sem afetar a visualização. O Amazon Redshift não verifica as dependências até que a exibição seja consultada. Expressões de tabela comuns recursivas (rCTE) não são compatíveis com visualizações de vinculação tardia. Para consultar detalhes sobre exibições de vinculação tardia, execute a função PG_GET_LATE_BINDING_VIEW_COLS.

Quando você incluir a cláusula WITH NO SCHEMA BINDING, as tabelas e as exibições referenciadas na instrução SELECT deverão ser qualificadas com um nome de esquema. Deve haver um esquema quando a exibição for criada, mesmo se a tabela referenciada não existir. Por exemplo, a seguinte instrução retorna um erro.

```
create view myevent as select eventname from event
with no schema binding;
```

A instrução a seguir é executada com êxito.

```
create view myevent as select eventname from public.event
with no schema binding;
```

> **Note**
>
> Não é possível atualizar, inserir ou excluir em uma visualização.

Observações de uso

Visualizações de vinculação tardia

Uma exibição de vinculação tardia só verifica os objetos de banco de dados subjacentes, como tabelas e outras exibições, depois que a exibição for consultada. Consequentemente, você pode alterar ou remover os objetos subjacentes sem remover e recriar a exibição. Se você eliminar objetos subjacentes, as consultas para a exibição de vinculação tardia apresentarão falha. Se a consulta à exibição de vinculação tardia fizer referência às colunas no objeto subjacente que não estão presentes, ela apresentará falha.

Se você descartar e recriar a tabela ou exibição subjacente de uma exibição de vinculação tardia, o novo objeto será criado com permissões de acesso padrão. Você precisará conceder permissões aos objetos subjacentes para usuários que farão consultas na visualização.

Para criar uma exibição de vinculação tardia, inclua a cláusula WITH NO SCHEMA BINDING. O exemplo a seguir cria uma exibição sem vinculação de esquema.

```
create view event_vw as select * from public.event
with no schema binding;
```

```
select * from event_vw limit 1;

eventid | venueid | catid | dateid | eventname     | starttime
--------+---------+-------+--------+---------------+--------------------
      2 |     306 |     8 |   2114 | Boris Godunov | 2008-10-15 20:00:00
```

O exemplo a seguir mostra que você pode alterar uma tabela subjacente sem recriar a exibição.

```
alter table event rename column eventname to title;
```

```
select * from event_vw limit 1;

eventid | venueid | catid | dateid | title         | starttime
--------+---------+-------+--------+---------------+--------------------
      2 |     306 |     8 |   2114 | Boris Godunov | 2008-10-15 20:00:00
```

Você pode fazer referência às tabelas externas do Amazon Redshift Spectrum somente em uma visualização de vinculação tardia. Uma aplicação de visualizações de vinculação tardia consulta as tabelas do Amazon Redshift e do Redshift Spectrum. Por exemplo, você pode usar o comando UNLOAD para arquivar os dados mais antigos no Amazon S3. Em seguida, crie uma tabela externa do Redshift Spectrum que faça referência aos dados no Amazon S3 e crie uma exibição que consulte as duas tabelas. O exemplo a seguir usa uma cláusula UNION ALL para unir a tabela `SALES` do Amazon Redshift e a tabela `SPECTRUM.SALES` do Redshift Spectrum.

```
create view sales_vw as
select * from public.sales
union all
select * from spectrum.sales
```

```
with no schema binding;
```

Para obter mais informações sobre como criar tabelas externas do Redshift Spectrum, incluindo a tabela `SPECTRUM.SALES`, consulte Conceitos básicos do Amazon Redshift Spectrum.

> **Important**
>
> Ao criar uma visualização padrão com base em uma visualização de vinculação tardia, a definição da visualização padrão inclui a definição da visualização de vinculação tardia no momento em que a visualização padrão foi criada, incluindo o proprietário da visualização de vinculação tardia. Se você fizer uma alteração na visualização de vinculação tardia subjacente, ela não será usada na visualização padrão enquanto você não recriar a visualização padrão. Portanto, quando a visualização padrão for consultada, ela sempre usará a definição e o proprietário da visualização de vinculação tardia para verificação de permissão no momento da criação dessa visualização padrão.

Para atualizar a exibição padrão a fim de referenciar a definição mais recente da exibição de vinculação tardia, execute CREATE OR REPLACE VIEW com a definição de exibição inicial usada por você para criar a exibição padrão.

Consulte o exemplo a seguir da criação de uma exibição padrão a partir de uma exibição de vinculação tardia.

```
create view sales_vw_lbv as
select * from public.sales
with no schema binding;

show view sales_vw_lbv;
                           Show View DDL statement
------------------------------------------------------------------------------
 create view sales_vw_lbv as select * from public.sales with no schema binding;
(1 row)

create view sales_vw as
select * from sales_vw_lbv;

show view sales_vw;
                                                 Show View DDL statement
-------------------------------------------------------------------------------------------------
```

```
 SELECT sales_vw_lbv.price, sales_vw_lbv."region" FROM (SELECT sales.price,
 sales."region" FROM sales) sales_vw_lbv;
(1 row)
```

A exibição de vinculação tardia, conforme mostrada na instrução DDL da exibição padrão, é definida quando a exibição padrão é criada e não será atualizada com nenhuma alteração feita posteriormente por você na exibição de vinculação tardia.

Exemplos

Os comandos de exemplo usam um conjunto de objetos e dados de amostra chamado banco de dados TICKIT. Para obter mais informações, consulte Banco de dados de amostra.

O comando a seguir cria uma exibição denominada myevent a partir de uma tabela denominada EVENT.

```
create view myevent as select eventname from event
where eventname = 'LeAnn Rimes';
```

O comando a seguir cria uma exibição denominada myuser a partir de uma tabela denominada USERS.

```
create view myuser as select lastname from users;
```

O comando a seguir cria ou substitui uma exibição denominada myuser a partir de uma tabela denominada USERS.

```
create or replace view myuser as select lastname from users;
```

O exemplo a seguir cria uma exibição sem vinculação de esquema.

```
create view myevent as select eventname from public.event
with no schema binding;
```

DEALLOCATE

Libera uma instrução preparada.

Sintaxe

```
DEALLOCATE [PREPARE] plan_name
```

Parâmetros

PREPARE

Esta palavra-chave é opcional e ignorada.

plan_name

Nome da instrução preparada para a liberação.

Observações sobre o uso

DEALLOCATE é usada para liberar uma instrução SQL preparada anteriormente. Se você não liberar explicitamente uma instrução preparada, ela será liberada quando a sessão atual for finalizada. Para obter mais informações sobre instruções preparadas, consulte PREPARE.

Consulte também

EXECUTE, PREPARE

DECLARE

Define um novo cursor. Use um cursor para recuperar algumas linhas de cada vez do conjunto de resultados de uma consulta maior.

Quando a primeira linha do cursor é buscada, o conjunto inteiro de resultados é materializado no nó líder, na memória ou em disco, se necessário. Devido ao impacto da performance negativa potencial de usar cursores com grandes conjuntos de resultado, recomendamos abordagens alternativas sempre que possível. Para obter mais informações, consulte Considerações sobre a performance ao usar cursores.

Você deve declarar um cursor em um bloco de transação. Apenas um cursor pode ser aberto por sessão.

Para obter mais informações, consulte FETCHCLOSE.

Sintaxe

```
DECLARE cursor_name CURSOR FOR query
```

Parâmetros

nome_cursor

Nome do novo cursor.

query

Uma instrução SELECT que preenche o cursor.

Observações de uso de DECLARE CURSOR

Se seu aplicativo cliente usa uma conexão ODBC e sua consulta cria um conjunto de resultados que é muito grande para caber na memória, você pode transmitir o conjunto de resultados para seu aplicativo cliente usando um cursor. Quando você usa um cursor, o conjunto inteiro de resultados é materializado no nó líder, e a partir de então seu cliente pode buscar os resultados adicionais.

> **Note**
>
> Para permitir cursores em ODBC para Microsoft Windows, habilite a opção Use Declare/Fetch no DSN ODBC que você usa para o Amazon Redshift. Recomendamos configurar o tamanho do cache de ODBC usando o campo Cache Size nas opções da caixa de diálogo do DSN de ODBC, para 4.000 ou superior em clusters de nós múltiplos para minimizar round trips. Em um cluster de nó único, defina o tamanho do cache para 1.000.

Devido ao impacto da performance negativa potencial do uso de cursores, recomendamos abordagens alternativas sempre que possível. Para obter mais informações, consulte Considerações sobre a performance ao usar cursores.

Os cursores do Amazon Redshift são compatíveis com as seguintes restrições:

- Apenas um cursor pode ser aberto por sessão.
- Os cursores devem ser usados em uma transação (BEGIN … END).

- O tamanho máximo do conjunto de resultados cumulativo para todos os cursores é restringido com base no tipo de nó do cluster. Esse é um total em todo o cluster compartilhado entre todos os cursores abertos simultaneamente. Um único cursor pode usar apenas parte dele, porque parte da capacidade é reservada para que vários cursores possam ser executados simultaneamente. Se precisar de conjuntos de resultados maiores, você poderá redimensionar para uma configuração de nós maior.

  Para obter mais informações, consulte Restrições de cursor.

Restrições de cursor

Quando a primeira linha de um cursor é buscada, o conjunto inteiro de resultados é materializado no nó líder. Se o resultado não couber na memória, ele será gravado em disco conforme necessário. Para proteger a integridade do nó líder, o Amazon Redshift impõe limites ao tamanho dos conjuntos de resultados dos cursores, com base no tipo de nó do cluster.

O valor na tabela a seguir é um total em todo o cluster, e não um limite por cursor. Para garantir que vários cursores possam permanecer abertos simultaneamente, parte dessa capacidade total é reservada para uso simultâneo. Como resultado, um único cursor não pode usar sozinho todo o máximo cumulativo; o maior conjunto de resultados que um único cursor pode conter é menor que o total mostrado na tabela. Um cursor individual pode falhar quando seu conjunto de resultados atingir esse limite por cursor, mesmo quando a capacidade total de cursores do cluster não estiver sendo totalmente utilizada.

A tabela a seguir mostra o tamanho máximo cumulativo do conjunto de resultados para cada tipo de nó do cluster. O tamanho máximo do conjunto de resultados é fornecido em megabytes.

Tipo de nó	Tamanho máximo cumulativo do conjunto de resultados por cluster (MB)
Vários nós rg.12xlarge	7.200.000
Vários nós rg.4xlarge	1.600.000
Vários nós rg.xlarge	240.000
Vários nós rg.large	100.000
rg.large, nó único	8.000

Tipo de nó	Tamanho máximo cumulativo do conjunto de resultados por cluster (MB)
Vários nós ra3.16xlarge	14.400.000
Vários nós ra3.4xlarge	3.200.000
Vários nós ra3.xlplus	1.000.000
Nó único ra3.xlplus	64.000
Vários nós ra3.large	240.000
Nó único ra3.large	8.000
Amazon Redshift Sem Servidor	150.000
Vários nós dc2.8xlarge	3.200.000
Vários nós dc2.large	192.000
Nó único dc2.large	8.000

Para visualizar a configuração de cursor ativa para um cluster, consulte a tabela de sistema STV_CURSOR_CONFIGURATION como superusuário. Para exibir o estado de cursores ativos, consulte a tabela de sistema STV_ACTIVE_CURSORS. Somente as linhas dos cursores do próprio usuário são visíveis para ele, mas um superusuário pode visualizar todos os cursores.

Considerações sobre a performance ao usar cursores

Como os cursores materializam o conjunto de resultados inteiro no nó líder antes de começar a retornar resultados para o cliente, usar cursores com conjuntos muito grandes de resultados pode ter um impacto negativo na performance. É altamente recomendável não usar cursores com conjuntos muito grandes de resultados. Em alguns casos, como quando seu aplicativo usa uma conexão ODBC, os cursores podem ser a única solução viável. Se possível, recomendamos o uso destas alternativas:

- Use UNLOAD para exportar uma tabela grande. Quando você usa UNLOAD, os nós de computação funcionam em paralelo para transferir os dados diretamente para arquivos de dados

no Amazon Simple Storage Service. Para obter mais informações, consulte Descarregar dados no Amazon Redshift.

- Defina o parâmetro de tamanho de busca JDBC no seu aplicativo cliente. Se você utiliza uma conexão JDBC e estiver encontrando erros de falta de memória por parte do cliente, é possível habilitar seu cliente para recuperar conjuntos de resultados em lotes menores configurando o parâmetro de tamanho de busca JDBC. Para obter mais informações, consulte Como configurar o parâmetro JDBC para o tamanho da busca.

Os nós RG têm tamanhos máximos de conjunto de resultados de cursor menores do que os nós RA3 com capacidade de computação semelhante. Se as workloads que utilizam cursores se aproximarem ou excederem o limite do seu tipo de nó RG, considere as seguintes opções:

- Otimize o tamanho dos conjuntos de resultados. Aplique filtros de consulta mais seletivos, reduza o número de colunas recuperadas ou implemente paginação no nível do aplicativo para processar os resultados em lotes menores.
- Migre para um tipo de nó RG maior. Um tipo de nó maior oferece maior capacidade para cursores. Se a quantidade de nós permitir e o desempenho da workload não for afetado, você poderá reduzir proporcionalmente o número de nós para manter um custo total do cluster comparável.

Para determinar o tamanho atual do conjunto de resultados do cursor, consulte a visualização SYS_QUERY_HISTORY para obter o valor `returned_bytes` associado à primeira instrução FETCH executada no cursor. Consulte mais informações em SYS_QUERY_HISTORY.

Exemplo de DECLARE CURSOR

O exemplo a seguir declara um cursor denominado LOLLAPALOOZA para selecionar informações de vendas para o evento Lollapalooza e busca linhas do conjunto de resultados usando o cursor:

```
-- Begin a transaction

begin;

-- Declare a cursor

declare lollapalooza cursor for
select eventname, starttime, pricepaid/qtysold as costperticket, qtysold
from sales, event
where sales.eventid = event.eventid
```

```
and eventname='Lollapalooza';

-- Fetch the first 5 rows in the cursor lollapalooza:

fetch forward 5 from lollapalooza;

  eventname   |      starttime      | costperticket | qtysold
--------------+---------------------+---------------+---------
 Lollapalooza | 2008-05-01 19:00:00 |   92.00000000 |       3
 Lollapalooza | 2008-11-15 15:00:00 |  222.00000000 |       2
 Lollapalooza | 2008-04-17 15:00:00 |  239.00000000 |       3
 Lollapalooza | 2008-04-17 15:00:00 |  239.00000000 |       4
 Lollapalooza | 2008-04-17 15:00:00 |  239.00000000 |       1
(5 rows)

-- Fetch the next row:

fetch next from lollapalooza;

  eventname   |      starttime      | costperticket | qtysold
--------------+---------------------+---------------+---------
 Lollapalooza | 2008-10-06 14:00:00 |  114.00000000 |       2

-- Close the cursor and end the transaction:

close lollapalooza;
commit;
```

O exemplo a seguir percorre um refcursor com todos os resultados de uma tabela:

```
CREATE TABLE tbl_1 (a int, b int);
INSERT INTO tbl_1 values (1, 2),(3, 4);

CREATE OR REPLACE PROCEDURE sp_cursor_loop() AS $$
DECLARE
    target record;
    curs1 cursor for select * from tbl_1;
BEGIN
    OPEN curs1;
    LOOP
        fetch curs1 into target;
        exit when not found;
        RAISE INFO 'a %', target.a;
```

```
    END LOOP;
    CLOSE curs1;
END;
$$ LANGUAGE plpgsql;

CALL sp_cursor_loop();

SELECT message
   from svl_stored_proc_messages
   where querytxt like 'CALL sp_cursor_loop()%';

  message
----------
      a 1
      a 3
```

DELETE

Exclui linhas de tabelas.

> **Note**
>
> O tamanho máximo de uma única instrução SQL é 16 MB.

Sintaxe

```
[ WITH [RECURSIVE] common_table_expression [, common_table_expression , ...] ]
DELETE [ FROM ] { table_name | materialized_view_name }
    [ { USING } table_name, ... ]
    [ WHERE condition ]
```

Parâmetros

Cláusula WITH

Cláusula opcional que especifica uma ou mais expressões de tabela-comum. Consulte Cláusula WITH.

FROM

A palavra-chave FROM é opcional, exceto quando a cláusula USING for especificada. As instruções `delete from event;` e `delete event;` são operações equivalentes que eliminam todas as linhas da tabela EVENT.

> **Note**
>
> Para excluir todas as linhas de uma tabela, use TRUNCATE na tabela. TRUNCATE é muito mais eficiente do que DELETE e não requer VACUUM nem ANALYZE. No entanto, esteja ciente de que TRUNCATE confirma a transação em que é executado.

table_name

Uma tabela temporária ou persistente. Somente o proprietário da tabela ou um usuário com o privilégio DELETE na tabela pode excluir linhas da tabela.

Considere usar o comando TRUNCATE para operações de exclusão não qualificadas rápidas em grandes tabelas. Consulte TRUNCATE.

> **Note**
>
> Depois de excluir um grande número de linhas de uma tabela:
>
> - Limpe a tabela para recuperar espaço e reclassificar as linhas.
> - Analise a tabela para atualizar as estatísticas do planejador de consulta.

materialized_view_name

Uma visão materializada. O comando DELETE funciona em uma visão materializada usada para Ingestão de streaming para uma visão materializada. Somente o proprietário da visão materializada ou um usuário com privilégio DELETE na visão materializada pode excluir linhas dela.

Você não pode executar DELETE em uma visão materializada para ingestão de streaming com uma política de segurança por linha (RLS) que não tenha a permissão IGNORE RLS concedida ao usuário. Há uma exceção: se o usuário que está executando DELETE tiver IGNORE RLS concedido, ele será executado com êxito. Para obter mais informações, consulte Propriedade e gerenciamento da política de RLS.

USING nome_tabela, ...

A palavra-chave USING é usada para inserir uma lista da tabela quando tabelas adicionais são mencionadas na condição de cláusula WHERE. Por exemplo, o comando a seguir exclui todas as linhas da tabela EVENT que atendem a condição de junção nas tabelas EVENT e SALES. A tabela SALES deve ser explicitamente denominada na lista FROM:

```
delete from event using sales where event.eventid=sales.eventid;
```

Se você repetir o nome da tabela de destino na cláusula USING, a operação DELETE executa uma junção automática. Você pode usar um subconsulta na cláusula WHERE em vez da sintaxe USING como alternativa para gravar a mesma consulta.

WHERE condição

Cláusula opcional que limita a exclusão de linhas àquelas que correspondem à condição. Por exemplo, a condição pode ser uma restrição em uma coluna, uma condição de junção ou uma condição baseada no resultado de uma consulta. A consulta pode fazer referência a tabelas diferentes do destino do comando DELETE. Por exemplo:

```
delete from t1
where col1 in(select col2 from t2);
```

Se nenhuma condição for especificada, todas as linhas na tabela serão excluídas.

Observações de uso

- As operações DELETE mantêm bloqueios exclusivos quando executadas em visões materializadas de streaming do Amazon Redshift conectadas a qualquer um dos seguintes:
  - Um Amazon Kinesis data stream
  - Um tópico Amazon Managed Streaming for Apache Kafka
  - Um stream externo compatível, como um tópico do Confluent Cloud Kafka

  Para obter mais informações, consulte Ingestão de streaming para uma visão materializada.

Exemplos

Excluir todas as linhas da tabela CATEGORY:

```
delete from category;
```

Excluir linhas com valores CATID entre 0 e 9 da tabela CATEGORY:

```
delete from category
where catid between 0 and 9;
```

Excluir linhas da tabela LISTING cujos valores SELLERID não existem na tabela SALES:

```
delete from listing
where listing.sellerid not in(select sales.sellerid from sales);
```

As duas consultas a seguir excluem uma linha da tabela CATEGORY, com base em uma junção à tabela EVENTO e a uma restrição adicional na coluna CATID:

```
delete from category
using event
where event.catid=category.catid and category.catid=9;
```

```
delete from category
where catid in
(select category.catid from category, event
where category.catid=event.catid and category.catid=9);
```

A consulta a seguir exclui todas as linhas da visão materializada `mv_cities`. Neste exemplo, o nome da visão materializada é uma amostra:

```
delete from mv_cities;
```

DESC DATASHARE

Exibe uma lista dos objetos de banco de dados dentro de um datashare que são adicionados a ele usando ALTER DATASHARE. O Amazon Redshift exibe os nomes, bancos de dados, esquemas e tipos de tabelas, visualizações e funções.

Informações adicionais sobre objetos da unidade de compartilhamento de dados podem ser encontradas usando visualizações do sistema. Para obter mais informações, consulte SVV_DATASHARE_OBJECTS e SVV_DATASHARES.

Sintaxe

```
DESC DATASHARE datashare_name [ OF [ ACCOUNT account_id ] NAMESPACE namespace_guid ]
```

Parâmetros

datashare_name

O nome do datashare.

NAMESPACE namespace_guid

Um valor que especifica o namespace que o datashare usa. Quando você executa o DESC DATAHSARE como um administrador de cluster de consumidor, especifique o parâmetro NAMESPACE para exibir os datashares de entrada.

ACCOUNT account_id

Valor que especifica a conta a qual o datashare pertence.

Observações sobre o uso

Como administrador de conta de consumidor, quando você executa o DESC DATASHARE para ver os datashares de entrada da AWS, especifique a opção NAMESPACE. Quando você executa o DESC DATASHARE para ver os datashares de entrada em contas da AWS, especifique as opções ACCOUNT e NAMESPACE.

Exemplos

O exemplo a seguir exibe as informações para datashares de saída em um cluster de produtores.

```
DESC DATASHARE salesshare;

producer_account |          producer_namespace           | share_type  | share_name   |
 object_type |        object_name           |  include_new
-----------------+--------------------------------------+-------------+--------------
+-------------+------------------------------+-------------
 123456789012    | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d  | OUTBOUND    |  salesshare  |
 TABLE       | public.tickit_sales_redshift |
 123456789012    | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d  | OUTBOUND    |  salesshare  |
 SCHEMA      | public                       |   t
```

O exemplo a seguir exibe as informações para datashares de entrada em um cluster de produtores.

```
DESC DATASHARE salesshare of ACCOUNT '123456789012' NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';

 producer_account |          producer_namespace          | share_type | share_name |
 object_type |         object_name          |  include_new
------------------+--------------------------------------+------------+------------
+-------------+------------------------------+-------------
 123456789012     | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare |
 table       | public.tickit_sales_redshift |
 123456789012     | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare |
 schema      | public                       |
(2 rows)
```

DESC IDENTITY PROVIDER

Exibe informações sobre um provedor de identidades. Somente um superusuário pode descrever um provedor de identidades.

Sintaxe

```
DESC IDENTITY PROVIDER identity_provider_name
```

Parâmetros

identity_provider_name

O nome do provedor de identidades.

Exemplo

O exemplo a seguir exibe informações sobre o provedor de identidades.

```
DESC IDENTITY PROVIDER azure_idp;
```

Exemplo de resultado.

```
  uid   |   name    | type  |             instanceid             | namespc |

                                        params
```

```
                           | enabled
--------+----------+-------+--------------------------------------+---------
+-------------------------------------------------------------------------------------------------
+---------
 126692 | azure_idp | azure | e40d4bb2-7670-44ae-bfb8-5db013221d73 | aad     |
 {"issuer":"https://login.microsoftonline.com/e40d4bb2-7670-44ae-bfb8-5db013221d73/
v2.0", "client_id":"871c010f-5e61-4fb1-83ac-98610a7e9110", "client_secret":'',
 "audience":["https://analysis.windows.net/powerbi/connector/AmazonRedshift", "https://
analysis.windows.net/powerbi/connector/AWSRDS"]} | t
(1 row)
```

DETACH MASKING POLICY

Desanexa uma política de mascaramento dinâmico de dados que já está anexada a uma coluna. Para obter mais informações sobre mascaramento dinâmico de dados, consulte Mascaramento dinâmico de dados.

Superusuários e usuários ou perfis que têm o perfil sys:secadmin podem desanexar uma política de mascaramento.

Sintaxe

```
DETACH MASKING POLICY
{
  policy_name ON table_name
  | database_name.policy_name ON database_name.schema_name.table_name
}
( output_column_names )
FROM { user_name | ROLE role_name | PUBLIC };
```

Parâmetros

policy_name

O nome da política de mascaramento a ser desanexada.

database_name

O nome do banco de dados no qual a política e a relação são criadas. A política e a relação precisam estar no mesmo banco de dados. O banco de dados pode ser o conectado ou um banco de dados que comporte as permissões federadas do Amazon Redshift.

schema_name

O nome do esquema ao qual pertence a relação.

table_name

O nome da tabela da qual deseja desanexar a política de mascaramento.

output_column_names

Os nomes das colunas às quais a política de mascaramento foi anexada.

user_name

O nome do usuário ao qual a política de mascaramento foi anexada.

Só é possível definir uma opção entre user_name, role_name e PUBLIC em uma instrução DETACH MASKING POLICY.

role_name

O nome do perfil ao qual a política de mascaramento foi anexada.

Só é possível definir uma opção entre user_name, role_name e PUBLIC em uma instrução DETACH MASKING POLICY.

PUBLIC

Mostra que a política foi anexada a todos os usuários na tabela.

Só é possível definir uma opção entre user_name, role_name e PUBLIC em uma instrução DETACH MASKING POLICY.

Para o uso da DETACH MASKING POLICY no Catálogo de Permissões Federadas do Amazon Redshift, consulte Gerenciar o controle de acesso com permissões federadas do Amazon Redshift.

DETACH RLS POLICY

Desanexe uma política de segurança no nível da linha em uma tabela de um ou mais usuários ou funções.

Superusuários e usuários ou funções que têm a função `sys:secadmin` podem desanexar uma política.

Sintaxe

```
DETACH RLS POLICY
{
  policy_name ON [TABLE] table_name [, ...]
  | database_name.policy_name ON [TABLE] database_name.schema_name.table_name [, ...]
}
FROM { user_name | ROLE role_name | PUBLIC } [, ...];
```

Parâmetros

policy_name

O nome da política.

database_name

O nome do banco de dados no qual a política e a relação são criadas. A política e a relação precisam estar no mesmo banco de dados. O banco de dados pode ser o conectado ou um banco de dados que comporte as permissões federadas do Amazon Redshift.

schema_name

O nome do esquema ao qual pertence a relação.

table_name

A relação à qual a política de segurança no nível da linha está anexada.

FROM { user_name | ROLE role_name | PUBLIC} [, ...]

Especifica se a política está desanexada de um ou mais usuários ou funções especificados.

Para o uso da DETACH RLS POLICY no Catálogo de Permissões Federadas do Amazon Redshift, consulte Gerenciar o controle de acesso com permissões federadas do Amazon Redshift.

Observações de uso

Ao trabalhar com a instrução DETACH RLS POLICY, observe o seguinte:

- É possível desanexar uma política de uma relação, usuário, função ou público.

Exemplos

O exemplo a seguir desvincula uma tabela de uma função.

```
DETACH RLS POLICY policy_concerts ON tickit_category_redshift FROM ROLE analyst, ROLE
 dbadmin;
```

DROP DATABASE

Remove um banco de dados.

Não é possível executar DROP DATABASE em um bloco de transação (BEGIN ... END). Para obter mais informações sobre transações, consulte Níveis de isolamento no Amazon Redshift.

Sintaxe

```
DROP DATABASE database_name [ FORCE ]
```

Parâmetros

database_name

Nome do banco de dados a ser removido. Você não pode descartar os bancos de dados dev, padb_harvest, template0, template1, ou sys:internal, além de não pode descartar o banco de dados atual.

Para descartar um banco de dados externo, descarte o esquema externo. Para obter mais informações, consulte DROP SCHEMA.

FORCE

Quando você especifica FORCE, DROP DATABASE tenta encerrar as conexões ativas antes de eliminar o banco de dados. Se todas as conexões ativas forem encerradas com êxito dentro de um tempo limite, a eliminação prosseguirá. Se nem todas as conexões forem encerradas, o comando lançará um erro.

Observações sobre o uso de DROP DATABASE

Ao usar a instrução DROP DATABASE, considere o seguinte:

- Em geral, recomendamos não descartar um banco de dados que contenha uma unidade de compartilhamento de dados AWS Data Exchange usando a instrução DROP DATASHARE. Caso altere, as Contas da AWS com acesso à unidade de compartilhamento de dados perdem o acesso. Executar esse tipo de alteração pode violar os termos do produto de dados no AWS Data Exchange.

  O exemplo a seguir mostra um erro quando um banco de dados que contém uma unidade de compartilhamento de dados do AWS Data Exchange é descartada.

```
DROP DATABASE test_db;
ERROR:   Drop of database test_db that contains ADX-managed datashare(s)
 requires session variable datashare_break_glass_session_var to be set to value
 'ce8d280c10ad41'
```

  Para permitir o descarte do banco de dados, defina a seguinte variável e execute a instrução DROP DATABASE novamente.

```
SET datashare_break_glass_session_var to 'ce8d280c10ad41';
```

```
DROP DATABASE test_db;
```

  Nesse caso, o Amazon Redshift gera um valor único aleatório para definir a variável de sessão para permitir DROP DATABASE para um banco de dados que contenha uma unidade de compartilhamento de dados do AWS Data Exchange.

Exemplos

O exemplo a seguir permite remover um banco de dados com o nome TICKIT_TEST:

```
drop database tickit_test;
```

DROP DATASHARE

Descarta um datashare. Esse comando é irreversível.

Somente um superusuário ou o proprietário do datashare pode descartar um datashare.

Privilégios obrigatórios

A seguir estão os privilégios obrigatórios para DROP DATASHARE:

- Superusuário
- Usuários com o privilégio DROP DATASHARE
- Proprietário da unidade de compartilhamento de dados

Sintaxe

```
DROP DATASHARE datashare_name;
```

Parâmetros

datashare_name

Nome do banco de dados a ser descartado.

Observações sobre o uso de DROP DATASHARE

Ao usar a instrução DROP DATASHARE, considere o seguinte:

- Em geral, recomendamos não descartar uma unidade de compartilhamento de dados AWS Data Exchange usando a instrução DROP DATASHARE. Caso altere, as Contas da AWS com acesso à unidade de compartilhamento de dados perdem o acesso. Executar esse tipo de alteração pode violar os termos do produto de dados no AWS Data Exchange.

  O exemplo a seguir mostra um erro quando uma unidade de compartilhamento de dados do AWS Data Exchange é descartada.

  ```
  DROP DATASHARE salesshare;
  ERROR:  Drop of ADX-managed datashare salesshare requires session variable
   datashare_break_glass_session_var to be set to value '620c871f890c49'
  ```

  Para permitir o descarte de uma unidade de compartilhamento de dados do AWS Data Exchange, defina a seguinte variável e execute a instrução DROP DATASHARE novamente.

  ```
  SET datashare_break_glass_session_var to '620c871f890c49';
  ```

```
DROP DATASHARE salesshare;
```

Nesse caso, o Amazon Redshift gera um valor único aleatório para definir a variável de sessão para permitir DROP DATASHARE para uma unidade de compartilhamento de dados do AWS Data Exchange.

Exemplos

O exemplo a seguir descarta uma unidade de compartilhamento de dados chamada `salesshare`.

```
DROP DATASHARE salesshare;
```

DROP EXTERNAL VIEW

Elimina uma exibição externa do banco de dados. A eliminação de uma exibição externa a remove de todos os mecanismos SQL aos quais a exibição está associada, como Amazon Athena e Amazon EMR Spark. O comando não pode ser revertido. Para obter mais informações sobre visualizações do Catálogo de Dados, consulte Visualizações do AWS Glue Data Catalog.

Sintaxe

```
DROP EXTERNAL VIEW schema_name.view_name [ IF EXISTS ]
{catalog_name.schema_name.view_name | awsdatacatalog.dbname.view_name |
 external_schema_name.view_name}
```

Parâmetros

schema_name.view_name

O esquema anexado ao banco de dados do AWS Glue, seguido do nome da exibição.

IF EXISTS

Só eliminará a exibição se ela existir.

catalog_name.schema_name.view_name | awsdatacatalog.dbname.view_name | external_schema_name.view_name

A notação do esquema a ser usado durante a eliminação da exibição. Você pode especificar o uso do AWS Glue Data Catalog, um banco de dados do Glue criado por você, ou um esquema

externo também criado por você. Consulte CREATE DATABASE e CREATE EXTERNAL SCHEMA para obter mais informações.

query_definition

A definição da consulta SQL executada pelo Amazon Redshift para alterar a exibição.

Exemplos

O exemplo a seguir elimina uma exibição do Data Catalog chamada sample_schema.glue_data_catalog_view.

```
DROP EXTERNAL VIEW sample_schema.glue_data_catalog_view IF EXISTS
```

DROP FUNCTION

Remove uma função definida pelo usuário (UDF) do banco de dados. A assinatura da função, ou a lista de tipos de dados de argumento, deve ser especificada pelas funções múltiplas que podem existir com o mesmo nome mas assinaturas diferentes. Você não pode descartar uma função integrada do Amazon Redshift.

Esse comando é irreversível.

Privilégios obrigatórios

A seguir estão os privilégios obrigatórios para DROP FUNCTION:

- Superusuário
- Usuários com o privilégio DROP FUNCTION
- Proprietário da função

Sintaxe

```
DROP FUNCTION name
( [arg_name] arg_type   [, ...] )
[ CASCADE | RESTRICT ]
```

Parâmetros

name

Nome da função a ser removida.

nome_arg

O nome de um argumento de entrada. DROP FUNCTION ignora nomes de argumento, pois somente os tipos de dados de argumento são necessários para determinar a identidade da função.

tipo_arg

Tipo de dados do argumento de entrada. Você pode fornecer uma lista separada por vírgulas com um máximo de 32 tipos de dados.

CASCADE

Palavra-chave que especifica a remoção automática de objetos que dependem da função, como exibições.

Para criar uma exibição que não dependa de uma função, inclua a cláusula WITH NO SCHEMA BINDING na definição de exibição. Para obter mais informações, consulte CREATE VIEW.

RESTRICT

Palavra-chave que especifica que se houver objetos dependentes da função, a função não deve ser descartada e deve retornar uma mensagem. Esta ação é o padrão.

Exemplos

O exemplo a seguir remove a função denominada `f_sqrt`:

```
drop function f_sqrt(int);
```

Para remover uma função com dependências, use a opção CASCADE, conforme exibido no exemplo a seguir:

```
drop function f_sqrt(int)cascade;
```

DROP GROUP

Exclui um grupo de usuários. Esse comando é irreversível. Esse comando não exclui os usuários individuais em um grupo.

Consulte DROP USER para excluir um usuário individual.

Sintaxe

```
DROP GROUP name
```

Parâmetro

name

Nome do grupo de usuários a ser excluído.

Exemplo

O seguinte exemplo exclui o grupo de usuários `guests`:

```
DROP GROUP guests;
```

Você não pode descartar um grupo se o grupo tiver privilégios sobre um objeto. Se você tentar remover esse grupo, receberá o erro a seguir.

```
ERROR: group "guests" can't be dropped because the group has a privilege on some object
```

Se o grupo tiver privilégios para um objeto, revogue-os antes de remover o grupo. Para encontrar os objetos para os quais o grupo `guests` tem privilégios, use o exemplo a seguir. Para ter mais informações sobre a visualização de metadados utilizada no exemplo, consulte SVV_RELATION_PRIVILEGES.

```
SELECT DISTINCT namespace_name, relation_name, identity_name, identity_type
FROM svv_relation_privileges
WHERE identity_type='group' AND identity_name='guests';

+----------------+---------------+---------------+---------------+
| namespace_name | relation_name | identity_name | identity_type |
```

```
+----------------+---------------+---------------+---------------+
| public         | table1        | guests        | group         |
+----------------+---------------+---------------+---------------+
| public         | table2        | guests        | group         |
+----------------+---------------+---------------+---------------+
```

O exemplo a seguir revoga todos os privilégios em todas as tabelas no esquema `public` do grupo de usuários `guests` e, em seguida, remove o grupo.

```
REVOKE ALL ON ALL TABLES IN SCHEMA public FROM GROUP guests;
DROP GROUP guests;
```

DROP IDENTITY PROVIDER

Exclui um provedor de identidades. Esse comando é irreversível. Somente um superusuário pode remover um provedor de identidades.

Sintaxe

```
DROP IDENTITY PROVIDER identity_provider_name [ CASCADE ]
```

Parâmetros

identity_provider_name

O nome do provedor de identidades a excluir.

CASCADE

Exclui usuários e funções anexados ao provedor de identidades quando ele é excluído.

Exemplo

O exemplo a seguir exclui o provedor de identidades oauth_provider.

```
DROP IDENTITY PROVIDER oauth_provider;
```

Se você remover o provedor de identidades, alguns usuários talvez não consigam fazer login ou usar ferramentas de cliente configuradas para utilizá-lo.

DROP LIBRARY

Remove uma biblioteca personalizada Python do banco de dados. Somente o proprietário da biblioteca ou um superusuário pode remover uma biblioteca.

DROP LIBRARY não pode ser executado em um bloco de transação (BEGIN … END). Para obter mais informações sobre transações, consulte Níveis de isolamento no Amazon Redshift.

Esse comando é irreversível. O comando DROP LIBRARY é confirmado imediatamente. Se uma UDF que depende da biblioteca estiver em execução simultaneamente, a UDF pode falhar, mesmo se a UDF estiver sendo executada em uma transação.

Para obter mais informações, consulte CREATE LIBRARY.

Privilégios obrigatórios

A seguir estão os privilégios obrigatórios para DROP LIBRARY:

- Superusuário
- Usuários com o privilégio DROP LIBRARY
- Proprietário da biblioteca

Sintaxe

```
DROP LIBRARY library_name
```

Parâmetros

nome_biblioteca

O nome da biblioteca.

DROP MASKING POLICY

Descarta uma política de mascaramento dinâmico de dados de todos os bancos de dados. Não é possível descartar uma política de mascaramento que ainda esteja anexada a uma ou mais tabelas. Para obter mais informações sobre mascaramento dinâmico de dados, consulte Mascaramento dinâmico de dados.

Superusuários e usuários ou perfis que têm o perfil sys:secadmin podem descartar uma política de mascaramento.

Sintaxe

```
DROP MASKING POLICY { policy_name | database_name.policy_name };
```

Parâmetros

policy_name

O nome da política de mascaramento que deseja descartar.

database_name

O nome do banco de dados do qual a política será descartada. O banco de dados pode ser o conectado ou um banco de dados que comporte as permissões federadas do Amazon Redshift.

Para o uso da DROP MASKING POLICY no Catálogo de Permissões Federadas do Amazon Redshift, consulte Gerenciar o controle de acesso com permissões federadas do Amazon Redshift.

DROP MODEL

Remove um modelo do banco de dados. Somente o proprietário do modelo ou um superusuário pode descartar um modelo.

O DROP MODEL também exclui toda a função de previsão associada derivada desse modelo, todos os artefatos do Amazon Redshift relacionados ao modelo e todos os dados do Amazon S3 relacionados ao modelo. Enquanto o modelo ainda estiver sendo treinado no Amazon SageMaker AI, DROP MODEL cancelará essas operações.

Esse comando é irreversível. O comando DROP MODEL é confirmado imediatamente.

Permissões obrigatórias

A seguir estão as permissões obrigatórias para DROP MODEL:

- Superusuário
- Usuários com a permissão DROP MODEL
- Proprietário do modelo

- Proprietário do esquema

Sintaxe

```
DROP MODEL [ IF EXISTS ] model_name
```

Parâmetros

SE EXISTIR

Cláusula que indica que, se o esquema especificado existe, o comando não deve fazer alterações e retorna uma mensagem informando que o esquema existe.

model_name

O nome do modelo. O nome do modelo em um esquema deve ser exclusivo.

Exemplos

O exemplo a seguir descarta o modelo demo_ml.customer_churn.

```
DROP MODEL demo_ml.customer_churn
```

DROP MATERIALIZED VIEW

Remove uma visualização materializada.

Para obter mais informações sobre visões materializadas, consulte Visões materializadas no Amazon Redshift.

Sintaxe

```
DROP MATERIALIZED VIEW [ IF EXISTS ] mv_name [, ... ] [ CASCADE | RESTRICT ]
```

Parâmetros

IF EXISTS

Uma cláusula que especifica para verificar se a visão materializada nomeada existe. Se a visão materializada não existir, o comando `DROP MATERIALIZED VIEW` retornará uma mensagem

de erro. Essa cláusula é útil ao criar scripts, para evitar que o script falhe se você descartar uma visão materializada não existente.

mv_name

O nome da visão materializada a ser descartada.

CASCADE

Cláusula que indica que se deve remover automaticamente os objetos dos quais a visão materializada depende, como outras visualizações.

RESTRICT

Cláusula que indica que não se deve remover a visão materializada se qualquer objeto depender dela. Esse é o padrão.

Observações sobre o uso

Somente o proprietário de uma visão materializada pode usar `DROP MATERIALIZED VIEW` naquela exibição. Um superusuário ou um usuário que recebeu especificamente privilégios DROP podem ser exceções a isso.

Quando você escreve uma instrução de descarte para uma visão materializada e existe uma visualização com um nome correspondente, isso resulta em um erro que instrui você a usar DROP VIEW. Um erro ocorre mesmo no caso em que você usa `DROP MATERIALIZED VIEW IF EXISTS`.

Eliminar uma visualização materializada do Iceberg é uma operação somente de metadados. O Amazon Redshift remove a entrada da tabela do AWS Glue Data Catalog, mas não exclui os arquivos de dados subjacentes nem os arquivos de metadados do Iceberg no Amazon S3. Você é responsável por remover arquivos órfãos após descartar uma visualização materializada do Iceberg. Para remover dados órfãos e arquivos de metadados, use Excluir arquivos órfãos em AWS Glue ou manutenção de tabelas para Buckets de Tabelas do Amazon S3.

A opção CASCADE não é aceita para visualizações materializadas do Iceberg.

Exemplo

O exemplo a seguir descarta a visão materializada `tickets_mv`.

```
DROP MATERIALIZED VIEW tickets_mv;
```

DROP PROCEDURE

Descarta um procedimento. Para descartar um procedimento, são necessários o nome do procedimento e os tipos de dados do argumento de entrada (assinatura). Opcionalmente, você pode incluir todos os tipos de dados de argumentos, incluindo argumentos OUT. Para encontrar a assinatura de um procedimento, use o comando SHOW PROCEDURE. Para obter mais informações sobre assinaturas de procedimento, consulte PG_PROC_INFO.

Privilégios obrigatórios

A seguir estão os privilégios obrigatórios para DROP PROCEDURE:

- Superusuário
- Usuários com o privilégio DROP PROCEDURE
- Proprietário do procedimento

Sintaxe

```
DROP PROCEDURE sp_name ( [ [ argname ] [ argmode ] argtype [, ...] ] )
```

Parâmetros

sp_name

O nome do procedimento a ser removido.

argname

O nome de um argumento de entrada. DROP PROCEDURE ignora nomes de argumento, pois somente os tipos de dados de argumento são necessários para determinar a identidade do procedimento.

argmode

O modo de um argumento, que pode ser IN, OUT ou INOUT. Argumentos OUT são opcionais pois eles não são usados para identificar um procedimento armazenado.

argtype

Tipo de dados do argumento de entrada. Para obter uma lista dos tipos de dados compatíveis, consulte Tipos de dados.

Exemplos

O exemplo a seguir descarta um procedimento armazenado chamado `quarterly_revenue`.

```
DROP PROCEDURE quarterly_revenue(volume INOUT bigint, at_price IN numeric,result OUT
 int);
```

DROP RLS POLICY

Descarta uma política de segurança no nível da linha para todas as tabelas em todos os bancos de dados.

Superusuários e usuários ou funções que têm a função sys:secadmin podem descartar uma política.

Sintaxe

```
DROP RLS POLICY [ IF EXISTS ]
{ policy_name | database_name.policy_name }
[ CASCADE | RESTRICT ]
```

Parâmetros

SE EXISTIR

Uma cláusula que indica se a política especificada já existe.

policy_name

O nome da política.

database_name

O nome do banco de dados do qual a política será descartada. O banco de dados pode ser o conectado ou um banco de dados que comporte as permissões federadas do Amazon Redshift.

CASCADE

Uma cláusula que indica para desanexar automaticamente a política de todas as tabelas anexadas antes de descartar a política.

RESTRICT

Uma cláusula que indica que a política não se deve ser descartada quando ela é anexada a algumas tabelas. Esse é o padrão.

Para o uso da DROP RLS POLICY no Catálogo de Permissões Federadas do Amazon Redshift, consulte Gerenciar o controle de acesso com permissões federadas do Amazon Redshift.

Exemplos

O exemplo a seguir descarta a política de segurança no nível da linha.

```
DROP RLS POLICY policy_concerts;
```

DROP ROLE

Remove uma função de um banco de dados. Somente o proprietário da função que criou a função, um usuário com a opção WITH ADMIN ou um superusuário pode descartar uma função.

Você não pode descartar uma função concedida a um usuário ou outra função que dependa dessa função.

Privilégios obrigatórios

A seguir estão os privilégios obrigatórios para DROP ROLE:

- Superusuário
- Proprietário da função que é o usuário que criou a função ou um usuário que recebeu a função com o privilégio WITH ADMIN OPTION.

Sintaxe

```
DROP ROLE role_name [ FORCE | RESTRICT ]
```

Parâmetros

role_name

O nome da função.

[FORCE | RESTRICT]

A configuração padrão é RESTRICT. O Amazon Redshift emite um erro quando você tenta eliminar uma função que tenha herdado outra função. Use FORCE para remover todas as atribuições de função, se houver alguma.

Exemplos

O exemplo a seguir usa a função `sample_role`.

```
DROP ROLE sample_role FORCE;
```

O exemplo a seguir tenta descartar a função sample_role1 que foi concedida a um usuário com a opção RESTRICT padrão.

```
CREATE ROLE sample_role1;
GRANT ROLE sample_role1 TO user1;
DROP ROLE sample_role1;
ERROR:  cannot drop this role since it has been granted on a user
```

Para descartar com êxito a função sample_role1 que foi concedida a um usuário, use a opção FORCE.

```
DROP ROLE sample_role1 FORCE;
```

O exemplo a seguir tenta descartar a função sample_role2 que tem uma outra função dependente dela com a opção RESTRICT padrão.

```
CREATE ROLE sample_role1;
CREATE ROLE sample_role2;
GRANT ROLE sample_role1 TO sample_role2;
DROP ROLE sample_role2;
ERROR:  cannot drop this role since it depends on another role
```

Para descartar com êxito a função sample_role2 que tem outra função dependente dela, use a opção FORCE.

```
DROP ROLE sample_role2 FORCE;
```

DROP SCHEMA

Exclui um esquema. No caso do esquema externo, você também poderá descartar o banco de dados externo associado ao esquema. Esse comando é irreversível.

Privilégios obrigatórios

A seguir estão os privilégios obrigatórios para DROP SCHEMA:

- Superusuário
- Proprietário do esquema
- Usuários com o privilégio DROP SCHEMA

Sintaxe

```
DROP SCHEMA [ IF EXISTS ] name [, ...]
[ DROP EXTERNAL DATABASE ]
[ CASCADE | RESTRICT ]
```

Parâmetros

IF EXISTS

Cláusula que indica que, se o esquema especificado não existe, o comando não deve fazer alterações e retorna uma mensagem informando que o esquema não existe, em vez de encerrar com um erro.

Esta cláusula é útil durante scripting para que o script não falhe se o comando DROP SCHEMA for executado em um esquema não existente.

name

Nomes dos esquemas a serem descartados. É possível especificar vários nomes de esquemas separados por vírgulas.

DROP EXTERNAL DATABASE

Cláusula indicando que, se um esquema externo for removido, descarte o banco de dados externo associado ao esquema externo, se existir algum. Se não houver banco de dados externo, o comando retornará uma mensagem informando que não existe nenhum banco de dados externo. Se vários esquemas externos forem removidos, todos os bancos de dados associados aos esquemas especificados serão descartados.

Se um banco de dados externo contiver objetos dependentes, como tabelas, inclua a opção CASCADE para descartar também os objetos dependentes.

Quando você solta um banco de dados externo, o banco de dados também é descartado para qualquer outro esquema externo associado ao banco de dados. Tabelas definidas em outros esquemas externos usando o banco de dados também são descartadas.

O DROP EXTERNAL DATABASE não é compatível com bancos de dados externos armazenados em um metastore do HIVE.

CASCADE

Palavra-chave indicando que todos os objetos no esquema devem ser removidos automaticamente. Se DROP EXTERNAL DATABASE for especificado, todos os objetos no banco de dados externo também serão removidos.

RESTRICT

Palavra-chave indicando que um esquema ou banco de dados externo não deve ser removido se contiver algum objeto. Esta ação é o padrão.

Exemplo

O exemplo a seguir exclui um esquema denominado S_SALES. Este exemplo usa RESTRICT como mecanismo de segurança para que o esquema não seja excluído se contiver quaisquer objetos. Nesse caso, você precisa excluir objetos do esquema antes de excluir o esquema.

```
drop schema s_sales restrict;
```

O exemplo a seguir exclui um esquema chamado S_SALES e todos os objetos que dependem dele.

```
drop schema s_sales cascade;
```

O exemplo a seguir remove o esquema S_SALES se ele existir, ou não faz nada e retorna uma mensagem se o esquema não existir.

```
drop schema if exists s_sales;
```

O exemplo a seguir exclui um esquema externo denominado S_SPECTRUM e o banco de dados externo associado a ele. Esse exemplo usa RESTRICT para que o esquema e o banco de dados não sejam excluídos se contiverem objetos. Nesse caso, é preciso excluir os objetos dependentes antes de excluir o esquema e o banco de dados.

```
drop schema s_spectrum drop external database restrict;
```

O exemplo a seguir exclui vários esquemas e os bancos de dados externos associados a eles, juntamente com todos os objetos dependentes.

```
drop schema s_sales, s_profit, s_revenue drop external database cascade;
```

DESCARTAR TABELA

Remove uma tabela de um banco de dados.

Se você está tentando eliminar as linhas de uma tabela sem remover a tabela, use o comando DELETE ou TRUNCATE.

DROP TABLE elimina restrições que existem na tabela de destino. Várias tabelas podem ser removidas com um único comando DROP TABLE.

DROP TABLE com uma tabela externa não pode ser executado em uma transação (BEGIN … END). Para obter mais informações sobre transações, consulte Níveis de isolamento no Amazon Redshift.

Para encontrar um exemplo em que o privilégio DROP é concedido a um grupo, consulte GRANT Exemplos.

Privilégios obrigatórios

A seguir estão os privilégios obrigatórios para DROP TABLE:

- Superusuário
- Usuários com o privilégio DROP TABLE
- Proprietário da tabela com o privilégio USAGE no esquema

Sintaxe

```
DROP TABLE [ IF EXISTS ] name [, ...] [ CASCADE | RESTRICT ]
```

Parâmetros

IF EXISTS

Cláusula que indica que, se a tabela especificada não existe, o comando não deve fazer alterações e deve retornar uma mensagem informando que a tabela não existe, em vez de encerrar com um erro.

Esta cláusula é útil durante scripting para que o script não falhe se o comando DROP TABLE for executado em uma tabela não existente.

name

Nome da tabela a ser removida.

CASCADE

Cláusula que indica que para eliminar automaticamente os objetos que dependem da tabela, como exibições.

Para criar uma exibição que não dependa de outros objetos de banco de dados, como exibições e tabelas, inclua a cláusula WITH NO SCHEMA BINDING na definição de exibição. Para obter mais informações, consulte CREATE VIEW.

RESTRICT

Cláusula que indica que a tabela não se deve ser removida se contiver objetos que dependam dela. Esta ação é o padrão.

Exemplos

Remover uma tabela sem dependências

O exemplo a seguir cria e remove uma tabela chamada FEEDBACK que não tem dependências:

```
create table feedback(a int);

drop table feedback;
```

Se uma tabela contém as colunas que estão referidas por exibições ou por outras tabelas, o Amazon Redshift exibe uma mensagem como a seguir.

```
Invalid operation: cannot drop table feedback because other objects depend on it
```

Remover duas tabelas simultaneamente

O conjunto de comandos a seguir criar uma tabela FEEDBACK e uma tabela BUYERS. Em seguida, remove ambas as tabelas com um único comando:

```
create table feedback(a int);

create table buyers(a int);

drop table feedback, buyers;
```

Remover uma tabela com uma dependência

As etapas a seguir mostram como remover uma tabela chamada FEEDBACK usando a opção CASCADE.

Primeiro, crie uma tabela simples chamada FEEDBACK usando o comando CREATE TABLE:

```
create table feedback(a int);
```

Em seguida, use o comando CREATE VIEW para criar uma exibição chamada FEEDBACK_VIEW dependente da tabela FEEDBACK:

```
create view feedback_view as select * from feedback;
```

O exemplo a seguir remove a tabela FEEDBACK e também a exibição FEEDBACK_VIEW, pois FEEDBACK_VIEW depende da tabela FEEDBACK:

```
drop table feedback cascade;
```

Visualizar as dependências de uma tabela

Para retornar as dependências da tabela, use o exemplo a seguir. Substitua *my_schema* e *my_table* pelo esquema e tabela próprios.

```
SELECT dependent_ns.nspname as dependent_schema
, dependent_view.relname as dependent_view
, source_ns.nspname as source_schema
, source_table.relname as source_table
, pg_attribute.attname as column_name
```

```
FROM pg_depend
JOIN pg_rewrite ON pg_depend.objid = pg_rewrite.oid
JOIN pg_class as dependent_view ON pg_rewrite.ev_class = dependent_view.oid
JOIN pg_class as source_table ON pg_depend.refobjid = source_table.oid
JOIN pg_attribute ON pg_depend.refobjid = pg_attribute.attrelid
    AND pg_depend.refobjsubid = pg_attribute.attnum
JOIN pg_namespace dependent_ns ON dependent_ns.oid = dependent_view.relnamespace
JOIN pg_namespace source_ns ON source_ns.oid = source_table.relnamespace
WHERE
source_ns.nspname = 'my_schema'
AND source_table.relname = 'my_table'
AND pg_attribute.attnum > 0
ORDER BY 1,2
LIMIT 10;
```

Para eliminar *my_table* e as dependências, use o exemplo a seguir. O exemplo também retorna todas as dependências da tabela que foi descartada.

```
DROP TABLE my_table CASCADE;

SELECT dependent_ns.nspname as dependent_schema
, dependent_view.relname as dependent_view
, source_ns.nspname as source_schema
, source_table.relname as source_table
, pg_attribute.attname as column_name
FROM pg_depend
JOIN pg_rewrite ON pg_depend.objid = pg_rewrite.oid
JOIN pg_class as dependent_view ON pg_rewrite.ev_class = dependent_view.oid
JOIN pg_class as source_table ON pg_depend.refobjid = source_table.oid
JOIN pg_attribute ON pg_depend.refobjid = pg_attribute.attrelid
    AND pg_depend.refobjsubid = pg_attribute.attnum
JOIN pg_namespace dependent_ns ON dependent_ns.oid = dependent_view.relnamespace
JOIN pg_namespace source_ns ON source_ns.oid = source_table.relnamespace
WHERE
source_ns.nspname = 'my_schema'
AND source_table.relname = 'my_table'
AND pg_attribute.attnum > 0
ORDER BY 1,2
LIMIT 10;

+-----------------+----------------+--------------+--------------+-------------+
| dependent_schema | dependent_view | source_schema | source_table | column_name |
+-----------------+----------------+--------------+--------------+-------------+
```

Remover uma tabela usando IF EXISTS

O exemplo a seguir remove a tabela FEEDBACK se ela existir, ou não faz nada e retorna uma mensagem se o esquema não existir:

```
drop table if exists feedback;
```

DROP

Elimina um modelo de um banco de dados.

Privilégios obrigatórios

Para eliminar um modelo, é necessário ter uma das seguintes opções:

- Privilégios de superusuário.
- Privilégio DROP TEMPLATE e privilégio USAGE no esquema que contém o modelo.

Sintaxe

```
DROP TEMPLATE [database_name.][schema_name.]template_name;
```

Parâmetros

database_name

(Opcional) O nome do banco de dados no qual o modelo é criado. Se não especificado, será usado o banco de dados atual.

schema_name

(Opcional) O nome do esquema no qual o modelo é criado. Se não for especificado, o modelo será pesquisado no caminho de pesquisa atual.

template_name

O nome do modelo a ser removido. No exemplo a seguir, o nome do banco de dados é `demo_database`, o nome do esquema é `demo_schema` e o nome do modelo é `test`.

```
DROP TEMPLATE demo_database.demo_schema.test;
```

Exemplos

O seguinte exemplo remove o modelo test_template do esquema atual:

```
DROP TEMPLATE test_template;
```

O seguinte exemplo remove o modelo test_template do esquema test_schema:

```
DROP TEMPLATE test_schema.test_template;
```

DROP USER

Remove um usuário de um banco de dados. Vários usuários podem ser removidos com um único comando DROP USER. É necessário ser um superusuário do banco de dados ou ter a permissão DROP USER para executar esse comando.

Sintaxe

```
DROP USER [ IF EXISTS ] name [, ... ]
```

Parâmetros

IF EXISTS

Cláusula que indica que, se o usuário especificado não existe, o comando não deve fazer alterações e deve retornar uma mensagem informando que o usuário não existe, em vez de encerrar com um erro.

Essa cláusula é útil durante o scripting para que o script não falhe se o comando DROP USER for executado em um usuário não existente.

name

Nome do usuário a ser removido. É possível especificar vários usuários, com uma vírgula separando cada nome de usuário do texto.

Observações de uso

Não é possível descartar o usuário chamado `rdsdb` ou o usuário administrador do banco de dados, que normalmente é chamado de `awsuser` ou `admin`.

Você não pode remover um usuário se o usuário tem um objeto de banco de dados, como um esquema, um banco de dados, uma tabela ou uma exibição, ou se o usuário tem algum privilégio em um banco de dados, uma tabela ou grupo. Se você tentar remover esse usuário, receberá um dos erros abaixo.

```
ERROR: user "username" can't be dropped because the user owns some object [SQL
 State=55006]

ERROR: user "username" can't be dropped because the user has a privilege on some object
 [SQL State=55006]
```

Para obter instruções detalhadas sobre como encontrar os objetos pertencentes a um usuário do banco de dados, consulte Como resolvo o erro “usuário não pode ser descartado” no Amazon Redshift? no Centro de Conhecimentos.

> **Note**
>
> O Amazon Redshift verifica somente o banco de dados atual antes de descartar um usuário. DROP USER não retorna um erro se o usuário tiver objetos de banco de dados ou algum privilégio de objetos em outro banco de dados. Se você remover um usuário que tem objetos em outro banco de dados, o proprietário desses objetos será alterado para “unknown”.

Se um usuário tiver um objeto, primeiro remova o objeto ou altere sua propriedade para outro usuário antes de remover o usuário original. Se o usuário tiver privilégios para um objeto, revogue os privilégios antes de remover o usuário. O exemplo a seguir mostra como remover um objeto, alterar a propriedade e revogar privilégios antes de remover o usuário.

```
drop database dwdatabase;
alter schema dw owner to dwadmin;
revoke all on table dwtable from dwuser;
drop user dwuser;
```

Exemplos

O exemplo a seguir remove um usuário chamado paulo:

```
drop user paulo;
```

O exemplo a seguir exclui dois usuários, paulo e martha:

```
drop user paulo, martha;
```

O exemplo a seguir exclui o usuário paulo caso ele exista, ou não faz nada e retorna uma mensagem caso a conta não exista:

```
drop user if exists paulo;
```

DROP VIEW

Remove uma exibição do banco de dados. Várias exibições podem ser removidas com um único comando DROP VIEW. Esse comando é irreversível.

Privilégios obrigatórios

A seguir estão os privilégios obrigatórios para DROP VIEW:

- Superusuário
- Usuários com o privilégio DROP VIEW
- Proprietário da exibição

Sintaxe

```
DROP VIEW [ IF EXISTS ] name [, ... ] [ CASCADE | RESTRICT ]
```

Parâmetros

IF EXISTS

Cláusula que indica que, se a exibição especificada não existe, o comando não deve fazer alterações e deve retornar uma mensagem informando que a exibição não existe, em vez de encerrar com um erro.

Esta cláusula é útil durante scripting para que o script não falhe se o comando DROP VIEW for executado em uma exibição não existente.

name

Nome da exibição a ser removida.

CASCADE

Cláusula que indica que para remover automaticamente os objetos que dependem da exibição, como outras exibições.

Para criar uma exibição que não dependa de outros objetos de banco de dados, como exibições e tabelas, inclua a cláusula WITH NO SCHEMA BINDING na definição de exibição. Para obter mais informações, consulte CREATE VIEW.

Observe que, se você incluir CASCADE e a contagem de objetos de banco de dados descartados for dez ou mais, é possível que o cliente do banco de dados não liste todos os objetos descartados nos resultados resumidos. Isso geralmente ocorre porque as ferramentas do cliente SQL têm limitações padrão nos resultados exibidos.

RESTRICT

Cláusula que indica que a exibição não se deve ser removida se contiver objetos que dependam dela. Esta ação é o padrão.

Exemplos

O exemplo a seguir remove a exibição chamada event:

```
drop view event;
```

Para remover uma exibição com dependências, use a opção CASCADE. Por exemplo, suponha que você comece com uma tabela chamada EVENT. Em seguida, crie a visualização eventview da tabela EVENT usando o comando CREATE VIEW, conforme mostrado no exemplo a seguir:

```
create view eventview as
select dateid, eventname, catid
from event where catid = 1;
```

Agora, crie uma segunda visualização chamada myeventview com base na primeira visualização, eventview:

```
create view myeventview as
select eventname, catid
from eventview where eventname <> ' ';
```

Neste momento, duas exibições foram criadas: eventview e myeventview.

A exibição myeventview é dependente daeventview, que é a exibição principal.

Para excluir a exibição eventview, o comando óbvio a ser usado é o seguinte:

```
drop view eventview;
```

Observe que se você executa o comando nesse caso, você recebe o seguinte erro:

```
drop view eventview;
ERROR: can't drop view eventview because other objects depend on it
HINT: Use DROP ... CASCADE to drop the dependent objects too.
```

Para resolver isso, execute o seguinte comando (como sugerido na mensagem de erro):

```
drop view eventview cascade;
```

Agora as exibições eventview e myeventview foram removidas com êxito.

O exemplo a seguir remove a exibição eventview se existir, ou não faz nada e retorna uma mensagem se a exibição não existir:

```
drop view if exists eventview;
```

END

Confirma a transação atual. Executa exatamente a mesma função que o comando COMMIT.

Consulte COMMIT para obter uma documentação mais detalhada.

Sintaxe

```
END [ WORK | TRANSACTION ]
```

Parâmetros

WORK

Palavra-chave opcional.

TRANSACTION

Palavra-chave opcional; WORK e TRANSACTION são sinônimos.

Exemplos

Todos os exemplos a seguir encerram o bloco de transação e confirmam a transação:

```
end;
```

```
end work;
```

```
end transaction;
```

Depois de qualquer um desses comandos, o Amazon Redshift encerra o bloco de transação e confirma as alterações.

EXECUTE

Executa uma instrução preparada anteriormente.

Sintaxe

```
EXECUTE plan_name [ (parameter [, ...]) ]
```

Parâmetros

plan_name

O nome da instrução preparada para execução.

Parâmetro

O valor real de um parâmetro da instrução preparada. Essa deve ser uma expressão que resulta em um valor de um tipo compatível com o tipo de dados especificado para a posição do parâmetro no comando PREPARE que criou a instrução preparada.

Observações de uso

EXECUTE é usado para executar uma instrução preparada anteriormente. Como as instruções preparadas existem somente pela duração de uma sessão, a instrução preparada deve ter sido criada por uma instrução PREPARE executada anteriormente na sessão atual.

Se a instrução PREPARE anterior especificou alguns parâmetros, um conjunto de parâmetros compatíveis deve ser passado para a instrução EXECUTE ou o Amazon Redshift retorna um erro. Ao contrário das funções, as instruções preparadas não são sobrecarregadas com base no tipo nem na quantidade de parâmetros especificados. O nome de uma instrução preparada deve ser exclusivo em uma sessão de banco de dados.

Quando um comando EXECUTE é emitido para a instrução preparada, o Amazon Redshift pode optar por revisar o plano de execução da consulta (para melhorar a performance com base nos valores de parâmetro especificados) antes de executar a instrução preparada. Além disso, para cada nova execução de uma instrução preparada, o Amazon Redshift pode revisar o plano de execução da consulta novamente com base em valores de parâmetro diferentes especificados com a instrução EXECUTE. Para examinar o plano de execução da consulta escolhido pelo Amazon Redshift para qualquer comando EXECUTE, use o comando EXPLAIN.

Para mais exemplos e informações sobre a criação e o uso de instruções preparadas, consulte PREPARE.

Consulte também

DEALLOCATE, PREPARE

EXPLAIN

Exibe o plano de execução para uma instrução de consulta sem executar a consulta. Para obter informações sobre o fluxo de trabalho de análise de consultas, consulte Fluxo de trabalho da análise de consulta.

Sintaxe

```
EXPLAIN [ VERBOSE ] query
```

Parâmetros

VERBOSE

Exibe o plano completo da consulta em vez de apenas um resumo.

query

Instrução da consulta a ser explicada. A consulta pode ser uma instrução SELECT, INSERT, CREATE TABLE AS, UPDATE ou DELETE.

Observações de uso

Às vezes, a performance do comando EXPLAIN é influenciado pelo tempo que ele leva para criar tabelas temporárias. Por exemplo, uma consulta que usa otimização comum de subexpressão requer tabelas temporárias para ser criada e analisada para retornar a saída EXPLAIN. O plano da consulta depende do esquema e das estatísticas das tabelas temporárias. Portanto, para esse tipo de consulta, o comando EXPLAIN pode levar mais tempo para ser executado que o esperado.

Você pode usar EXPLAIN somente com os seguintes comandos:

- SELECT
- SELECT INTO
- CREATE TABLE AS
- INSERT
- UPDATE
- DELETE

O comando EXPLAIN falhará se você o usar para outros comandos SQL, como data definition language (DDL) ou operações de banco de dados.

Os custos unitários relativos da saída EXPLAIN são usados pelo Amazon Redshift para escolher um plano de consulta. O Amazon Redshift compara os tamanhos de várias estimativas de recursos para determinar o plano.

Planejamento de consulta e etapas de execução

O plano de execução de uma instrução de consulta específica do Amazon Redshift divide a execução e o cálculo de uma consulta em uma sequência diferente de etapas e operações de tabela que, por fim, produz um conjunto de resultados finais para a consulta. Para obter informações sobre consulta paralela, consulte Processamento de consulta.

A tabela a seguir fornece um resumo de etapas que o Amazon Redshift pode usar para desenvolver um plano de execução para qualquer consulta enviada por um usuário para execução.

Operadores EXPLAIN	Etapas de execução da consulta	Descrição
SCAN:		
Sequential Scan	scan	Operador ou etapa de varredura de relação ou de varredura de tabela do Amazon Redshift. Faz a varredura da tabela inteira sequencialmente do começo ao fim. Além disso, avalia as restrições de consulta de cada linha (Filtro), caso especificada na cláusula WHERE. Também é usado para executar as instruções INSERT, UPDATE e DELETE.
JOINS: o Amazon Redshift usa operadores de junção diferentes com base no design físico das tabelas sendo juntadas, na localização dos dados necessários para a junção e nos atributos específicos da própria consulta. Varredura de subconsulta -- A varredura e a anexação de subconsulta são usadas para executar consultas UNION.		
Loop aninhado	nloop	A junção menos ideal; usada sobretudo para junções cruzadas (produtos cartesianos, sem condição de junção) e algumas junções de desigualdade.
Hash Join	hjoin	Também usada para junções externas (esquerda e direita) e internas. Costuma ser mais rápida do que uma junção de loop aninhado. A junção hash lê a tabela externa, executa o hash nas coluna de junção e encontra correspondências na tabela interna de hash. A etapa pode incorrer em um vazamento para o disco. (A entrada interna de hjoin é uma etapa de hash que pode ser baseada em disco.)
Merge Join	mjoin	Também usada para junções internas e externas (para juntar tabelas que são distribuí

Operadores EXPLAIN	Etapas de execução da consulta	Descrição
		das e classificadas nas colunas de junção). Normalmente o algoritmo mais rápido de junção do Amazon Redshift, sem incluir outras considerações de custos.
AGREGAR: Operadores e etapas usados para consultas que envolvem funções agregadas e operações GROUP BY.		
Aggregate	aggr	Operador/etapa para funções agregadas escalares.
HashAggregate	aggr	Operador/etapa para funções agregadas agrupadas. Pode operar a partir do disco como consequência de vazamento de uma tabela de hash para o disco.
GroupAggregate	aggr	Operador escolhido às vezes para consultas agregadas agrupadas se a configuração do Amazon Redshift para force_hash_grouping está desativada.
CLASSIFICAR: Operadores e etapas usados quando as consultas têm que classificar ou mesclar conjuntos de resultados.		
Sort	sort	Classificar realiza a classificação especificada pela cláusula ORDER BY, assim como outras operações, como junções e consultas UNION. Pode operar a partir do disco.
Merge	merge	Produz resultados finais classificados de uma consulta com base em resultados intermedi ários classificados derivados de operações realizadas em paralelo.
Operações EXCEPT, INTERSECT e UNION:		

Operadores EXPLAIN	Etapas de execução da consulta	Descrição
SetOp Except [Distinct]	hjoin	Usado para consultas EXCEPT. Pode operar a partir do disco em virtude do fato de que o hash de entrada pode ser baseado em disco.
Hash Intersect [Distinct]	hjoin	Usado para consultas INTERSECT. Pode operar a partir do disco em virtude do fato de que o hash de entrada pode ser baseado em disco.
Append [All |Distinct]	save	Anexo usado com a Varredura de subconsulta para implementar as consultas UNION e UNION ALL. Pode operar a partir do disco em virtude de “Save.”
Diversos/Outros:		
Hash	hash	Usado para fazer junções internas e junções externas (esquerda e direita). Fornece entrada para uma junção hash. O operador Hash cria a tabela de hash para a tabela interna de uma junção. (A tabela interna é a tabela verificada para ver se há correspondências e, em uma junção de duas tabelas, geralmente é a menor das duas.)
Limite	limite	Avalia a cláusula LIMIT.
Materialize	save	Materializa linhas para entrada em junções de loop aninhado e algumas junções de mesclagem. Pode operar a partir do disco.
--	parse	Usada para analisar dados de entrada textual durante uma carga.

Operadores EXPLAIN	Etapas de execução da consulta	Descrição
--	project	Usada para reorganizar colunas e expressões de computação, ou seja, projetar dados.
Resultado	--	Executa funções escalares que não envolvem acesso à tabela.
--	return	Retorna linhas ao principal ou ao cliente.
Subplan	--	Usado para determinadas subconsultas.
Unique	unique	Elimina duplicidades das consultas SELECT DISTINCT e UNION.
Window	window	Computa funções agregadas e funções da janela de classificação. Pode operar a partir do disco.
Operações de rede:		
Network (Broadcast)	bcast	Broadcast também é um atributo dos operadores e das etapas de Join Explain.
Network (Distribute)	dist	Distribui linhas para computar nós para processamento paralelo pelo cluster de data warehouse.
Network (Send to Leader)	return	Envia resultados de volta ao principal para processamento adicional.
Operações de DML (operadores que alteram dados):		
Insert (using Result)	insert	Insere dados.
Delete (Scan + Filter)	excluir	Exclui dados. Pode operar a partir do disco.
Update (Scan + Filter)	delete, insert	Implementado como exclusão e inserção.

Utilização de EXPLAIN para RLS

Se uma consulta contiver uma tabela sujeita às políticas de segurança no nível da linha (RLS), EXPLAIN exibirá um nó especial do RLS SecureScan. O Amazon Redshift também registra o mesmo tipo de nó na tabela do sistema STL_EXPLAIN. EXPLAIN não revela o predicado RLS que se aplica a dim_tbl. O tipo de nó RLS SecureScan serve como um indicador de que o plano de execução contém operações adicionais que são invisíveis para o usuário atual.

O exemplo a seguir mostra um nó RLS SecureScan.

```
EXPLAIN
SELECT D.cint
FROM fact_tbl F INNER JOIN dim_tbl D ON F.k_dim = D.k
WHERE F.k_dim / 10 > 0;
                               QUERY PLAN
------------------------------------------------------------------------
 XN Hash Join DS_DIST_ALL_NONE  (cost=0.08..0.25 rows=1 width=4)
   Hash Cond: ("outer".k_dim = "inner"."k")
   ->  *XN* *RLS SecureScan f  (cost=0.00..0.14 rows=2 width=4)*
         Filter: ((k_dim / 10) > 0)
   ->  XN Hash  (cost=0.07..0.07 rows=2 width=8)
         ->  XN Seq Scan on dim_tbl d  (cost=0.00..0.07 rows=2 width=8)
               Filter: (("k" / 10) > 0)
```

Para permitir a investigação completa dos planos de consulta sujeitos ao RLS, o Amazon Redshift oferece as permissões do sistema EXPLAIN RLS. Os usuários que receberam essa permissão podem inspecionar planos de consulta completos que também incluem predicados RLS.

O exemplo a seguir ilustra uma Seq Scan adicional abaixo do nó RLS SecureScan que também inclui o predicado da política de RLS (k_dim > 1).

```
EXPLAIN SELECT D.cint
FROM fact_tbl F INNER JOIN dim_tbl D ON F.k_dim = D.k
WHERE F.k_dim / 10 > 0;
                                    QUERY PLAN
-------------------------------------------------------------------------------
 XN Hash Join DS_DIST_ALL_NONE  (cost=0.08..0.25 rows=1 width=4)
   Hash Cond: ("outer".k_dim = "inner"."k")
   *->  XN RLS SecureScan f  (cost=0.00..0.14 rows=2 width=4)
         Filter: ((k_dim / 10) > 0)*
         ->  *XN* *Seq Scan on fact_tbl rls_table  (cost=0.00..0.06 rows=5 width=8)
```

```
               Filter: (k_dim > 1)*
   ->  XN Hash  (cost=0.07..0.07 rows=2 width=8)
         ->  XN Seq Scan on dim_tbl d  (cost=0.00..0.07 rows=2 width=8)
               Filter: (("k" / 10) > 0)
```

Embora a permissão EXPLAIN RLS seja concedida a um usuário, o Amazon Redshift registra o plano de consulta completo, incluindo predicados RLS na tabela do sistema STL_EXPLAIN. As consultas que forem executadas enquanto essa permissão não for concedida serão registradas sem os internos do RLS. Conceder ou remover a permissão EXPLAIN RLS não alterará o que o Amazon Redshift registrou em log no STL_EXPLAIN para consultas anteriores.

AWS Lake FormationRelações do Redshift protegidas entre e RLS

O exemplo a seguir ilustra um nó LF SecureScan, que você pode usar para visualizar relações entre Lake Formation e RLS.

```
EXPLAIN
SELECT *
FROM lf_db.public.t_share
WHERE a > 1;
QUERY PLAN
------------------------------------------------------------
XN LF SecureScan t_share  (cost=0.00..0.02 rows=2 width=11)
(2 rows)
```

Exemplos

> **Note**
>
> Para esses exemplos, o resultado da amostra pode variar dependendo da configuração do Amazon Redshift.

O exemplo a seguir retorna o plano de consulta para uma consulta que seleciona EVENTID, EVENTNAME, VENUEID e VENUENAME das tabelas EVENT e VENUE:

```
explain
select eventid, eventname, event.venueid, venuename
from event, venue
```

```
where event.venueid = venue.venueid;
```

```
                                QUERY PLAN
------------------------------------------------------------------------------
XN Hash Join DS_DIST_OUTER  (cost=2.52..58653620.93 rows=8712 width=43)
Hash Cond: ("outer".venueid = "inner".venueid)
->  XN Seq Scan on event  (cost=0.00..87.98 rows=8798 width=23)
->  XN Hash  (cost=2.02..2.02 rows=202 width=22)
->  XN Seq Scan on venue  (cost=0.00..2.02 rows=202 width=22)
(5 rows)
```

O exemplo a seguir retorna o plano de consulta para a mesma consulta com saída "verbose":

```
explain verbose
select eventid, eventname, event.venueid, venuename
from event, venue
where event.venueid = venue.venueid;
```

```
                                QUERY PLAN
------------------------------------------------------------------------------
{HASHJOIN
:startup_cost 2.52
:total_cost 58653620.93
:plan_rows 8712
:plan_width 43
:best_pathkeys <>
:dist_info DS_DIST_OUTER
:dist_info.dist_keys (
TARGETENTRY
{
VAR
:varno 2
:varattno 1
...

XN Hash Join DS_DIST_OUTER  (cost=2.52..58653620.93 rows=8712 width=43)
Hash Cond: ("outer".venueid = "inner".venueid)
->  XN Seq Scan on event  (cost=0.00..87.98 rows=8798 width=23)
->  XN Hash  (cost=2.02..2.02 rows=202 width=22)
->  XN Seq Scan on venue  (cost=0.00..2.02 rows=202 width=22)
(519 rows)
```

O exemplo a seguir retorna o plano de consulta para uma instrução CREATE TABLE AS (CTAS):

```
explain create table venue_nonulls as
select * from venue
where venueseats is not null;

QUERY PLAN
------------------------------------------------------------
XN Seq Scan on venue  (cost=0.00..2.02 rows=187 width=45)
Filter: (venueseats IS NOT NULL)
(2 rows)
```

FETCH

Recupera linhas usando um cursor. Para obter informações sobre como declarar um cursor, consulte DECLARE.

FETCH recupera linhas com base na posição atual do cursor. Quando um cursor é criado, ele é posicionado antes da primeira linha. Depois de um comando FETCH, o cursor é posicionado na última linha recuperada. Se FETCH for executado até o final das linhas disponíveis, como um comando FETCH ALL, o cursor é posicionado depois da última linha.

FORWARD 0 busca a linha atual sem mover o cursor, ou seja, procura a linha buscada mais recentemente. Se o cursor estiver posicionado antes da primeira linha ou depois da última linha, nenhuma linha será retornada.

Quando a primeira linha do cursor é buscada, o conjunto inteiro de resultados é materializado no nó líder, na memória ou em disco, se necessário. Devido ao impacto da performance negativa potencial de usar cursores com grandes conjuntos de resultado, recomendamos abordagens alternativas sempre que possível. Para obter mais informações, consulte Considerações sobre a performance ao usar cursores.

Para obter mais informações, consulte DECLARECLOSE.

Sintaxe

```
FETCH [ NEXT | ALL | {FORWARD [ count | ALL ] } ] FROM cursor
```

Parâmetros

NEXT

Busca a próxima linha. Esse é o padrão.

ALL

Busca todas as linhas restantes. (Mesmo que FORWARD ALL.) ALL não é compatível para clusters de nó único.

FORWARD [contagem | ALL]

Busca a contagem das próximas linhas, ou todas as linhas restantes. `FORWARD 0` busca a linha atual. Para clusters de nó único, o valor máximo para a contagem é `1000`. FORWARD ALL não é compatível para clusters de nó único.

cursor

Nome do novo cursor.

Exemplo de FETCH

O exemplo a seguir declara um cursor denominado LOLLAPALOOZA para selecionar informações de vendas para o evento Lollapalooza e busca linhas do conjunto de resultados usando o cursor:

```
-- Begin a transaction

begin;

-- Declare a cursor

declare lollapalooza cursor for
select eventname, starttime, pricepaid/qtysold as costperticket, qtysold
from sales, event
where sales.eventid = event.eventid
and eventname='Lollapalooza';

-- Fetch the first 5 rows in the cursor lollapalooza:

fetch forward 5 from lollapalooza;

  eventname   |      starttime      | costperticket | qtysold
```

```
-------------+---------------------+--------------+---------
 Lollapalooza | 2008-05-01 19:00:00 |   92.00000000 |       3
 Lollapalooza | 2008-11-15 15:00:00 |  222.00000000 |       2
 Lollapalooza | 2008-04-17 15:00:00 |  239.00000000 |       3
 Lollapalooza | 2008-04-17 15:00:00 |  239.00000000 |       4
 Lollapalooza | 2008-04-17 15:00:00 |  239.00000000 |       1
(5 rows)

-- Fetch the next row:

fetch next from lollapalooza;

  eventname   |      starttime      | costperticket | qtysold
-------------+---------------------+--------------+---------
 Lollapalooza | 2008-10-06 14:00:00 |  114.00000000 |       2

-- Close the cursor and end the transaction:

close lollapalooza;
commit;
```

GRANT

Define permissões de acesso para um usuário ou uma função.

As permissões incluem opções de acesso como leitura de dados em tabelas e visualizações, gravação de dados e criação de tabelas. Use este comando para fornecer permissões específicas para uma tabela, banco de dados, esquema, função, procedimento, linguagem ou coluna. Para revogar permissões de um objeto de banco de dados, use o comando REVOKE.

As permissões também incluem as seguintes opções de acesso do produtor da unidade de compartilhamento de dados:

- Concessão de acesso à unidade de compartilhamento de dados para namespaces e contas de consumidores.
- Concessão de permissão para alterar uma unidade de compartilhamento de dados adicionando ou removendo objetos da unidade de compartilhamento de dados.
- Concessão de permissão para compartilhar uma unidade de compartilhamento de dados adicionando ou removendo namespaces de consumidores da unidade de compartilhamento de dados.

As opções de acesso do consumidor à unidade de compartilhamento de dados são as seguintes:

- Concessão a usuários de acesso total a bancos de dados criados a partir de uma unidade de compartilhamento de dados ou a esquemas externos que apontem para esses bancos de dados.
- Concessão a usuários de permissões no nível do objeto em bancos de dados criados a partir de uma unidade de compartilhamento de dados como a que você pode para objetos de banco de dados. Para conceder esse nível de permissão, você deve usar a cláusula WITH PERMISSIONS ao criar um banco de dados a partir da unidade de compartilhamento de dados. Para obter mais informações, consulte CREATE DATABASE.

Para obter mais informações sobre permissões da unidade de compartilhamento de dados, consulte Permissões que você pode conceder a unidades de compartilhamento de dados.

As permissões também incluem o seguinte Catálogo de Permissões Federadas do Amazon Redshift:

- Conceder permissões em nível de tabela a usuários e perfis.
- Conceder permissões refinadas em nível de coluna em tabelas, visualizações e visões materializadas.
- Conceder permissões com escopo a usuários e perfis.
- Conceder permissões em nível de banco de dados no Catálogo de Permissões Federadas do Amazon Redshift.

Para acessar mais informações sobre o gerenciamento de permissões no Catálogo de Permissões Federadas do Amazon Redshift, consulte Gerenciar o controle de acesso no catálogo de permissões federadas do Amazon Redshift. Para acessar mais informações sobre as sintaxes de concessão/revogação aceitas pelo Catálogo de Permissões Federadas do Amazon Redshift, consulte Concessão/revogação.

As permissões também incluem o privilégio CONNECT para usuários federados do Centro de Identidade do AWS IAM. Esse privilégio permite que os administradores controlem o acesso do usuário por meio de permissões granulares em cada grupo de trabalho ou cluster do Amazon Redshift em que as permissões federadas do Amazon Redshift estão habilitadas. O administrador do Amazon Redshift pode especificar quais usuários ou grupos federados do Centro de Identidade do AWS IAM têm acesso para se conectar diretamente ao grupo de trabalho do Amazon Redshift, fornecendo controle refinado sobre o acesso do usuário em cada grupo de trabalho ou cluster do Centro de Identidade do AWS IAM.

Você também pode conceder perfis para gerenciar permissões de banco de dados e controlar o que os usuários podem fazer em relação aos seus dados. Ao definir perfis e atribuí-los aos usuários, você pode limitar as ações que esses usuários podem realizar, como limitar os usuários somente aos comandos CREATE TABLE e INSERT. Para obter mais informações sobre o comando CREATE ROLE, consulte the section called “CREATE ROLE”. O Amazon Redshift tem alguns perfis definidos pelo sistema que você também pode usar para conceder permissões específicas aos seus usuários. Para obter mais informações, consulte the section called “Funções definidas pelo sistema do Amazon Redshift”.

Só é possível CONCEDER ou REVOGAR permissões de USO em um esquema externo para usuários de banco de dados e grupos de usuários que usam a sintaxe ON SCHEMA. Ao usar ON EXTERNAL SCHEMA com AWS Lake Formation, só é possível conceder (GRANT) e revogar (REVOKE) permissões para um perfil do AWS Identity and Access Management (IAM). Para obter a lista de permissões, consulte a sintaxe.

Para procedimentos armazenados, a única permissão que pode ser concedida é EXECUTE.

Não é possível executar GRANT (em um recurso externo) em um bloco de transação (BEGIN ... END). Para obter mais informações sobre transações, consulte Níveis de isolamento no Amazon Redshift.

Para ver quais permissões os usuários receberam para um banco de dados, use HAS_DATABASE_PRIVILEGE. Para ver quais permissões os usuários receberam para um esquema, use HAS_SCHEMA_PRIVILEGE. Para ver quais permissões os usuários receberam para uma tabela, use HAS_TABLE_PRIVILEGE.

Sintaxe

```
GRANT { { SELECT | INSERT | UPDATE | DELETE | DROP | REFERENCES | ALTER | TRUNCATE }
 [,...] | ALL [ PRIVILEGES ] }
    ON { [ TABLE ] table_name [, ...] | ALL TABLES IN SCHEMA schema_name [, ...] }
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

GRANT { { CREATE | USAGE | TEMPORARY | TEMP | ALTER } [,...] | ALL [ PRIVILEGES ] }
    ON DATABASE db_name [, ...]
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]
```

```
GRANT { { CREATE | USAGE | ALTER | DROP } [,...] | ALL [ PRIVILEGES ] }
    ON SCHEMA schema_name [, ...]
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

GRANT { EXECUTE | ALL [ PRIVILEGES ] }
    ON { FUNCTION function_name ( [ [ argname ] argtype [, ...] ] ) [, ...] | ALL
 FUNCTIONS IN SCHEMA schema_name [, ...] }
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

GRANT { EXECUTE | ALL [ PRIVILEGES ] }
    ON { PROCEDURE procedure_name ( [ [ argname ] argtype [, ...] ] ) [, ...] | ALL
 PROCEDURES IN SCHEMA schema_name [, ...] }
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

GRANT USAGE
    ON LANGUAGE language_name [, ...]
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

GRANT { { ALTER | DROP} [,...] | ALL [ PRIVILEGES ] }
    ON COPY JOB job_name [,...]
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

GRANT { { ALTER | DROP | USAGE } [,...] | ALL [ PRIVILEGES ] }
    ON TEMPLATE [database_name.][schema_name.]template_name [,...]
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]
```

Conceder permissões por coluna para tabelas

A seguir está a sintaxe para permissões por coluna em tabelas e visualizações do Amazon Redshift.

```
GRANT { { SELECT | UPDATE } ( column_name [, ...] ) [, ...] | ALL [ PRIVILEGES ]
 ( column_name [,...] ) }
     ON { [ TABLE ] table_name [, ...] }

     TO { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
```

Conceder permissões ASSUMEROLE

A seguir está a sintaxe para as permissões ASSUMEROLE concedidas a usuários e grupos com um perfil especificado. Para começar a usar o privilégio ASSUMEROLE, consulte Observações de uso para conceder a permissão ASSUMEROLE.

```
GRANT ASSUMEROLE
       ON { 'iam_role' [, ...] | default | ALL }
       TO { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
       FOR { ALL | COPY | UNLOAD | EXTERNAL FUNCTION | CREATE MODEL } [, ...]
```

Conceder permissões para integração do Redshift Spectrum com o Lake Formation

A seguir está a sintaxe para integração do Redshift Spectrum com Lake Formation.

```
GRANT { SELECT | ALL [ PRIVILEGES ] } ( column_list )
    ON EXTERNAL TABLE schema_name.table_name
    TO { IAM_ROLE iam_role } [, ...] [ WITH GRANT OPTION ]

GRANT { { SELECT | ALTER | DROP | DELETE | INSERT }  [, ...] | ALL [ PRIVILEGES ] }
    ON EXTERNAL TABLE schema_name.table_name [, ...]
    TO { { IAM_ROLE iam_role } [, ...] | PUBLIC } [ WITH GRANT OPTION ]

GRANT { { CREATE | ALTER | DROP }  [, ...] | ALL [ PRIVILEGES ] }
    ON EXTERNAL SCHEMA schema_name [, ...]
    TO { IAM_ROLE iam_role } [, ...] [ WITH GRANT OPTION ]
```

Conceder permissões da unidade de compartilhamento de dados

Permissões da unidade de compartilhamento de dados no lado do produtor

Esta é a sintaxe para usar GRANT a fim de conceder permissões ALTER ou SHARE a um usuário ou perfil. O usuário pode alterar a unidade de compartilhamento de dados com a permissão ALTER ou conceder o uso a um consumidor com a permissão SHARE. ALTER e SHARE são as únicas permissões que você pode conceder em uma unidade de compartilhamento de dados a usuários e funções.

```
GRANT { ALTER | SHARE } ON DATASHARE datashare_name
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]
```

A seguir está a sintaxe para usar GRANT para permissões de uso de unidade de compartilhamento de dados no Amazon Redshift. Você concede acesso de uma unidade de compartilhamento de dados a um consumidor usando a permissão USAGE. Você não pode conceder essa permissão a usuários ou grupos de usuários. Essa permissão também não é compatível com WITH GRANT OPTION para a instrução GRANT. Somente usuários ou grupos de usuários com a permissão SHARE concedida anteriormente a eles para (FOR) a unidade de compartilhamento de dados podem executar esse tipo de instrução GRANT.

```
GRANT USAGE
    ON DATASHARE datashare_name
    TO NAMESPACE 'namespaceGUID' | ACCOUNT 'accountnumber' [ VIA DATA CATALOG ]
```

Veja a seguir um exemplo de como conceder o uso de uma unidade de compartilhamento de dados a uma conta do Lake Formation.

```
GRANT USAGE ON DATASHARE salesshare TO ACCOUNT '123456789012' VIA DATA CATALOG;
```

Permissões da unidade de compartilhamento de dados no lado do consumidor

A seguir está a sintaxe para permissões de uso de compartilhamento de dados GRANT em um banco de dados específico ou esquema criado a partir de um datashare.

Outras permissões necessárias para consumidores terem acesso a um banco de dados criado a partir de uma unidade de compartilhamento de dados variam dependendo do comando CREATE DATABASE usado para criar o banco de dados a partir da unidade de compartilhamento de dados ter usado ou não a cláusula WITH PERMISSIONS. Para obter mais informações sobre o comando CREATE DATABASE e a cláusula WITH PERMISSIONS, consulte CREATE DATABASE.

Bancos de dados criados sem usar a cláusula WITH PERMISSIONS

Ao conceder USAGE em um banco de dados criado a partir de uma unidade de compartilhamento de dados sem a cláusula WITH PERMISSIONS, você não precisa conceder permissões separadamente nos objetos no banco de dados compartilhado. As entidades com uso concedido em bancos de dados criados a partir de unidades de compartilhamento de dados sem a cláusula WITH PERMISSIONS têm acesso automático a todos os objetos no banco de dados.

Bancos de dados criados usando a cláusula WITH PERMISSIONS

Quando você concede USAGE em um banco de dados no qual o banco de dados compartilhado foi criado a partir de uma unidade de compartilhamento de dados com a cláusula WITH PERMISSIONS,

as identidades no lado do consumidor ainda devem receber as permissões relevantes para objetos de banco de dados no banco de dados compartilhado para terem acesso a eles, assim como você concederia permissões para objetos de banco de dados locais. Para conceder permissões a objetos em um banco de dados criado a partir de uma unidade de compartilhamento de dados, use a sintaxe de três partes `database_name.schema_name.object_name`. Para conceder permissões a objetos em um esquema externo apontando para um esquema compartilhado no banco de dados compartilhado, use a sintaxe de duas partes `schema_name.object_name`.

```
GRANT USAGE ON { DATABASE shared_database_name [, ...] | SCHEMA shared_schema}
    TO { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
```

Concessão de permissões em escopo

Com as permissões em escopo, é possível conceder permissões a um usuário ou função em todos os objetos de um tipo em um banco de dados ou esquema. Usuários e funções com permissões em escopo têm as permissões especificadas em todos os objetos atuais e futuros no banco de dados ou esquema.

Você pode visualizar o escopo das permissões com escopo em nível de banco de dados em SVV_DATABASE_PRIVILEGES. Você pode visualizar o escopo das permissões com escopo em nível de esquema em SVV_SCHEMA_PRIVILEGES.

Para ter mais informações sobre permissões com escopo definido, consulte Permissões em escopo.

Esta é a sintaxe para concessão de permissões em escopo para usuários ou perfis.

```
GRANT { CREATE | USAGE | ALTER | DROP } [,...] | ALL [ PRIVILEGES ] }
FOR SCHEMAS IN
DATABASE db_name
TO { username [ WITH GRANT OPTION ] | ROLE role_name } [, ...]

GRANT
{ { SELECT | INSERT | UPDATE | DELETE | DROP | ALTER | TRUNCATE | REFERENCES }
 [, ...] } | ALL [PRIVILEGES] } }
FOR TABLES IN
{SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
TO { username [ WITH GRANT OPTION ] | ROLE role_name} [, ...]

GRANT { EXECUTE | ALL [ PRIVILEGES ] }
FOR FUNCTIONS IN
{SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
```

```
TO { username [ WITH GRANT OPTION ] | ROLE role_name | } [, ...]

GRANT { EXECUTE | ALL [ PRIVILEGES ] }
FOR PROCEDURES IN
{SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
TO { username [ WITH GRANT OPTION ] | ROLE role_name | } [, ...]

GRANT USAGE
FOR LANGUAGES IN
{DATABASE db_name}
TO { username [ WITH GRANT OPTION ] | ROLE role_name } [, ...]

GRANT { { CREATE | ALTER | DROP} [,...] | ALL [ PRIVILEGES ] }
FOR COPY JOBS
IN DATABASE db_name
TO { username [ WITH GRANT OPTION ] | ROLE role_name } [, ...]

GRANT { { ALTER | DROP | USAGE } [,...] | ALL [ PRIVILEGES ] }
FOR TEMPLATES IN
{SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
TO { username [ WITH GRANT OPTION ] | ROLE role_name } [, ...]
```

Observe que as permissões no escopo não fazem distinção entre permissões para funções e procedimentos. Por exemplo, a declaração a seguir concede a bob a permissão EXECUTE para funções e procedimentos no esquema `Sales_schema`.

```
GRANT EXECUTE FOR FUNCTIONS IN SCHEMA Sales_schema TO bob;
```

Conceder permissões de machine learning

A seguir está a sintaxe para permissões de modelos de machine learning no Amazon Redshift.

```
GRANT CREATE MODEL
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

GRANT { EXECUTE | ALL [ PRIVILEGES ] }
    ON MODEL model_name [, ...]

    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]
```

Conceder permissões de perfil

A sintaxe a seguir concede perfis no Amazon Redshift.

```
GRANT { ROLE role_name } [, ...] TO { { user_name [ WITH ADMIN OPTION ] } |
 ROLE role_name }[, ...]
```

A sintaxe a seguir concede permissões de sistema a perfis no Amazon Redshift. Observe que você só pode conceder permissões para perfis, não para usuários.

```
GRANT
  {
    { CREATE USER | DROP USER | ALTER USER |
    CREATE SCHEMA | DROP SCHEMA |
    ALTER DEFAULT PRIVILEGES |
    ACCESS CATALOG | ACCESS SYSTEM TABLE
    CREATE TABLE | DROP TABLE | ALTER TABLE |
    CREATE OR REPLACE FUNCTION | CREATE OR REPLACE EXTERNAL FUNCTION |
    DROP FUNCTION |
    CREATE OR REPLACE PROCEDURE | DROP PROCEDURE |
    CREATE OR REPLACE VIEW | DROP VIEW |
    CREATE MODEL | DROP MODEL |
    CREATE DATASHARE | ALTER DATASHARE | DROP DATASHARE |
    CREATE LIBRARY | DROP LIBRARY |
    CREATE ROLE | DROP ROLE |
    TRUNCATE TABLE
    VACUUM | ANALYZE | CANCEL |
    IGNORE RLS | EXPLAIN RLS |
    EXPLAIN MASKING }[, ...]
  }
  | { ALL [ PRIVILEGES ] }
TO ROLE role_name [, ...]
```

Conceder permissões de explicação para políticas de segurança

Veja a seguir a sintaxe para conceder permissões para explicar os filtros de política de segurança de uma consulta no plano EXPLAIN. As políticas de segurança possíveis incluem segurança por linha e políticas de mascaramento dinâmico de dados.

```
GRANT EXPLAIN { RLS | MASKING } TO ROLE rolename
```

Veja a seguir a sintaxe para conceder permissões para ignorar políticas de segurança no nível da linha para uma consulta. Essa sintaxe não se aplica às políticas de mascaramento dinâmico de dados.

```
GRANT IGNORE RLS TO ROLE rolename
```

Veja a seguir a sintaxe para conceder permissões de tabela de pesquisa à política de segurança especificada. As políticas de segurança possíveis incluem segurança por linha e políticas de mascaramento dinâmico de dados.

```
GRANT SELECT ON [ TABLE ] table_name [, ...]
TO { RLS | MASKING } POLICY policy_name [, ...]
```

Conceder permissões de conexão

Veja a seguir a sintaxe para conceder permissões para usuários federados (ou grupos) do Centro de Identidade do AWS IAM se conectarem a um grupo de trabalho/cluster:

```
GRANT CONNECT [ON WORKGROUP]
TO <prefix>:<username> | ROLE <prefix>:<rolename> | PUBLIC;
```

Conceder a permissão DEBUG

Veja a seguir a sintaxe para conceder a permissão DEBUG em um banco de dados no warehouse do Amazon Redshift com permissões federadas. Você pode conceder a permissão DEBUG a uma identidade global, a um grupo IdC (Centro de Identidade do AWS IAM) ou aos administradores de uma conta de consumidor. Para começar a usar a permissão DEBUG, consulte Observações de uso para conceder a permissão DEBUG.

```
GRANT DEBUG
    ON DATABASE federated_db_name
    TO { <prefix>:<name>
       | ROLE <idc_prefix>:<group_name>
       | ADMINS OF ACCOUNT 'account_id' }
```

Parâmetros

SELECT

Concede permissão para selecionar dados de uma tabela ou visualização usando uma instrução SELECT. A permissão SELECT também é necessária para fazer referência a valores de coluna existentes para as operações UPDATE ou DELETE.

INSERT

Concede permissão para carregar dados em uma tabela usando uma instrução INSERT ou COPY.

UPDATE

Concede permissão para atualizar uma coluna de tabela usando a instrução UPDATE. As operações UPDATE também exigem a permissão SELECT, pois devem fazer referência a colunas da tabela para determinar que linhas atualizar ou computar novos valores para as colunas.

DELETE

Concede permissão para excluir uma linha de dados de uma tabela. As operações DELETE também necessitam do privilégio SELECT, pois devem fazer referência a colunas da tabela para determinar que linhas excluir.

DROP

Dependendo do objeto do banco de dados, concede as seguintes permissões ao usuário ou perfil:

- Para tabelas, DROP concede permissão para remover uma tabela ou visualização. Para obter mais informações, consulte DESCARTAR TABELA.
- Para bancos de dados, DROP concede permissão para remover um banco de dados. Para obter mais informações, consulte DROP DATABASE.
- Para esquemas, DROP concede permissão para remover um esquema. Para obter mais informações, consulte DROP SCHEMA.

REFERENCES

Concede permissão para criar uma restrição de chave estrangeira. Você precisa conceder essa permissão na tabela referenciada e na tabela que faz a referência. Caso contrário, o usuário não poderá criar a restrição.

ALTER

Dependendo do objeto do banco de dados, as seguintes permissões são concedidas ao usuário ou ao grupo de usuários:

- Para tabelas, ALTER concede permissão para alterar uma tabela ou visão. Para obter mais informações, consulte ALTER TABLE.
- Para bancos de dados, ALTER concede permissão para alterar um banco de dados. Para obter mais informações, consulte ALTER DATABASE.
- Para esquemas, ALTER concede permissão para alterar um esquema. Para obter mais informações, consulte ALTER SCHEMA.
- Para tabelas externas, ALTER concede permissão para alterar uma tabela em um AWS Glue Data Catalog habilitado para o Lake Formation. Essa permissão só é aplicada ao usar o Lake Formation.

TRUNCATE

Concede permissão para truncar uma tabela. Sem essa permissão, somente o proprietário de uma tabela ou um superusuário pode truncar uma tabela. Para obter mais informações sobre o comando TRUNCATE, consulte the section called “TRUNCATE”.

ALL [PRIVILEGES]

Concede todas as permissões disponíveis de uma vez ao usuário ou ao perfil especificado. A palavra-chave PRIVILEGES é opcional.

GRANT ALL ON SCHEMA não concede permissões CREATE para esquemas externos.

É possível conceder a permissão ALL para uma tabela em um AWS Glue Data Catalog habilitado para o Lake Formation. Nesse caso, permissões individuais (como SELECT, ALTER e assim por diante) são registradas no catálogo de dados.

> **Note**
>
> O Amazon Redshift não é compatível com as permissões RULE e TRIGGER. Para obter mais informações, acesse Recursos incompatíveis do PostgreSQL.

ASSUMEROLE

Concede permissões para executar comandos COPY, UNLOAD, EXTERNAL FUNCTION e CREATE MODEL a usuários, perfis ou grupos com um perfil especificado. O usuário, perfil ou

grupo assume esse perfil ao executar o comando especificado. Para começar a usar a permissão ASSUMEROLE, consulte Observações de uso para conceder a permissão ASSUMEROLE.

ON [TABLE] table_name

Concede as permissões especificadas em uma tabela ou visualização. A palavra-chave TABLE é opcional. Você pode listar várias tabelas e exibições em uma instrução.

ON ALL TABLES IN SCHEMA schema_name

Concede as permissões especificadas em todas as tabelas e exibições no esquema de referência.

(column_name [,...]) ON TABLE table_name

Concede as permissões especificadas a usuários, grupos ou PUBLIC nas colunas especificadas da tabela ou visualização do Amazon Redshift.

(column_list) ON EXTERNAL TABLE schema_name.table_name

Concede as permissões especificadas para um perfil do IAM nas colunas especificadas da tabela do Lake Formation no esquema mencionado.

ON EXTERNAL TABLE schema_name.table_name

Concede as permissões especificadas para um perfil do IAM nas tabelas especificadas do Lake Formation no esquema mencionado.

ON EXTERNAL SCHEMA schema_name

Concede as permissões especificadas para um perfil do IAM no esquema mencionado.

ON iam_role

Concede as permissões especificadas para um perfil do IAM.

TO username

Indica o usuário que está recebendo as permissões.

TO IAM_ROLE iam_role

Indica o perfil do IAM que está recebendo as permissões.

WITH GRANT OPTION

Indica que o usuário que recebe as permissões pode, por sua vez, conceder as mesmas permissões a outros. WITH GRANT OPTION não pode ser concedido a um grupo ou a PUBLIC.

ROLE role_name

Concede as permissões ao perfil.

GROUP nome_grupo

Concede as permissões a um grupo de usuários. Pode ser uma lista separada por vírgulas para especificar vários grupos de usuários.

PUBLIC

Concede as permissões especificadas a todos os usuários, incluindo os usuários criados posteriormente. PUBLIC representa um grupo que inclui sempre todos os usuários. As permissões de um usuário individual consistem na soma das permissões concedidas a PUBLIC, das permissões concedidas a todos os grupos aos quais o usuário pertence e de quaisquer permissões concedidas ao usuário individualmente.

Conceder PUBLIC a um EXTERNAL TABLE do Lake Formation resulta em conceder a permissão para o grupo todos do Lake Formation.

CONNECT [ON WORKGROUP] TO { <prefix>:<username> | ROLE <prefix>:<rolename> | PUBLIC }

Concede a permissão para se conectar a um grupo de trabalho ou cluster a usuários ou grupos federados do Centro de Identidade do AWS IAM. O prefixo identifica o provedor de identidades. Quando concedida a PUBLIC, a permissão se aplica a todos os usuários federados do Centro de Identidade do AWS IAM, incluindo os usuários criados posteriormente. Essa permissão é aplicável somente quando as permissões federadas do Amazon Redshift estão habilitadas no grupo de trabalho ou cluster.

CREATE

Dependendo do objeto do banco de dados, as seguintes permissões são concedidas ao usuário ou ao grupo de usuários:

- Para bancos de dados, CREATE permite que os usuários criem esquemas no banco de dados.
- Para esquemas, CREATE permite que os usuários criem objetos em um esquema. Para renomear um objeto, o usuário deve ter a permissão CREATE e ser proprietário do objeto a ser renomeado.
- Não há suporte para CREATE ON SCHEMA nos esquemas externos do Amazon Redshift Spectrum. Para conceder o uso de tabelas externas em um esquema externo, conceda USAGE ON SCHEMA a usuários que precisam de acesso. Somente o proprietário de um esquema externo ou um superusuário tem permissão para criar tabelas externas no esquema externo.

Para transferir a propriedade de um esquema externo, use ALTER SCHEMA para alterar o proprietário.

TEMPORARY | TEMP

Concede permissão para criar tabelas temporárias no banco de dados especificado. Para executar as consultas do Amazon Redshift Spectrum, o usuário do banco de dados precisa ter permissão para criar tabelas temporárias no banco de dados.

> **Note**
>
> Por padrão, os usuários recebem permissão para criar tabelas temporárias por associação automática ao grupo PUBLIC. Para remover a permissão para que qualquer usuário crie tabelas temporárias, revogue a permissão TEMP do grupo PUBLIC. Em seguida, conceda explicitamente a permissão para criar tabelas temporárias para usuários ou grupos de usuários específicos.

ON DATABASE nome_bd

Concede as permissões especificadas em um banco de dados.

DEBUG

Concede permissão para ver registros em log de registro seguros não editados para um banco de dados em um warehouse do Amazon Redshift com permissões federadas. Por padrão, quando uma consulta acessa dados de controle de acesso refinado (FGAC) nesse banco de dados, o Amazon Redshift mascara valores confidenciais nos registros de log que são gerados para essa consulta. A concessão de DEBUG permite que o beneficiário especificado veja esses registros de log sem supressão. Você pode conceder a permissão DEBUG somente em um banco de dados em um warehouse do Amazon Redshift com permissões federadas e somente para uma identidade global, um grupo do IdC ou os administradores de uma conta de consumidor. Para acessar mais informações sobre registro em log seguro, consulte Registro em log seguro no Guia de gerenciamento do Amazon Redshift; para notas de uso, consulte Observações de uso para conceder a permissão DEBUG.

ADMINS OF ACCOUNT 'account_id'

Para a permissão DEBUG, concede a permissão a todos os superusuários na conta de consumidor especificada. Os registros autorizados dessa forma também não são suprimidos quando o Amazon Redshift exporta dados da tabela do sistema para as Tabelas do Amazon S3.

Esse formulário de beneficiário é compatível somente com a permissão DEBUG, e a conta que você especificar deve ser a mesma conta que possui o produtor.

USAGE

Concede a permissão USAGE em um esquema específico, que torna objetos naquele esquema acessíveis aos usuários. As ações específicas nesses objetos devem ser concedidas separadamente (por exemplo, permissões SELECT ou UPDATE em tabelas) para esquemas locais do Amazon Redshift. Por padrão, todos os usuários têm permissões CREATE e USAGE no esquema PUBLIC.

Ao conceder USAGE a esquemas externos usando a sintaxe ON SCHEMA, você não precisa conceder ações separadamente nos objetos no esquema externo. As permissões de catálogo correspondentes controlam as permissões granulares nos objetos externos do esquema.

ON SCHEMA schema_name

Concede as permissões especificadas em um esquema.

GRANT CREATE ON SCHEMA e a permissão CREATE em GRANT ALL ON SCHEMA não são compatíveis com os esquemas externos do Amazon Redshift Spectrum. Para conceder o uso de tabelas externas em um esquema externo, conceda USAGE ON SCHEMA a usuários que precisam de acesso. Somente o proprietário de um esquema externo ou um superusuário tem permissão para criar tabelas externas no esquema externo. Para transferir a propriedade de um esquema externo, use ALTER SCHEMA para alterar o proprietário.

EXECUTE ON ALL FUNCTIONS IN SCHEMA schema_name

Concede as permissões especificadas em todas as funções no esquema de referência.

O Amazon Redshift não comporta instruções GRANT ou REVOKE para entradas pg_proc builtin definidas no namespace pg_catalog.

EXECUTE ON PROCEDURE procedure_name

Concede a permissão EXECUTE em um procedimento armazenado específico. Como os nomes de procedimento armazenado podem ser sobrecarregados, você deverá incluir a lista de argumentos para o procedimento. Para obter mais informações, consulte Nomeação de procedimentos armazenados.

EXECUTE ON ALL PROCEDURES IN SCHEMA schema_name

Concede as permissões especificadas em todos os procedimentos armazenados no esquema de referência.

USAGE ON LANGUAGE nome_linguagem

Concede a permissão USAGE em um idioma.

> **Note**
>
> A partir de 1.º de novembro de 2025, o Amazon Redshift não permitirá mais a criação de UDFs do Python. As UDFs do Python existentes continuarão a funcionar normalmente até 30 de junho de 2026. A partir de 1.º de julho de 2026, o Amazon Redshift não permitirá mais a criação de UDFs do Python. Recomendamos enfaticamente que você migre as UDFs existentes do Python para as UDFs do Lambda antes de 1.º de novembro de 2025. Para ter mais informações sobre como criar e usar UDFs do Lambda, consulte UDFs escalares do Lambda. Para ter informações sobre a conversão de UDFs do Python existentes em UDFs do Lambda, consulte a publicação de blog .

A permissão USAGE ON LANGUAGE é necessária para criar funções definidas pelo usuário (UDFs) executando o comando CREATE FUNCTION. Para obter mais informações, consulte Segurança e permissões de UDFs.

A permissão USAGE ON LANGUAGE é necessária para criar procedimentos armazenados executando o comando CREATE PROCEDURE. Para obter mais informações, consulte Segurança e privilégios para procedimentos armazenados.

Para UDFs Python, use `plpythonu`. Para UDFs SQL, use `sql`. Para procedimentos armazenados, use `plpgsql`.

ON COPY JOB job_name

Concede as permissões especificadas em um esquema.

FOR { ALL | COPY | UNLOAD | EXTERNAL FUNCTION | CREATE MODEL } [, ...]

Especifica o comando SQL para o qual a permissão é concedida. Você pode especificar ALL para conceder a permissão nas instruções COPY, UNLOAD, EXTERN FUNCTION e CREATE MODEL. Esta cláusula aplica-se apenas à concessão da permissão ASSUMEROLE.

ALTER

Concede a permissão ALTER aos usuários para adicionar ou remover objetos de uma unidade de compartilhamento de dados ou para definir a propriedade PUBLICACCESSIBLE. Para obter mais informações, consulte ALTER DATASHARE.

SHARE

Concede permissões a usuários e grupos de usuários para adicionar consumidores de dados a uma unidade de compartilhamento de dados. Essa permissão é necessária para permitir que o consumidor específico (conta ou namespace) acesse a unidade de compartilhamento de dados de seus clusters. O consumidor pode ser o mesmo ou de uma conta da AWS diferente, com o mesmo ou um namespace de cluster diferente, conforme especificado por um identificador globalmente exclusivo (GUID).

ON DATASHARE datashare_name

Concede as permissões especificadas na unidade de compartilhamento de dados de referência. Para obter informações sobre granularidade de controle de acesso de consumidor, consulte Compartilhamento de dados em diferentes níveis no Amazon Redshift.

USAGE

Quando USO é concedido a uma conta de consumidor ou namespace dentro da mesma conta, a conta de consumidor específica ou namespace dentro da conta pode acessar o datashare e os objetos do datashare de forma somente leitura.

TO NAMESPACE 'clusternamespace GUID'

Indica um namespace na mesma conta em que os consumidores podem receber as permissões especificadas para a unidade de compartilhamento de dados. Os namespaces usam um GUID alfanumérico de 128 bits.

TO ACCOUNT 'accountnumber' [VIA DATA CATALOG]

Indica o número de outra conta cujos consumidores podem receber as permissões especificadas para a unidade de compartilhamento de dados. Especificar “VIA DATA CATALOG” indica que você está concedendo o uso da unidade de compartilhamento de dados a uma conta do Lake Formation. Omitir esse parâmetro significa que você está concedendo uso a uma conta que possui o cluster.

ON DATABASE shared_database_name> [, ...]

Concede as permissões de uso especificadas no banco de dados especificado que é criado na unidade de compartilhamento de dados especificada.

ON SCHEMA shared_schema

Concede as permissões especificadas no esquema especificado que é criado na unidade de compartilhamento de dados especificada.

FOR { SCHEMAS | TABLES | FUNCTIONS | PROCEDURES | LANGUAGES | COPY JOBS} IN

Especifica os objetos de banco de dados aos quais conceder permissão. Os parâmetros após IN definem o escopo da permissão concedida.

CREATE MODEL

Concede a permissão CREATE MODEL a usuários ou grupos de usuários específicos.

ON MODEL model_name

Concede a permissão EXECUTE em um modelo específico.

ACCESS CATALOG

Concede a permissão para exibir metadados relevantes de objetos aos quais a função tem acesso.

{ role } [, ...]

A função a ser concedida a outra função, um usuário ou PUBLIC.

PUBLIC representa um grupo que inclui sempre todos os usuários. As permissões de um usuário individual consistem na soma das permissões concedidas a PUBLIC, das permissões concedidas a todos os grupos aos quais o usuário pertence e de quaisquer permissões concedidas ao usuário individualmente.

TO { { user_name [WITH ADMIN OPTION] } | role }[, ...]

Concede a função especificada a um usuário especificado com a WITH ADMIN OPTION, outra função ou PUBLIC.

A cláusula WITH ADMIN OPTION fornece as opções de administração para todas as funções concedidas a todos os beneficiários.

EXPLAIN { RLS | MASKING } TO ROLE rolename

Concede permissão para explicar os filtros de política de segurança de uma consulta no plano EXPLAIN a um perfil. RLS concede permissão para explicar filtros de política de segurança por linha. MASKING concede permissão para explicar os filtros da política de mascaramento dinâmico de dados.

IGNORE RLS TO ROLE rolename

Concede permissão para ignorar políticas de segurança por linha para uma consulta a um perfil.

TO { RLS | MASKING } POLICY policy_name

Indica a política de segurança que está recebendo as permissões. TO RLS POLICY indica uma política de segurança por linha. TO MASKING POLICY indica uma política de mascaramento dinâmico de dados.

Observações de uso

Para saber mais sobre as observações de uso de GRANT, consulte the section called “Observações de uso”.

Exemplos

Para conferir exemplos de como usar GRANT, consulte the section called “Exemplos”.

Observações de uso

Para conceder privilégios a um objeto, você deve atender a um dos seguintes critérios:

- Ser o proprietário do objeto.
- Ser um superusuário.
- Ter um privilégio concedido para o objeto e privilégio.

Por exemplo, o comando a seguir fornece ao usuário HR a capacidade de executar comandos SELECT na tabela de funcionários e conceder e revogar o mesmo privilégio para outros usuários.

```
grant select on table employees to HR with grant option;
```

HR não pode conceder privilégios a nenhuma operação além de SELECT ou a qualquer outra tabela além da de funcionários.

Por exemplo, o comando a seguir fornece ao usuário HR a capacidade de executar comandos ALTER na tabela de funcionários e conceder e revogar o mesmo privilégio para outros usuários.

```
grant ALTER on table employees to HR with grant option;
```

HR não pode conceder privilégios a nenhuma operação além de ALTER ou a qualquer outra tabela além da de funcionários.

Ter os privilégios concedidos em uma exibição não significa ter privilégios nas tabelas subjacentes. Da mesma forma, ter os privilégios concedidos em um esquema não significa ter privilégios nas tabelas do esquema. Em vez disso, conceda acesso às tabelas subjacentes explicitamente.

Para conceder privilégios para uma tabela do AWS Lake Formation, a função do IAM associada ao esquema externo da tabela deve ter permissão para conceder privilégios para a tabela externa. O exemplo a seguir cria um esquema externo com uma função do IAM associada `myGrantor`. A função do IAM `myGrantor` tem permissão para conceder permissões a outros. O comando GRANT usa a permissão da função do IAM `myGrantor` que está associada ao esquema externo para conceder permissão para a função do IAM `myGrantee`.

```
create external schema mySchema
from data catalog
database 'spectrum_db'
iam_role 'arn:aws:iam::123456789012:role/myGrantor'
create external database if not exists;
```

```
grant select
on external table mySchema.mytable
to iam_role 'arn:aws:iam::123456789012:role/myGrantee';
```

Se você conceder todos os privilégios a uma função do IAM, privilégios individuais serão concedidos no catálogo de dados habilitado para o Lake Formation relacionado. Por exemplo, o comando GRANT ALL a seguir resulta na exibição dos privilégios individuais concedidos (SELECT, ALTER, DROP, DELETE e INSERT) no console do Lake Formation.

```
grant all
on external table mySchema.mytable
to iam_role 'arn:aws:iam::123456789012:role/myGrantee';
```

Superusuários podem acessar todos os objetos, independentemente de comandos GRANT e REVOKE que definem privilégios de objeto.

Observações de uso para controle de acesso no nível da coluna

As observações de uso a seguir são aplicáveis a privilégios de nível de coluna em tabelas e visualizações do Amazon Redshift. Essas observações descrevem tabelas; as mesmas observações se aplicam às visualizações, a menos que uma exceção seja explicitamente indicada.

- Para uma tabela do Amazon Redshift, é possível conceder somente os privilégios SELECT e UPDATE em nível da coluna. Para uma exibição do Amazon Redshift, somente é possível conceder o privilégio SELECT em nível de coluna.
- A palavra-chave ALL é um sinônimo de privilégios SELECT e UPDATE combinados quando usada no contexto de um GRANT em nível de coluna em uma tabela.
- Se você não tiver o privilégio SELECT em todas as colunas de uma tabela, a execução de uma operação SELECT * retornará somente as colunas às quais você tem acesso. Ao usar uma visualização, uma operação SELECT * tenta acessar todas as colunas na visualização. Se você não tiver permissão para acessar todas as colunas, essas consultas apresentarão falha com um erro de permissão negada.
- SELECT * não se expande apenas a colunas acessíveis nos seguintes casos:
  - Não é possível criar uma visão normal com apenas colunas acessíveis usando SELECT *.
  - Não é possível criar uma visão materializada com apenas colunas acessíveis usando SELECT *.
- Se tiver o privilégio SELECT ou UPDATE em uma tabela ou exibição, e adicionar uma coluna, você ainda terá os mesmos privilégios na tabela ou exibição e, portanto, todas as colunas dela.
- Somente o proprietário de uma tabela ou um superusuário pode conceder privilégios em nível de coluna.
- A cláusula WITH GRANT OPTION não é compatível com privilégios de nível de coluna.
- Não é possível manter o mesmo privilégio em nível de tabela e em nível de coluna. Por exemplo, o usuário `data_scientist` não pode ter o privilégio SELECT na tabela `employee` e o privilégio SELECT na coluna `employee.department`. Ao conceder o mesmo privilégio a uma tabela e a uma coluna dentro da tabela, considere os seguintes resultados:
  - Se um usuário tiver um privilégio em nível de tabela em uma tabela, conceder o mesmo privilégio em nível de coluna não terá efeito.
  - Se um usuário tiver um privilégio em nível de tabela em uma tabela, a revogação do mesmo privilégio para uma ou mais colunas da tabela retornará um erro. Em vez disso, revogue o privilégio em nível de tabela.
  - Se um usuário tiver um privilégio em nível de coluna, conceder o mesmo privilégio em nível de tabela retornará um erro.
  - Se um usuário tiver um privilégio em nível de coluna, a revogação do mesmo privilégio no nível da tabela revoga os privilégios de coluna e tabela para todas as colunas da tabela.
- Não é possível conceder privilégios em nível de coluna em exibições de vinculação tardia.

- Você deve ter o privilégio SELECT no nível da tabela nas tabelas base para criar uma visão materializada. Mesmo que você tenha privilégios no nível da coluna para colunas específicas, não poderá criar uma visão materializada somente para essas colunas. No entanto, você poderá conceder o privilégio SELECT às colunas de uma visão materializada, semelhante às visualizações regulares.
- Para pesquisar concessões de privilégios de nível de coluna, use a exibição PG_ATTRIBUTE_INFO.

Observações de uso para conceder a permissão ASSUMEROLE

As observações de uso a seguir são aplicáveis à concessão da permissão ASSUMEROLE no Amazon Redshift.

Use a permissão ASSUMEROLE para controlar as permissões de acesso ao perfil do IAM para usuários, perfis e grupos de banco de dados em comandos como COPY, UNLOAD, EXTERNAL FUNCTION ou CREATE MODEL. Depois de conceder a permissão ASSUMEROLE a um usuário, perfil ou grupo para um perfil do IAM, o usuário, perfil ou grupo poderá assumir esse perfil ao executar o comando. A permissão ASSUMEROLE permite que você conceda acesso aos comandos apropriados conforme necessário.

Somente um superusuário de banco de dados pode conceder ou revogar a permissão ASSUMEROLE para usuários, perfis e grupos. Um superusuário sempre mantém a permissão ASSUMEROLE.

Para habilitar o uso da permissão ASSUMEROLE para usuários, perfis e grupos, um superusuário executa estas duas ações:

- Execute a seguinte instrução uma vez no cluster:

```
revoke assumerole on all from public for all;
```

- Conceda a permissão ASSUMEROLE aos usuários, perfis e grupos para os comandos apropriados.

Você pode especificar o encadeamento de perfis na cláusula ON ao conceder a permissão ASSUMEROLE. É usado vírgula para separar funções em uma cadeia de funções, por exemplo, `Role1,Role2,Role3`. Se o encadeamento de perfis tiver sido especificado ao conceder a permissão ASSUMEROLE, você deverá especificar a cadeia de perfis ao executar operações

concedidas pela permissão ASSUMEROLE. Não é possível especificar perfis individuais dentro da cadeia de perfis ao executar operações concedidas pela permissão ASSUMEROLE. Por exemplo, se um usuário, perfil ou grupo receber a cadeia de perfis `Role1,Role2,Role3`, não é possível especificar apenas `Role1` para executar operações.

Se um usuário tentar executar uma operação COPY, UNLOAD, EXTERNAL FUNCTION ou CREATE MODEL e não tiver recebido a permissão ASSUMEROLE, uma mensagem semelhante à seguinte será exibida.

```
ERROR:  User awsuser does not have ASSUMEROLE permission on IAM role
 "arn:aws:iam::123456789012:role/RoleA" for COPY
```

Para listar usuários que receberam acesso a perfis do IAM e comandos por meio da permissão ASSUMEROLE, consulte HAS_ASSUMEROLE_PRIVILEGE. Para listar os perfis do IAM e as permissões comandos que foram concedidos a um usuário especificado, consulte PG_GET_IAM_ROLE_BY_USER. Para listar usuários, perfis e grupos que receberam acesso a um perfil do IAM especificado por você, consulte PG_GET_GRANTEE_BY_IAM_ROLE.

Observações de uso para conceder permissões de machine learning

Você não pode conceder ou revogar diretamente as permissões relacionadas a uma função de ML. Uma função de ML pertence a um modelo de ML, e as permissões são controladas por meio do modelo. Em vez disso, você pode conceder permissões relacionadas ao modelo de ML. O exemplo a seguir demonstra como conceder permissões a todos os usuários para executar a função de ML associada ao modelo `customer_churn`.

```
GRANT EXECUTE ON MODEL customer_churn TO PUBLIC;
```

Você também pode conceder todas as permissões a um usuário para o modelo de ML `customer_churn`.

```
GRANT ALL on MODEL customer_churn TO ml_user;
```

A concessão da permissão `EXECUTE` relacionada a uma função de ML falhará se houver uma função de ML no esquema, mesmo que essa função de ML já tenha a permissão `EXECUTE` por meio de `GRANT EXECUTE ON MODEL`. Recomendamos usar um esquema separado ao usar o comando `CREATE MODEL` para manter as funções de ML em um esquema separado. O exemplo a seguir demonstra como fazer isso.

```
CREATE MODEL ml_schema.customer_churn
FROM customer_data
TARGET churn
FUNCTION ml_schema.customer_churn_prediction
IAM_ROLE default
SETTINGS (
 S3_BUCKET 'amzn-s3-demo-bucket'
);
```

Observações de uso para conceder a permissão DEBUG

A permissão DEBUG se aplica a bancos de dados nos warehouses do Amazon Redshift com permissões federadas que usam controle de acesso refinado (FGAC). Por padrão, quando uma consulta acessa dados protegidos por FGAC nesse banco de dados, o Amazon Redshift edita valores confidenciais nos registros de log do sistema que são gerados para essa consulta. Isso protege os dados do produtor do consumidor. A concessão de DEBUG permite que entidades especificadas vejam esses registros de log sem supressão. Para ver a lista de campos suprimidos e acessar mais informações sobre registro em log seguro, consulte Registro em log seguro no Guia do desenvolvedor do Amazon Redshift.

Somente um superusuário ou o proprietário do banco de dados pode conceder ou revogar a permissão DEBUG. Você pode conceder a permissão DEBUG somente em um banco de dados em um warehouse do Amazon Redshift com permissões federadas, não em um banco de dados local ou em um banco de dados criado a partir de um AWS Lake Formation ou um catálogo do AWS Glue Data Catalog.

A permissão DEBUG aceita as seguintes formas:

- Concessão a uma identidade global. Use `TO` *`<prefix>:<name>`*, em que o prefixo identifica o provedor de identidade (por exemplo, `IAM` para um usuário do IAM, `IAMR` para um perfil do IAM ou o prefixo do namespace IdC (Centro de Identidade do AWS IAM) para um usuário do IdC). Somente a identidade especificada pode ver os registros não suprimidos que foram gerados por suas próprias consultas. Nenhum outro usuário, incluindo um superusuário, pode vê-los.
- Concessão a um grupo do IdC. Use `TO ROLE` *`<idc_prefix>:<group_name>`*, em que o prefixo identifica a instância do IdC (Centro de Identidade do AWS IAM). Todos os membros podem ver os registros não suprimidos que foram gerados por suas próprias consultas.
- Concessão aos administradores da conta. Use `TO ADMINS OF ACCOUNT '`*`account_id`*`'`. Todos os superusuários (no momento do acesso ao log) na conta de consumidor especificada

podem ver os registros não suprimidos. Os registros autorizados dessa forma também não são suprimidos quando o Amazon Redshift exporta dados da tabela do sistema para as Tabelas do Amazon S3. Você pode especificar somente a mesma conta proprietária do produtor.

As seguintes observações de uso são aplicáveis à permissão DEBUG:

- A permissão DEBUG não está incluída em GRANT ALL ON DATABASE. Você deve concedê-la explicitamente.
- Você pode conceder DEBUG em apenas um banco de dados por instrução e não pode combiná-la com outras permissões na mesma instrução.
- Você não pode conceder DEBUG a PUBLIC, a um usuário local ou a um grupo local.
- A permissão DEBUG é apenas prospectiva. A concessão de DEBUG afeta somente os registros de log gerados após a concessão. A revogação da permissão DEBUG afeta somente os registros gerados após a revogação; os registros gerados enquanto a permissão DEBUG estava ativa permanecem visíveis durante toda a vida útil do log.
- Se você cancelar o registro do cluster ou do namespace de AWS Glue Data Catalog, as concessões DEBUG existentes serão mantidas, mas não terão efeito, pois o warehouse não usa mais permissões federadas. Não é necessário revogá-las. Se você registrar o cluster ou o namespace novamente, as concessões entrarão em vigor novamente. As concessões DEBUG são removidas somente quando o banco de dados é descartado. Para obter mais informações, consulte Cancelar o registro de AWS Glue Data Catalog.
- Quando uma consulta se une a vários bancos de dados com permissões federadas, você deve conceder a permissão DEBUG em cada banco de dados que contribua com dados protegidos pelo FGAC para a consulta. Caso contrário, os registros permanecerão suprimidos.
- Quando você concede a permissão DEBUG a um perfil do IAM, um superusuário local que assumir esse perfil (`SET SESSION AUTHORIZATION`) usando ALTER USER SET GLOBAL IDENTITY poderá ver os registros não suprimidos do perfil. Usuários do IAM e usuários do IdC não podem ser representados dessa forma.

Exemplos

O exemplo a seguir concede o privilégio SELECT na tabela SALES ao usuário `fred`.

```
grant select on table sales to fred;
```

O exemplo a seguir concede o privilégio SELECT em todas as tabelas no esquema QA_TICKIT ao usuário `fred`.

```
grant select on all tables in schema qa_tickit to fred;
```

O exemplo a seguir concede todos os privilégios de esquema no esquema QA_TICKIT ao grupo de usuários QA_USERS. Os privilégios de esquema são CREATE e USAGE. USAGE permite que os usuários acessem os objetos no esquema, mas não concede privilégios como INSERT ou SELECT em tais objetos. Conceder privilégios em cada objeto separadamente.

```
create group qa_users;
grant all on schema qa_tickit to group qa_users;
```

O exemplo a seguir concede todos os privilégios na tabela SALES no esquema QA_TICKIT a todos os usuários do grupo QA_USERS.

```
grant all on table qa_tickit.sales to group qa_users;
```

O exemplo a seguir concede todos os privilégios na tabela SALES no esquema QA_TICKIT a todos os usuários dos grupos QA_USERS e RO_USERS.

```
grant all on table qa_tickit.sales to group qa_users, group ro_users;
```

O exemplo a seguir concede o privilégio DROP na tabela SALES no esquema QA_TICKIT a todos os usuários no grupo QA_USERS.

```
grant drop on table qa_tickit.sales to group qa_users;>
```

A sequência de comandos a seguir mostra como o acesso a um esquema não concede privilégios a uma tabela no esquema.

```
create user schema_user in group qa_users password 'Abcd1234';
create schema qa_tickit;
create table qa_tickit.test (col1 int);
grant all on schema qa_tickit to schema_user;

set session authorization schema_user;
select current_user;
```

```
current_user
--------------
schema_user
(1 row)


select count(*) from qa_tickit.test;


ERROR: permission denied for relation test [SQL State=42501]


set session authorization dw_user;
grant select on table qa_tickit.test to schema_user;
set session authorization schema_user;
select count(*) from qa_tickit.test;


count
-------
0
(1 row)
```

A sequência de comandos a seguir mostra como o acesso a uma exibição não implica acesso às suas tabelas subjacentes. O usuário chamado VIEW_USER não pode selecionar da tabela DATE, embora todos os privilégios de VIEW_DATE tenham sido concedidos a ele.

```
create user view_user password 'Abcd1234';
create view view_date as select * from date;
grant all on view_date to view_user;
set session authorization view_user;
select current_user;


current_user
--------------
view_user
(1 row)


select count(*) from view_date;
```

```
count
-------
365
(1 row)


select count(*) from date;


ERROR:  permission denied for relation date
```

O exemplo a seguir concede o privilégio SELECT nas colunas `cust_name` e `cust_phone` da tabela `cust_profile` ao usuário `user1`.

```
grant select(cust_name, cust_phone) on cust_profile to user1;
```

O exemplo a seguir concede o privilégio SELECT nas colunas `cust_name` e `cust_phone`, e o privilégio UPDATE na coluna `cust_contact_preference` da tabela `cust_profile` ao grupo `sales_group`.

```
grant select(cust_name, cust_phone), update(cust_contact_preference) on cust_profile to
 group sales_group;
```

O exemplo a seguir mostra o uso da palavra-chave ALL para conceder privilégios SELECT e UPDATE em três colunas da tabela `cust_profile` ao grupo `sales_admin`.

```
grant ALL(cust_name, cust_phone,cust_contact_preference) on cust_profile to group
 sales_admin;
```

O exemplo a seguir concede o privilégio SELECT na coluna `cust_name` da exibição `cust_profile_vw` ao usuário `user2`.

```
grant select(cust_name) on cust_profile_vw to user2;
```

Exemplos da concessão de acesso a unidades de compartilhamento de dados

Os exemplos a seguir mostram permissões de uso de datashare GRANT em um banco de dados específico ou esquema criado a partir de um datashare.

No exemplo a seguir, um administrador no lado do produtor concede a permissão USAGE na unidade de compartilhamento de dados `salesshare` ao namespace especificado.

```
GRANT USAGE ON DATASHARE salesshare TO NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

No exemplo a seguir, um administrador no lado do consumidor concede a permissão USAGE no `sales_db` a `Bob`.

```
GRANT USAGE ON DATABASE sales_db TO Bob;
```

No exemplo a seguir, um administrador no lado do consumidor concede a permissão GRANT USAGE no esquema `sales_schema` à função `Analyst_role`. `sales_schema` é um esquema externo que aponta para sales_db.

```
GRANT USAGE ON SCHEMA sales_schema TO ROLE Analyst_role;
```

Neste ponto, `Bob` e `Analyst_role` podem ter acesso a todos os objetos do banco de dados em `sales_schema` e `sales_db`.

O exemplo a seguir mostra a concessão de permissão adicional no nível do objeto para objetos em um banco de dados compartilhado. Essas permissões extras só serão necessárias se o comando CREATE DATABASE usado para criar o banco de dados compartilhado usar a cláusula WITH PERMISSIONS. Se o comando CREATE DATABASE não usou WITH PERMISSIONS, a concessão de USAGE no banco de dados compartilhado concede acesso total a todos os objetos nesse banco de dados.

```
GRANT SELECT ON sales_db.sales_schema.tickit_sales_redshift to Bob;
```

Exemplos de concessão de permissões em escopo definido

O exemplo a seguir concede uso para todos os esquemas atuais e futuros no banco de dados `Sales_db` à função `Sales`.

```
GRANT USAGE FOR SCHEMAS IN DATABASE Sales_db TO ROLE Sales;
```

O exemplo a seguir concede a permissão SELECT para todas as tabelas atuais e futuras no banco de dados `Sales_db` ao usuário `alice` e também dá a `alice` a permissão para conceder permissões em escopo em tabelas em `Sales_db` a outros usuários.

```
GRANT SELECT FOR TABLES IN DATABASE Sales_db TO alice WITH GRANT OPTION;
```

O exemplo a seguir concede a permissão EXECUTE para funções no esquema `Sales_schema` ao usuário bob.

```
GRANT EXECUTE FOR FUNCTIONS IN SCHEMA Sales_schema TO bob;
```

O exemplo a seguir concede todas as permissões para todas as tabelas no esquema do `ShareSchema` do banco de dados `ShareDb` à função `Sales`. Ao especificar o esquema, você pode especificar o banco de dados do esquema usando o formato de duas partes `database.schema`.

```
GRANT ALL FOR TABLES IN SCHEMA ShareDb.ShareSchema TO ROLE Sales;
```

O exemplo a seguir é o mesmo do anterior. Você pode especificar o banco de dados usando a palavra-chave `DATABASE`, em vez de usar um formato de duas partes.

```
GRANT ALL FOR TABLES IN SCHEMA ShareSchema DATABASE ShareDb TO ROLE Sales;
```

Exemplos de concessão do privilégio ASSUMEROLE

Veja a seguir exemplos de concessão do privilégio ASSUMEROLE.

O exemplo a seguir mostra a instrução REVOKE que um superusuário executa uma vez no cluster para habilitar o uso do privilégio ASSUMEROLE para usuários e grupos. Em seguida, o superusuário concede o privilégio ASSUMEROLE aos usuários e grupos para os comandos apropriados. Para obter informações sobre como habilitar o uso do privilégio ASSUMEROLE para usuários e grupos, consulte Observações de uso para conceder a permissão ASSUMEROLE.

```
revoke assumerole on all from public for all;
```

O exemplo a seguir concede o privilégio ASSUMEROLE ao usuário `reg_user1` para a função do IAM `Redshift-S3-Read` para executar operações COPY.

```
grant assumerole on 'arn:aws:iam::123456789012:role/Redshift-S3-Read'
to reg_user1 for copy;
```

O exemplo a seguir concede o privilégio ASSUMEROLE ao usuário `reg_user1` para a cadeia de função do IAM `RoleA` e `RoleB` para executar operações COPY.

```
grant assumerole
on 'arn:aws:iam::123456789012:role/RoleA,arn:aws:iam::210987654321:role/RoleB'
to reg_user1
for unload;
```

Veja a seguir um exemplo do comando UNLOAD usando a cadeia de função do IAM `RoleA` e `RoleB`.

```
unload ('select * from venue limit 10')
to 's3://companyb/redshift/venue_pipe_'
iam_role 'arn:aws:iam::123456789012:role/RoleA,arn:aws:iam::210987654321:role/RoleB';
```

O exemplo a seguir concede o privilégio ASSUMEROLE ao usuário `reg_user1` para a função do IAM `Redshift-Exfunc` para criar funções externas.

```
grant assumerole on 'arn:aws:iam::123456789012:role/Redshift-Exfunc'
to reg_user1 for external function;
```

O exemplo a seguir concede o privilégio ASSUMEROLE ao usuário `reg_user1` para a função do IAM `Redshift-model` para criar modelos de Machine Learning.

```
grant assumerole on 'arn:aws:iam::123456789012:role/Redshift-ML'
to reg_user1 for create model;
```

Exemplos de concessão do privilégio ROLE

O exemplo a seguir concede a função sample_role1 ao usuário user1.

```
CREATE ROLE sample_role1;
GRANT ROLE sample_role1 TO user1;
```

O exemplo a seguir concede sample_role1 ao user1 com a WITH ADMIN OPTION, define a sessão atual para user1 e user1 concede sample_role1 ao user2.

```
GRANT ROLE sample_role1 TO user1 WITH ADMIN OPTION;
SET SESSION AUTHORIZATION user1;
GRANT ROLE sample_role1 TO user2;
```

O exemplo a seguir concede a função sample_role1 a sample_role2.

```
GRANT ROLE sample_role1 TO ROLE sample_role2;
```

O exemplo a seguir concede a função sample_role2 a sample_role3 e sample_role4. Em seguida, ele tenta conceder sample_role3 a sample_role1.

```
GRANT ROLE sample_role2 TO ROLE sample_role3;
GRANT ROLE sample_role3 TO ROLE sample_role2;
ERROR: cannot grant this role, a circular dependency was detected between these roles
```

O exemplo a seguir concede privilégios de sistema CREATE USER a sample_role1.

```
GRANT CREATE USER TO ROLE sample_role1;
```

O exemplo a seguir concede a função definida pelo sistema `sys:dba` a user1.

```
GRANT ROLE sys:dba TO user1;
```

O exemplo a seguir tenta conceder sample_role3 em uma dependência circular a sample_role2.

```
CREATE ROLE sample_role3;
GRANT ROLE sample_role2 TO ROLE sample_role3;
GRANT ROLE sample_role3 TO ROLE sample_role2; -- fail
ERROR:  cannot grant this role, a circular dependency was detected between these roles
```

INSERT

Tópicos

- Sintaxe
- Parâmetros
- Observações de uso
- Exemplos de INSERT

Insere novas linhas em uma tabela, Você pode inserir uma única linha com a sintaxe VALUES, várias linhas com a sintaxe VALUES, ou uma ou mais linhas definidas pelos resultados de uma consulta (INSERT INTO...SELECT).

> **Note**
>
> Recomendamos veementemente o uso do comando COPY para carregar grandes quantidades de dados. O uso de instruções INSERT individuais para povoar uma tabela pode ser proibitivamente lento. Como alternativa, se seus dados já existirem em outras tabelas de banco de dados do Amazon Redshift, use INSERT INTO SELECT ou CREATE TABLE AS para melhorar a performance. Para obter mais informações sobre o uso do comando COPY para carregar tabelas, consulte Carregar dados no Amazon Redshift.

> **Note**
>
> O tamanho máximo de uma única instrução SQL é 16 MB.

Sintaxe

```
INSERT INTO table_name [ ( column [, ...] ) ]
{DEFAULT VALUES |
VALUES ( { expression | DEFAULT } [, ...] )
[, ( { expression | DEFAULT } [, ...] )
[, ...] ] |
query }
```

Parâmetros

table_name

Uma tabela temporária ou persistente. Somente o proprietário da tabela ou um usuário com o privilégio INSERT na tabela pode inserir linhas. Se usar a cláusula de consulta para inserir linhas, você deve ter o privilégio SELECT nas tabelas mencionadas na consulta.

> **Note**
>
> Use INSERT (tabela externa) para inserir resultados de uma consulta SELECT em tabelas existentes no catálogo externo. Para obter mais informações, consulte INSERT (tabela externa).

column

Você pode inserir valores em uma ou mais colunas da tabela. Você pode listar os nomes de colunas de destino em qualquer ordem. Se você não especificar uma lista de colunas, os valores a serem inseridos deverão corresponder a colunas de tabela na ordem em que foram declaradas na instrução CREATE TABLE. Se o número de valores a serem inseridos for menor do que o número de colunas na tabela, as primeiras n colunas serão carregadas.

O valor padrão declarado ou um valor nulo é carregado em qualquer coluna que não esteja listada (implícita ou explicitamente) na instrução INSERT.

DEFAULT VALUES

Se, durante a criação da tabela, tiverem sido atribuídos valores padrão às colunas na tabela, use essas palavras-chave para inserir uma linha que consista totalmente em valores padrão. Se alguma coluna não tiver valores padrão, valores nulos serão inseridos nela. Se qualquer uma das colunas for declarada NOT NULL, a instrução INSERT retornará um erro.

VALUES

Use esta palavra-chave para inserir uma ou mais linhas, cada linha consistindo em um ou mais valores. A lista VALUES de cada linha deve estar alinhada com a lista de colunas. Para inserir várias linhas, use um delimitador de vírgula entre cada lista de expressões. Não repita a palavra-chave VALUES. Todas as listas VALUES para uma instrução INSERT de várias linhas devem conter o mesmo número de valores.

expression

Um único valor ou uma expressão que avalia um único valor. Cada valor deve ser compatível com o tipo de dados da coluna em que está sendo inserido. Se possível, um valor cujo tipo de dados não corresponde ao tipo de dados declarado da coluna será convertido automaticamente em um tipo de dados compatível. Por exemplo:

- Um valor decimal `1.1` é inserido em uma coluna INT como `1`.
- Um valor decimal `100.8976` é inserido em uma coluna DEC(5,2) como `100.90`.

Você pode converter explicitamente um valor em um tipo de dados compatível incluindo o tipo de sintaxe cast na expressão. Por exemplo, se a coluna COL1 na tabela T1 for uma coluna CHAR(3):

```
insert into t1(col1) values('Incomplete'::char(3));
```

Esta instrução insere o valor `Inc` na coluna.

Para uma instrução INSERT VALUES de linha única, você pode usar uma subconsulta escalar como uma expressão. O resultado da subconsulta é inserido na coluna apropriada.

> **Note**
>
> Subconsultas não são compatíveis como expressões para instruções INSERT VALUES de várias linhas.

DEFAULT

Use esta palavra-chave para inserir o valor padrão de uma coluna, como definido quando a tabela foi criada. Se não existir um valor padrão para uma coluna, um valor "null" será inserido. Você não poderá inserir um valor padrão em uma coluna com restrição NOT NULL se a coluna não tiver um valor padrão explícito atribuído a ela na instrução CREATE TABLE.

query

Insira uma ou mais linhas na tabela definindo qualquer consulta. Todas as linhas que a consulta produz são inseridas na tabela. A consulta deve retornar uma lista de colunas compatíveis com as colunas na tabela, mas os nomes das colunas não precisam obrigatoriamente corresponder.

Observações de uso

> **Note**
>
> Recomendamos veementemente o uso do comando COPY para carregar grandes quantidades de dados. O uso de instruções INSERT individuais para povoar uma tabela pode ser proibitivamente lento. Como alternativa, se seus dados já existirem em outras tabelas de banco de dados do Amazon Redshift, use INSERT INTO SELECT ou CREATE TABLE AS para melhorar a performance. Para obter mais informações sobre o uso do comando COPY para carregar tabelas, consulte Carregar dados no Amazon Redshift.

O formato de dados dos valores inseridos deve corresponder ao formato de dados especificado pela definição de CREATE TABLE.

Depois de inserir um grande número de linhas novas em uma tabela:

- Limpe a tabela para recuperar espaço e reclassificar as linhas.
- Analise a tabela para atualizar as estatísticas do planejador de consulta.

Quando os valores são inseridos em colunas DECIMAIS e ultrapassam a escala especificada, os valores carregados são arredondados para cima, conforme apropriado. Por exemplo, quando um valor `20.259` é inserido em uma coluna DECIMAL(8,2), o valor armazenado é `20.26`.

É possível inserir uma coluna GENERATED BY DEFAULT AS IDENTITY. Você pode atualizar as colunas definidas como GENERATED BY DEFAULT AS IDENTITY com os valores fornecidos. Para obter mais informações, consulte GENERATED BY DEFAULT AS IDENTITY.

Exemplos de INSERT

A tabela CATEGORY no banco de dados TICKIT contém as seguintes linhas:

```
 catid | catgroup |  catname  |                  catdesc
-------+----------+-----------+--------------------------------------------
     1 | Sports   | MLB       | Major League Baseball
     2 | Sports   | NHL       | National Hockey League
     3 | Sports   | NFL       | National Football League
     4 | Sports   | NBA       | National Basketball Association
     5 | Sports   | MLS       | Major League Soccer
     6 | Shows    | Musicals  | Musical theatre
     7 | Shows    | Plays     | All non-musical theatre
     8 | Shows    | Opera     | All opera and light opera
     9 | Concerts | Pop       | All rock and pop music concerts
    10 | Concerts | Jazz      | All jazz singers and bands
    11 | Concerts | Classical | All symphony, concerto, and choir concerts
(11 rows)
```

Crie uma tabela CATEGORY_STAGE com um esquema semelhante ao da tabela CATEGORY, mas defina os valores padrão para as colunas:

```
create table category_stage
(catid smallint default 0,
catgroup varchar(10) default 'General',
catname varchar(10) default 'General',
catdesc varchar(50) default 'General');
```

A instrução INSERT a seguir seleciona todas as linhas da tabela CATEGORY e as insere na tabela CATEGORY_STAGE.

```
insert into category_stage
(select * from category);
```

Os parênteses em volta da consulta são opcionais.

Este comando insere uma nova linha na tabela CATEGORY_STAGE com um valor especificado para cada coluna em ordem:

```
insert into category_stage values
(12, 'Concerts', 'Comedy', 'All stand-up comedy performances');
```

Você também pode inserir uma nova linha que combina valores específicos e valores padrão:

```
insert into category_stage values
(13, 'Concerts', 'Other', default);
```

Execute a consulta a seguir para obter as linhas inseridas:

```
select * from category_stage
where catid in(12,13) order by 1;

 catid | catgroup | catname |             catdesc
-------+----------+---------+----------------------------------
    12 | Concerts | Comedy  | All stand-up comedy performances
    13 | Concerts | Other   | General
(2 rows)
```

Os exemplos a seguir mostram algumas instruções INSERT VALUES de várias linhas. O primeiro exemplo insere valores CATID específicos para duas linhas e valores padrão para outras colunas em ambas as linhas.

```
insert into category_stage values
(14, default, default, default),
(15, default, default, default);

select * from category_stage where catid in(14,15) order by 1;
 catid | catgroup | catname | catdesc
-------+----------+---------+---------
    14 | General  | General | General
    15 | General  | General | General
```

```
(2 rows)
```

O próximo exemplo insere três linhas com várias combinações de valores específicos e valores padrão:

```
insert into category_stage values
(default, default, default, default),
(20, default, 'Country', default),
(21, 'Concerts', 'Rock', default);

select * from category_stage where catid in(0,20,21) order by 1;
 catid | catgroup | catname | catdesc
-------+----------+---------+---------
     0 | General  | General | General
    20 | General  | Country | General
    21 | Concerts | Rock    | General
(3 rows)
```

Neste exemplo, o primeiro conjunto de VALUES produz os mesmos resultados que especificar DEFAULT VALUES para uma instrução INSERT de linha única.

Os exemplos a seguir mostram o comportamento de INSERT quando uma tabela tem uma coluna IDENTITY. Primeiro, crie uma nova versão de tabela CATEGORY, depois insira linhas de CATEGORY nela:

```
create table category_ident
(catid int identity not null,
catgroup varchar(10) default 'General',
catname varchar(10) default 'General',
catdesc varchar(50) default 'General');


insert into category_ident(catgroup,catname,catdesc)
select catgroup,catname,catdesc from category;
```

Observe que você não pode inserir valores inteiros específicos na coluna CATID IDENTITY. Os valores da coluna IDENTITY são gerados automaticamente.

O exemplo a seguir demonstra que subconsultas não podem ser usadas como expressões em instruções INSERT VALUES de várias linhas:

```
insert into category(catid) values
```

```
((select max(catid)+1 from category)),
((select max(catid)+2 from category));

ERROR: can't use subqueries in multi-row VALUES
```

O exemplo a seguir mostra uma inserção em uma tabela temporária preenchida com dados da tabela venue usando a cláusula WITH SELECT. Para obter mais informações sobre a tabela venue, consulte Banco de dados de exemplo.

Primeiro, crie a tabela temporária #venuetemp.

```
CREATE TABLE #venuetemp AS SELECT * FROM venue;
```

Liste as linhas na tabela #venuetemp.

```
SELECT * FROM #venuetemp ORDER BY venueid;

venueid | venuename                | venuecity  | venuestate| venueseats
--------+--------------------------+------------+-----------+------------
1       Toyota Park                Bridgeview   IL          0
2       Columbus Crew Stadium      Columbus     OH          0
3       RFK Stadium                Washington   DC          0
4       CommunityAmerica Ballpark  Kansas City  KS          0
5       Gillette Stadium           Foxborough   MA          68756
...
```

Insira dez linhas duplicadas na tabela #venuetemp usando a cláusula WITH SELECT.

```
INSERT INTO #venuetemp (WITH venuecopy AS (SELECT * FROM venue) SELECT * FROM venuecopy
 ORDER BY 1 LIMIT 10);
```

Liste as linhas na tabela #venuetemp.

```
SELECT * FROM #venuetemp ORDER BY venueid;

venueid | venuename                | venuecity  | venuestate| venueseats
--------+--------------------------+------------+-----------+------------
1       Toyota Park                Bridgeview   IL          0
1       Toyota Park                Bridgeview   IL          0
2       Columbus Crew Stadium      Columbus     OH          0
2       Columbus Crew Stadium      Columbus     OH          0
```

```
3       RFK Stadium               Washington   DC          0
3       RFK Stadium               Washington   DC          0
4       CommunityAmerica Ballpark  Kansas City  KS          0
4       CommunityAmerica Ballpark  Kansas City  KS          0
5       Gillette Stadium          Foxborough   MA          68756
5       Gillette Stadium          Foxborough   MA          68756
...
```

INSERT (tabela externa)

Insere o resultado de uma consulta SELECT em tabelas externas ou catálogos externos existentes, como o AWS Glue, o AWS Lake Formation ou um metastore do Apache Hive. Use a mesma função do AWS Identity and Access Management (IAM) usada para o comando CREATE EXTERNAL SCHEMA para interagir com catálogos externos e com o Amazon S3.

Para tabelas não particionadas, o comando INSERT (tabela externa) grava dados no local do Amazon S3 definido na tabela, com base nas propriedades de tabela e no formato de arquivo especificados.

Para tabelas particionadas, INSERT (tabela externa) grava dados no local do Amazon S3 de acordo com a chave de partição especificada na tabela. Ele também registra novas partições no catálogo externo após a operação INSERT ser concluída.

Não é possível executar INSERT (tabela externa) em um bloco de transação (BEGIN ... END). Para obter mais informações sobre transações, consulte Níveis de isolamento no Amazon Redshift.

Sintaxe

```
INSERT INTO external_schema.table_name
{ select_statement }
```

Parâmetros

external_schema.table_name

O nome de um esquema externo existente e uma tabela externa de destino na qual será inserido.

select_statement

Uma instrução que insere uma ou mais linhas na tabela externa ao definir qualquer consulta. Todas as linhas que a consulta gera são gravadas no Amazon S3 no formato de texto ou Parquet com base na definição da tabela. A consulta deve retornar uma lista de colunas que

seja compatível com os tipos de dados da coluna na tabela externa. No entanto, os nomes das colunas não precisam corresponder.

Observações de uso

O número de colunas na consulta SELECT deve ser igual à soma de colunas de dados e de colunas de partição. O local e o tipo de dados de cada coluna de dados deve corresponder àqueles da tabela externa. O local das colunas de partição devem estar no final da consulta SELECT, na mesma ordem definida no comando CREATE EXTERNAL TABLE. Os nomes das colunas não precisam corresponder.

Em alguns casos, talvez você queira executar o comando INSERT (tabela externa) em um catálogo de dados do AWS Glue ou um metastore do Hive. No caso do AWS Glue, a função do IAM usada para criar o esquema externo deve ter as permissões de leitura e gravação no Amazon S3 e no AWS Glue. Se você usar um catálogo do AWS Lake Formation, essa função do IAM se tornará a proprietária da nova tabela do Lake Formation. Essa função do IAM deve ter pelo menos as seguintes permissões:

- Permissão SELECT, INSERT, UPDATE na tabela externa
- Permissão do local dos dados no caminho do Amazon S3 da tabela externa

Para garantir que os nomes dos arquivos sejam exclusivos, o Amazon Redshift usa, por padrão, o formato a seguir para o nome de cada arquivo carregado no Amazon S3.

*<date>*_*<time>*_*<microseconds>*_*<query_id>*_*<slice-number>*_part_*<part-number>*.*<format>*.

Um exemplo é `20200303_004509_810669_1007_0001_part_00.parquet`.

Considere o seguinte ao executar o comando INSERT (tabela externa):

- Não há suporte para tabelas externas com formato diferente de PARQUET ou TEXTFILE.
- Este comando oferece suporte às propriedades de tabela existentes, como 'write.parallel', 'write.maxfilesize.mb', 'compression_type' e 'serialization.null.format'. Para atualizar esses valores, execute o comando ALTER TABLE SET TABLE PROPERTIES.
- A propriedade de tabela 'numRows' é automaticamente atualizada no final da operação INSERT. A propriedade da tabela deve estar definida ou ter sido adicionada à tabela, caso ela não tenha sido criada pela operação CREATE EXTERNAL TABLE AS.

- Não há suporte para a cláusula LIMIT na consulta SELECT externa. Em vez disso, use uma cláusula LIMIT aninhada.
- É possível usar a tabela STL_UNLOAD_LOG para rastrear os arquivos que foram gravados no Amazon S3 por cada operação INSERT (tabela externa).
- O Amazon Redshift suporta somente a criptografia padrão do Amazon S3 para INSERT (tabela externa).

Exemplos de INSERT (tabela externa)

O exemplo a seguir insere os resultados da instrução SELECT na tabela externa.

```
INSERT INTO spectrum.lineitem
SELECT * FROM local_lineitem;
```

O exemplo a seguir insere os resultados da instrução SELECT em uma tabela externa particionada usando o particionamento estático. As colunas de partição são codificadas na instrução SELECT. As colunas de partição devem estar no final da consulta.

```
INSERT INTO spectrum.customer
SELECT name, age, gender, 'May', 28 FROM local_customer;
```

O exemplo a seguir insere os resultados da instrução SELECT em uma tabela externa particionada usando o particionamento dinâmico. As colunas de partição não são codificadas. Os dados são adicionados automaticamente às pastas de partição existentes ou a novas pastas se uma nova partição for adicionada.

```
INSERT INTO spectrum.customer
SELECT name, age, gender, month, day FROM local_customer;
```

LOCK

Restringe acesso a uma tabela de banco de dados. Este comando só é significativo quando executado em um bloco de transação.

O comando LOCK gera um bloqueio de tabela no modo "ACCESS EXCLUSIVE", aguardando, caso necessário, qualquer conflito de bloqueio ser liberado. Com esse tipo de bloqueio explícito, as leituras e gravações na tabela esperam ao receber tentativas de outras transações e sessões.

Um bloqueio de tabela explícito criado por um usuário impede temporariamente que outro usuário selecione dados dessa tabela ou carregue dados nela. O bloqueio é liberado quando a transação que contém o comando LOCK é concluída.

Os bloqueios menos restritivos de tabela são adquiridos implicitamente por comandos que mencionam as tabelas, como operações de gravação. Por exemplo, se um usuário tentar ler os dados de uma tabela enquanto outro usuário atualiza a tabela, os dados lidos serão um snapshot dos dados que já estiverem confirmados. (Em alguns casos, as consultas serão interrompidas se violarem regras de isolamento serializável.) Consulte Gerenciamento de operações de gravação simultâneas.

Algumas operações DDL, como DROP TABLE e TRUNCATE, criam bloqueios exclusivos. Essas operações impedem a leitura de dados.

Se ocorrer um conflito de bloqueio, o Amazon Redshift exibirá uma mensagem de erro para alertar o usuário que começou a transação conflitante. A transação que recebeu o conflito de bloqueio é interrompida. Toda vez que um conflito de bloqueio ocorre, o Amazon Redshift grava uma entrada na tabela STL_TR_CONFLICT.

Sintaxe

```
LOCK [ TABLE ] table_name [, ...]
```

Parâmetros

TABLE

Palavra-chave opcional.

table_name

Nome da tabela a ser bloqueada. Você pode bloquear mais de uma tabela usando uma lista de nomes de tabelas separados por vírgulas. Você não pode bloquear visualizações.

Exemplo

```
begin;

lock event, sales;
```

```
...
```

MERGE

Mescla condicionalmente as linhas de uma tabela de origem em uma tabela de destino. Tradicionalmente, isso só pode ser feito usando várias instruções insert, update ou delete separadamente. Para obter mais informações sobre as operações que podem ser combinadas com MERGE, consulte UPDATE, DELETE e INSERT.

Sintaxe

```
MERGE INTO target_table
USING source_table [ [ AS ] alias ]
ON match_condition
[ WHEN MATCHED THEN { UPDATE SET col_name = { expr } [,...] | DELETE }
WHEN NOT MATCHED THEN INSERT [ ( col_name [,...] ) ] VALUES ( { expr } [, ...] ) |
REMOVE DUPLICATES ]
```

Parâmetros

target_table

A tabela temporária ou permanente de destino da instrução MERGE.

source_table

A tabela temporária ou permanente que fornece as linhas a serem mescladas em target_table. source_table também pode ser uma tabela do Spectrum.

alias

O nome alternativo temporário para source_table.

Esse parâmetro é opcional. O alias precedente com AS também é opcional.

match_condition

Especifica predicados iguais entre a coluna da tabela de origem e a coluna da tabela de destino que são usados para determinar se as linhas em source_table podem ser combinadas com as linhas em target_table. Se a condição for atendida, MERGE executará matched_clause para essa linha. Caso contrário, MERGE executará not_matched_clause para essa linha.

WHEN MATCHED

Especifica a ação a ser executada quando a condição de correspondência entre uma linha de origem e uma linha de destino é avaliada como True. Você pode especificar uma ação UPDATE ou uma ação DELETE.

UPDATE

Atualiza a linha correspondente em target_table. Somente os valores no col_name que você especificar serão atualizados.

DELETE

Exclui a linha correspondente em target_table.

WHEN NOT MATCHED

Especifica a ação a ser executada quando a condição de correspondência é avaliada como False ou Unknown. Você só pode especificar a ação de inserção INSERT para essa cláusula.

INSERT

Insere linhas de source_table em target_table que não correspondem a nenhuma linha em target_table, de acordo com match_condition. O col_name de destino pode ser listado em qualquer ordem. Se você não fornecer nenhum valor de col_name, a ordem padrão será todas as colunas da tabela na ordem declarada.

col_name

Um ou mais nomes de coluna que você deseja modificar. Não inclui o nome da tabela ao especificar a coluna de destino.

expr

A expressão que define o novo valor para col_name.

REMOVE DUPLICATES

Especifica que o comando MERGE é executado no modo simplificado. O modo simplificado tem os seguintes requisitos:

- target_table e source_table devem ter o mesmo número de colunas, tipos de coluna compatíveis e a mesma ordem de coluna.
- A cláusula WHEN e as cláusulas UPDATE e INSERT devem ser omitidas do comando MERGE.

- A cláusula REMOVE DUPLICATES deve ser usada no comando MERGE.

No modo simplificado, o MERGE faz o seguinte:

- As linhas em target_table que têm uma correspondência em source_table são atualizadas para corresponder aos valores em source_table.
- As linhas em source_table que não têm uma correspondência em target_table são inseridas em target_table.
- Quando várias linhas em target_table correspondem à mesma linha em source_table, as linhas duplicadas são removidas. O Amazon Redshift mantém uma única linha e a atualiza. As linhas duplicadas que não correspondem a uma linha em source_table permanecem inalteradas.

REMOVE DUPLICATES oferece melhor performance do que WHEN MATCHED e WHEN NOT MATCHED. Recomendamos usar REMOVE DUPLICATES se target_table e source_table forem compatíveis e você não precisar preservar linhas duplicadas em target_table.

Observações de uso

- Para executar instruções MERGE, você deve ser o proprietário das tabelas source_table e target_table ou ter a permissão SELECT para essas tabelas. Além disso, você deve ter as permissões UPDATE, DELETE e INSERT para target_table, dependendo das operações incluídas em sua instrução MERGE.
- target_table não pode ser uma tabela do sistema, tabela de catálogo ou tabela externa.
- source_table e target_table não podem ser a mesma tabela.
- Não é possível usar a cláusula WITH em uma instrução MERGE.
- As linhas em target_table não podem estabelecer correspondência com várias linhas em source_table.

Considere o seguinte exemplo:

```
CREATE TABLE target (id INT, name CHAR(10));
CREATE TABLE source (id INT, name CHAR(10));

INSERT INTO target VALUES (1, 'Bob'), (2, 'John');
INSERT INTO source VALUES (1, 'Tony'), (1, 'Alice'), (3, 'Bill');

MERGE INTO target USING source ON target.id = source.id
WHEN MATCHED THEN UPDATE SET id = source.id, name = source.name
WHEN NOT MATCHED THEN INSERT VALUES (source.id, source.name);
```

```
ERROR: Found multiple matches to update the same tuple.

MERGE INTO target USING source ON target.id = source.id
WHEN MATCHED THEN DELETE
WHEN NOT MATCHED THEN INSERT VALUES (source.id, source.name);
ERROR: Found multiple matches to update the same tuple.
```

Nas duas instruções MERGE, a operação falha porque há várias linhas na tabela `source` com um valor de ID de 1.

- match_condition e expr não podem fazer referência parcial a colunas do tipo SUPER. Por exemplo, se seu objeto do tipo SUPER for uma matriz ou uma estrutura, você não poderá usar elementos individuais dessa coluna para match_condition ou expr, mas poderá usar a coluna inteira.

  Considere o seguinte exemplo:

```
CREATE TABLE IF NOT EXISTS target (key INT, value SUPER);
CREATE TABLE IF NOT EXISTS source (key INT, value SUPER);

INSERT INTO target VALUES (1, JSON_PARSE('{"key": 88}'));
INSERT INTO source VALUES (1, ARRAY(1, 'John')), (2, ARRAY(2, 'Bill'));

MERGE INTO target USING source ON target.key = source.key
WHEN matched THEN UPDATE SET value = source.value[0]
WHEN NOT matched THEN INSERT VALUES (source.key, source.value[0]);
ERROR: Partial reference of SUPER column is not supported in MERGE statement.
```

  Para obter mais informações sobre o tipo SUPER, consulte Tipo SUPER.

- Se source_table for grande, definir as colunas de junção de target_table e source_table como chaves de distribuição poderá melhorar a performance.
- Para usar a cláusula REMOVE DUPLICATES, você precisa das permissões SELECT, INSERT e DELETE para target_table.
- source_table pode ser uma visualização ou subconsulta. Veja a seguir um exemplo de uma instrução MERGE em que source_table é uma subconsulta que remove linhas duplicadas.

```
MERGE INTO target
USING (SELECT id, name FROM source GROUP BY 1, 2) as my_source
ON target.id = my_source.id
WHEN MATCHED THEN UPDATE SET id = my_source.id, name = my_source.name
WHEN NOT MATCHED THEN INSERT VALUES (my_source.id, my_source.name);
```

- O destino não pode ser uma fonte de dados de nenhuma subconsulta da mesma instrução MERGE. Por exemplo, o comando SQL a seguir retorna um erro como ERRO: a visualização/subconsulta de origem na instrução MERGE não pode fazer referência à tabela de destino. porque a subconsulta faz referência a `target` em vez de `source`.

```
MERGE INTO target
USING (SELECT id, name FROM target GROUP BY 1, 2) as my_source
ON target.id = my_source.id
WHEN MATCHED THEN UPDATE SET id = my_source.id, name = my_source.name
WHEN NOT MATCHED THEN INSERT VALUES (my_source.id, my_source.name);
```

Exemplos

O exemplo a seguir cria duas tabelas e executa uma operação MERGE nelas, atualizando as linhas correspondentes na tabela de destino e inserindo linhas que não correspondem. Depois, ele insere outro valor na tabela de origem e executa outra operação MERGE, desta vez excluindo as linhas correspondentes e inserindo a nova linha da tabela de origem.

Primeiro, crie e preencha as tabelas de origem e de destino.

```
CREATE TABLE target (id INT, name CHAR(10));
CREATE TABLE source (id INT, name CHAR(10));

INSERT INTO target VALUES (101, 'Bob'), (102, 'John'), (103, 'Susan');
INSERT INTO source VALUES (102, 'Tony'), (103, 'Alice'), (104, 'Bill');

SELECT * FROM target;
 id  |    name
-----+------------
 101 | Bob
 102 | John
 103 | Susan
(3 rows)

SELECT * FROM source;
 id  |    name
-----+------------
 102 | Tony
 103 | Alice
 104 | Bill
```

```
(3 rows)
```

Depois, mescle a tabela de origem com a tabela de destino, atualizando a tabela de destino com linhas correspondentes e insira linhas da tabela de origem que não tenham correspondência.

```
MERGE INTO target USING source ON target.id = source.id
WHEN MATCHED THEN UPDATE SET id = source.id, name = source.name
WHEN NOT MATCHED THEN INSERT VALUES (source.id, source.name);

SELECT * FROM target;
 id  |    name
-----+------------
 101 | Bob
 102 | Tony
 103 | Alice
 104 | Bill
(4 rows)
```

Observe que as linhas com valores de id 102 e 103 são atualizadas para corresponder aos valores dos nomes da tabela de destino. Além disso, uma nova linha com um valor de id 104 e o valor de nome Bill é inserida na tabela de destino.

Depois, insira uma nova linha na tabela de origem.

```
INSERT INTO source VALUES (105, 'David');

SELECT * FROM source;
 id  |    name
-----+------------
 102 | Tony
 103 | Alice
 104 | Bill
 105 | David
(4 rows)
```

Por fim, execute uma operação de mesclagem excluindo linhas correspondentes na tabela de destino e inserindo linhas que não correspondem.

```
MERGE INTO target USING source ON target.id = source.id
WHEN MATCHED THEN DELETE
WHEN NOT MATCHED THEN INSERT VALUES (source.id, source.name);
```

```
SELECT * FROM target;
 id  |    name
-----+------------
 101 | Bob
 105 | David
(2 rows)
```

As linhas com valores de id 102, 103 e 104 são excluídas da tabela de destino, e uma nova linha com um valor de id 105 e valor de nome David é inserida na tabela de destino.

O exemplo a seguir mostra a sintaxe simplificada de um comando MERGE que usa a cláusula REMOVE DUPLICATES.

```
CREATE TABLE target (id INT, name CHAR(10));
CREATE TABLE source (id INT, name CHAR(10));

INSERT INTO target VALUES (30, 'Tony'), (11, 'Alice'), (23, 'Bill');
INSERT INTO source VALUES (23, 'David'), (22, 'Clarence');

MERGE INTO target USING source ON target.id = source.id REMOVE DUPLICATES;

SELECT * FROM target;
id | name
---+------------
30 | Tony
11 | Alice
23 | David
22 | Clarence
(4 rows)
```

O exemplo a seguir mostra a sintaxe simplificada de um comando MERGE que usa a cláusula REMOVE DUPLICATES, que remove linhas duplicadas de target_table quando elas têm linhas correspondentes em source_table.

```
CREATE TABLE target (id INT, name CHAR(10));
CREATE TABLE source (id INT, name CHAR(10));

INSERT INTO target VALUES (30, 'Tony'), (30, 'Daisy'), (11, 'Alice'), (23, 'Bill'),
 (23, 'Nikki');
INSERT INTO source VALUES (23, 'David'), (22, 'Clarence');

MERGE INTO target USING source ON target.id = source.id REMOVE DUPLICATES;
```

```
SELECT * FROM target;
id | name
---+------------
30 | Tony
30 | Daisy
11 | Alice
23 | David
22 | Clarence
(5 rows)
```

Depois que MERGE é executado, há apenas uma linha com um valor de ID de 23 em target_table. Como não havia nenhuma linha em source_table com o valor de ID 30, as duas linhas duplicadas com valores de ID de 30 permanecem em target_table.

Consulte também

INSERT, UPDATE, DELETE

PREPARE

Prepare uma instrução para execução.

PREPARE cria uma instrução preparada. Quando a instrução PREPARE é executada, a instrução especificada (SELECT, INSERT, UPDATE ou DELETE) é analisada, reescrita e planejada. Quando um comando EXECUTE é emitido para a instrução preparada, o Amazon Redshift pode optar por revisar o plano de execução da consulta (para melhorar a performance com base nos valores de parâmetro especificados) antes de executar a instrução preparada.

Sintaxe

```
PREPARE plan_name [ (datatype [, ...] ) ] AS statement
```

Parâmetros

plan_name

Nome arbitrário dado a esta instrução preparada em particular. Deve ser exclusivo em uma única sessão e é usado subsequentemente para executar ou liberar uma instrução preparada anteriormente.

tipodedados

Tipo de dados de um parâmetro da instrução preparada. Para consultar os parâmetros na própria instrução preparada, use $1, $2 e assim por diante, até no máximo $32767.

instrução

Qualquer instrução SELECT, INSERT, UPDATE ou DELETE.

Observações de uso

Instruções preparadas podem ter parâmetros: valores que são substituídos quando a instrução é executada. Para incluir parâmetros em uma instrução preparada, forneça uma lista de tipos de dados na instrução PREPARE e, na própria instrução a ser preparada, procure os parâmetros por posição usando a anotação $1, $2, ... O número máximo de parâmetros é 32.767. Ao executar a instrução, especifique os valores reais para esses parâmetros da instrução EXECUTE. Para obter mais detalhes, consulte EXECUTE.

Instruções preparadas duram somente a sessão atual. Quando a sessão termina, a instrução preparada é descartada e deve ser recriada antes de ser usada novamente. Isso também significa que uma única instrução preparada não pode ser usada por vários clientes de banco de dados ao mesmo tempo. No entanto, cada cliente pode criar sua própria instrução preparada para usar. A instrução preparada pode ser removida manualmente usando o comando DEALLOCATE.

Instruções preparadas têm a maior vantagem de performance quando uma única sessão está sendo usada para executar um grande número de instruções semelhantes. Como mencionado, para cada nova execução de instrução preparada, o Amazon Redshift pode revisar o plano de execução da consulta para melhorar a performance com base nos valores de parâmetro especificados. Para examinar o plano de execução de consulta escolhido pelo Amazon Redshift para qualquer comando EXECUTE específico, use o comando EXPLAIN.

Para obter mais informações sobre o planejamento de consultas e as estatísticas coletadas pelo Amazon Redshift para otimização de consulta, consulte o comando ANALYZE.

Exemplos

Crie uma tabela temporária, prepare a instrução INSERT e execute-a:

```
DROP TABLE IF EXISTS prep1;
CREATE TABLE prep1 (c1 int, c2 char(20));
PREPARE prep_insert_plan (int, char)
```

```
AS insert into prep1 values ($1, $2);
EXECUTE prep_insert_plan (1, 'one');
EXECUTE prep_insert_plan (2, 'two');
EXECUTE prep_insert_plan (3, 'three');
DEALLOCATE prep_insert_plan;
```

Prepare uma instrução SELECT e execute-a:

```
PREPARE prep_select_plan (int)
AS select * from prep1 where c1 = $1;
EXECUTE prep_select_plan (2);
EXECUTE prep_select_plan (3);
DEALLOCATE prep_select_plan;
```

Consulte também

DEALLOCATE, EXECUTE

REFRESH MATERIALIZED VIEW

Atualiza uma visualização materializada.

Quando você cria uma visão materializada, seu conteúdo reflete o estado da tabela ou das tabelas do banco de dados subjacente na ocasião. Os dados na visão materializada permanecem inalterados, mesmo quando aplicações fazem alterações nos dados nas tabelas subjacentes.

Para atualizar os dados na visão materializada, você pode usar a instrução `REFRESH MATERIALIZED VIEW` a qualquer momento. Ao usar essa instrução, o Amazon Redshift identifica as alterações que ocorreram na tabela ou tabelas base e aplica essas alterações à visão materializada.

Para obter mais informações sobre visões materializadas, consulte Visões materializadas no Amazon Redshift.

Sintaxe

```
REFRESH MATERIALIZED VIEW mv_name [ RESTRICT | CASCADE ]
```

Parâmetros

mv_name

O nome da visão materializada a ser atualizada.

RESTRICT

Palavra-chave opcional. Atualiza a visão materializada especificada, mas não as respectivas visões materializadas dependentes. O padrão se nem RESTRICT nem CASCADE forem especificadas.

CASCADE

Palavra-chave opcional. Atualiza a visão materializada especificada e todas as respectivas visões materializadas dependentes.

Observações de uso

Somente o proprietário de uma visão materializada pode executar uma operação `REFRESH MATERIALIZED VIEW` nela. Além disso, o proprietário deve ter o privilégio SELECT nas tabelas base subjacentes para executar com êxito `REFRESH MATERIALIZED VIEW`.

Quando uma visualização materializada exige um recálculo total, o proprietário também deve ter o privilégio CREATE no esquema que contém a exibição materializada. Um recálculo total pode ser acionado por operações como VACUUM ou TRUNCATE em tabelas base.

O comando `REFRESH MATERIALIZED VIEW` é executado como uma transação própria. A semântica de transação do Amazon Redshift é seguida para determinar quais dados das tabelas base são visíveis para o comando `REFRESH`, ou quando as alterações feitas pelo comando `REFRESH` são tornadas visíveis para outras transações em execução no Amazon Redshift.

- Para visões materializadas incrementais, o `REFRESH MATERIALIZED VIEW` usa apenas as linhas da tabela base que já estão confirmadas. Portanto, se a operação de atualização for executada após uma instrução DML (Data Manipulation Language) na mesma transação, as alterações dessa instrução DML não serão visíveis para serem atualizadas.
- Para uma atualização completa de uma visão materializada, o `REFRESH MATERIALIZED VIEW` vê todas as linhas da tabela base visíveis para a transação de atualização, de acordo com a semântica de transação usual do Amazon Redshift.
- Dependendo do tipo de argumento de entrada, o Amazon Redshift ainda oferece suporte à atualização incremental de visões materializadas para as seguintes funções com tipos específicos de argumentos de entrada: DATE (timestamp), DATE_PART (date, time, interval, time-tz), DATE_TRUNC (timestamp, interval).
- A atualização incremental é compatível em uma visão materializada na qual a tabela base está e uma unidade de compartilhamento de dados.

- Não é possível atualizar visões materializadas compartilhadas de clusters remotos de unidades de compartilhamento de dados quando as visões materializadas contêm referências a outras visões materializadas, tabelas do Spectrum, tabelas definidas em um cluster diferente do Redshift ou UDFs. Essas visões materializadas podem ser atualizadas no cluster local (produtor).

Algumas operações no Amazon Redshift interagem com visões materializadas. Algumas dessas operações podem forçar uma operação `REFRESH MATERIALIZED VIEW` a recalcular totalmente a visão materializada, mesmo que a consulta que define a visão materializada use apenas os recursos SQL qualificados para atualização incremental. Por exemplo:

- Se as visões materializadas não forem atualizadas, as operações de limpeza em segundo plano podem ser bloqueadas. Depois de um período limite definido internamente, será permitido executar uma operação de vácuo. Quando essa operação de vácuo acontece, todas as visões materializadas dependentes são marcadas para recomputação na próxima atualização (mesmo que elas sejam incrementais). Para obter informações sobre VACUUM, consulte VACUUM. Para obter mais informações sobre eventos e alterações de estado, consulte STL_MV_STATE.
- Algumas operações iniciadas pelo usuário nas tabelas base forçam a recomputação total das visões materializadas na próxima vez que uma operação de REFRESH seja executada. Exemplos de tais operações são um VACUUM chamado manualmente, um redimensionamento clássico, uma operação ALTER DISTKEY, uma operação ALTER SORTKEY e uma operação truncada. Em alguns casos, as operações automáticas também podem resultar em uma visão materializada sendo totalmente recalculada na próxima vez que uma operação REFRESH for executada. Por exemplo, uma operação de exclusão automática por autovacuum pode causar uma recalculação completa. Para obter mais informações sobre eventos e alterações de estado, consulte STL_MV_STATE.

Visualizações materializadas do Iceberg

Para visualizações materializadas do Iceberg criadas com USING ICEBERG, as seguintes notas de uso se aplicam:

- O chamador deve ter a permissão ALTER na visualização materializada. O perfil definidor de MV (o perfil do IAM registrado no momento da criação) deve ter a permissão SELECT em todas as tabelas de origem.
- Para atualização incremental, o Amazon Redshift comporta as funções agregadas COUNT e SUM. As visualizações materializadas usando outros agregados (MIN, MAX, AVG) usam a atualização completa.

- Vários clusters do Amazon Redshift podem tentar atualizar a mesma visualização materializada do Iceberg simultaneamente. O Amazon Redshift usa o controle de simultaneidade otimista (OCC) por meio do AWS Glue Data Catalog para garantir que apenas uma atualização seja bem-sucedida. Se outro cluster concluir a atualização primeiro, a operação local será interrompida.
- As opções CASCADE e RESTRICT não são aceitas para visualizações materializadas do Iceberg.

As seguintes operações nas tabelas básicas forçam um recálculo completo na próxima atualização:

- Expiração do snapshot da tabela de origem (quando os snapshots gravados na última atualização não estão mais disponíveis).
- Modificação de dados na visualização materializada por um mecanismo ou ferramenta externa.

Atualização em cascata

A opção CASCADE atualiza a visão materializada especificada e todas as respectivas visões materializadas dependentes, em ordem de dependência: as MVs básicas são ATUALIZADAS antes das MVs na parte superior (ordem topológica). Com isso, é possível atualizar um conjunto aninhado de visualizações materializadas com um único comando.

A opção RESTRICT (o padrão se nem RESTRICT nem CASCADE forem especificadas) atualiza somente a visão materializada especificada.

Ao usar a opção CASCADE, as seguintes regras se aplicam:

- Somente o proprietário da visão materializada ou um superusuário pode executar o comando `REFRESH MATERIALIZED VIEW ... CASCADE`.
- Se alguma das visões materializadas em cascata não puder ser atualizada, toda a operação em cascata será interrompida.

A funcionalidade de atualização em cascata só pode ser usada em MVs aninhadas acima das visões materializadas locais e de streaming. Visões materializadas com outros tipos de origem, como Spectrum ou compartilhamento de dados, não são permitidas no modo cascata. CASCADE executa a atualização em uma única transação para todos as MVs aninhadas.

Atualização incremental para visões materializadas em uma unidade de compartilhamento de dados

O Amazon Redshift oferece suporte à atualização automática e incremental de visões materializadas em uma unidade de compartilhamento de dados do consumidor quando as tabelas base são compartilhadas. A atualização incremental é uma operação em que o Amazon Redshift identifica alterações em uma ou mais tabelas base que ocorreram após a atualização anterior e atualiza somente os registros correspondentes na visão materializada. Para obter mais informações sobre este comportamento, consulte CREATE MATERIALIZED VIEW.

Limitações para atualização incremental

Atualmente, o Amazon Redshift não oferece suporte à atualização incremental para visões materializadas definidas com uma consulta usando qualquer um dos seguintes elementos SQL:

- OUTER JOIN (RIGHT, LEFT ou FULL).
- Operações de conjuntos: (UNION, INTERSECT, EXCEPT, MINUS)
- UNION ALL quando ocorre em uma subconsulta e uma função agregada ou uma cláusula GROUP BY está presente na consulta ou a visão materializada de destino contém uma chave de classificação.
- Funções agregadas MEDIAN, PERCENTILE_CONT, LISTAGG, STDDEV_SAMP, STDDEV_POP, APPROXIMATE COUNT, APPROXIMATE PERCENTILE e funções agregadas bit a bit.

> **Note**
>
> A funções agregadas COUNT, SUM, MIN, MAX e AVG não são comportadas.

- Funções agregadas DISTINCT, como DISTINCT COUNT, DISTINCT SUM e assim por diante.
- Funções de janela.
- Uma consulta que usa tabelas temporárias para otimização de consultas, como para otimizar subexpressões comuns.
- Subconsultas
- Tabelas externas referenciando os formatos a seguir na consulta que define a visão materializada.
  - Delta Lake
  - Hudi

Há suporte para a atualização incremental para visões materializadas definidas usando formatos diferentes dos listados acima. Para obter mais informações, consulte Visões materializadas em tabelas externas de data lake no Amazon Redshift Spectrum.

- Funções mutáveis, como funções de data-hora, funções aleatórias e não-estáveis definidas pelo usuário.
- Com relação a limitações relacionadas à atualização incremental para integrações ETL zero, consulte Considerações ao usar integrações ETL zero com o Amazon Redshift.
- Acesso a tabelas de mais de um banco de dados.

Para obter mais informações sobre as limitações das visões materializadas, incluindo o efeito de operações em segundo plano, como VACUUM, nas operações de atualização de visões materializadas, consulte Observações de uso.

Para visualizações materializadas do Iceberg, o Amazon Redshift não comporta atualização incremental para visualizações materializadas definidas com qualquer um dos seguintes elementos SQL:

- OUTER JOIN (RIGHT, LEFT ou FULL).
- Operações de conjuntos: UNION, UNION ALL, INTERSECT, EXCEPT, MINUS.
- Funções agregadas além de COUNT e SUM.
- Funções agregadas DISTINCT, como COUNT(DISTINCT) e SUM(DISTINCT).
- Funções de janela
- Subconsultas
- GROUPING SETS, ROLLUP, CUBE
- DISTINCT

Quando a atualização incremental não é aceita, o Amazon Redshift executa automaticamente uma atualização completa.

Exemplos

O exemplo a seguir atualiza a visão materializada `tickets_mv`.

```
REFRESH MATERIALIZED VIEW tickets_mv;
```

O seguinte exemplo atualiza a visão materializada `products_mv` e todas as respectivas visões materializadas dependentes:

```
REFRESH MATERIALIZED VIEW products_mv CASCADE;
```

RESET

Restaura o valor de um parâmetro de configuração para seu valor padrão.

Você pode redefinir um único parâmetro especificado ou todos os parâmetros de uma vez. Para configurar um parâmetro para um valor específico, use o comando SET. Para exibir o valor atual de um parâmetro, use o comando SHOW.

Sintaxe

```
RESET { parameter_name | ALL }
```

A instrução a seguir define o valor de uma variável de contexto de sessão como NULL.

```
RESET { variable_name | ALL }
```

Parâmetros

parameter_name

Nome do parâmetro a ser redefinido. Consulte Modificar a configuração do servidor para obter mais documentos sobre parâmetros.

ALL

Redefine todos os parâmetros de tempo de execução, incluindo todas as variáveis de contexto da sessão.

variável

O nome da variável a ser redefinida. Se o valor a ser redefinido for uma variável de contexto de sessão, o Amazon Redshift o definirá como NULL.

Exemplos

O exemplo a seguir redefine o parâmetro `query_group` para seu valor padrão:

```
reset query_group;
```

O exemplo a seguir redefine todos os parâmetros de tempo de execução para seus valores padrão.

```
reset all;
```

O exemplo a seguir redefine as variáveis de contexto.

```
RESET app_context.user_id;
```

REVOKE

Remove permissões de acesso, como permissões para criar, descartar ou atualizar tabelas, de um usuário ou uma função.

Você só pode CONCEDER ou REVOGAR permissões de USO em um esquema externo para usuários de banco de dados e funções que usem a sintaxe ON SCHEMA. Ao usar ON EXTERNAL SCHEMA com AWS Lake Formation, só é possível conceder (GRANT) e revogar (REVOKE) permissões para um perfil do AWS Identity and Access Management (IAM). Para obter a lista de permissões, consulte a sintaxe.

Para procedimentos armazenados, USAGE ON LANGUAGE `plpgsql` é concedido para PUBLIC, por padrão. A permissão EXECUTE ON PROCEDURE é concedida somente ao proprietário e aos superusuários, por padrão.

Especifique as permissões que deseja remover no comando REVOKE. Para conceder permissões, use o comando GRANT.

Sintaxe

```
REVOKE [ GRANT OPTION FOR ]
{ { SELECT | INSERT | UPDATE | DELETE | DROP | REFERENCES | ALTER | TRUNCATE } [,...] |
 ALL [ PRIVILEGES ] }
ON { [ TABLE ] table_name [, ...] | ALL TABLES IN SCHEMA schema_name [, ...] }
FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
[ RESTRICT ]

REVOKE [ GRANT OPTION FOR ]
{ { CREATE | TEMPORARY | TEMP | ALTER } [,...] | ALL [ PRIVILEGES ] }
```

```
ON DATABASE db_name [, ...]
FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
[ RESTRICT ]

REVOKE [ GRANT OPTION FOR ]
{ { CREATE | USAGE | ALTER | DROP } [,...] | ALL [ PRIVILEGES ] }
ON SCHEMA schema_name [, ...]
FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
[ RESTRICT ]

REVOKE [ GRANT OPTION FOR ]
EXECUTE
    ON FUNCTION function_name ( [ [ argname ] argtype [, ...] ] ) [, ...]
    FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
[ RESTRICT ]

REVOKE [ GRANT OPTION FOR ]
{ { EXECUTE } [,...] | ALL [ PRIVILEGES ] }
    ON PROCEDURE procedure_name ( [ [ argname ] argtype [, ...] ] ) [, ...]
    FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
[ RESTRICT ]

REVOKE [ GRANT OPTION FOR ]
USAGE
    ON LANGUAGE language_name [, ...]
    FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
[ RESTRICT ]

REVOKE [GRANT OPTION FOR]
{ { ALTER | DROP} [,...] | ALL [ PRIVILEGES ] }
    ON COPY JOB job_name [,...]
    FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]

REVOKE [GRANT OPTION FOR]
{ { ALTER | DROP | USAGE } [,...] | ALL [ PRIVILEGES ] }
    ON TEMPLATE template_name [,...]
    FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
```

Revogar permissões por coluna para tabelas

A seguir está a sintaxe para permissões por coluna em tabelas e visualizações do Amazon Redshift.

```
REVOKE { { SELECT | UPDATE } ( column_name [, ...] ) [, ...] | ALL [ PRIVILEGES ]
 ( column_name [,...] ) }
```

```
    ON { [ TABLE ] table_name [, ...] }
    FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
    [ RESTRICT ]
```

Revogar permissões ASSUMEROLE

A seguir está a sintaxe para revogar a permissão ASSUMEROLE de usuários e grupos com um perfil especificado.

```
REVOKE ASSUMEROLE
    ON { 'iam_role' [, ...]  | default | ALL }
    FROM { user_name | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
    FOR { ALL | COPY | UNLOAD | EXTERNAL FUNCTION | CREATE MODEL }
```

Revogar permissões do Redshift Spectrum para Lake Formation

A seguir está a sintaxe para integração do Redshift Spectrum com Lake Formation.

```
REVOKE [ GRANT OPTION FOR ]
{ SELECT | ALL [ PRIVILEGES ] } ( column_list )
    ON EXTERNAL TABLE schema_name.table_name
    FROM { IAM_ROLE iam_role } [, ...]

REVOKE [ GRANT OPTION FOR ]
{ { SELECT | ALTER | DROP | DELETE | INSERT }  [, ...] | ALL [ PRIVILEGES ] }
    ON EXTERNAL TABLE schema_name.table_name [, ...]
    FROM { { IAM_ROLE iam_role } [, ...] | PUBLIC }

REVOKE [ GRANT OPTION FOR ]
{ { CREATE | ALTER | DROP }  [, ...] | ALL [ PRIVILEGES ] }
    ON EXTERNAL SCHEMA schema_name [, ...]
    FROM { IAM_ROLE iam_role } [, ...]
```

Revogar permissões de unidade de compartilhamento de dados

Permissões da unidade de compartilhamento de dados no lado do produtor

Esta é a sintaxe para usar REVOKE a fim de remover permissões ALTER ou SHARE de um usuário ou uma função. O usuário cujas permissões foram revogadas não pode mais alterar a unidade de compartilhamento de dados nem conceder o uso a um consumidor.

```
REVOKE { ALTER | SHARE } ON DATASHARE datashare_name
```

```
 FROM { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]
```

Esta é a sintaxe para usar REVOKE a fim de remover o acesso do consumidor a uma unidade de compartilhamento de dados.

```
REVOKE USAGE
 ON DATASHARE datashare_name
 FROM NAMESPACE 'namespaceGUID' [, ...] | ACCOUNT 'accountnumber' [ VIA DATA CATALOG ]
 [, ...]
```

Este é um exemplo de como revogar o uso de uma unidade de compartilhamento de dados de uma conta do Lake Formation.

```
REVOKE USAGE ON DATASHARE salesshare FROM ACCOUNT '123456789012' VIA DATA CATALOG;
```

Permissões da unidade de compartilhamento de dados no lado do consumidor

A seguir está a sintaxe REVOKE para permissões de uso de compartilhamento de dados em um banco de dados ou esquema específico criado a partir de um datashare. A revogação da permissão de uso de um banco de dados criado com a cláusula WITH PERMISSIONS não revoga nenhuma permissão adicional concedida a um usuário ou a uma função, inclusive permissões no nível de objeto concedidas para objetos subjacentes. Se você conceder novamente a permissão de uso a esse usuário ou função, eles manterão todas as permissões adicionais que tinham antes de você ter revogado o uso.

```
REVOKE USAGE ON { DATABASE shared_database_name [, ...] | SCHEMA shared_schema}
 FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
```

Revogação de permissões em escopo

Com as permissões em escopo, é possível conceder permissões a um usuário ou função em todos os objetos de um tipo em um banco de dados ou esquema. Usuários e funções com permissões em escopo têm as permissões especificadas em todos os objetos atuais e futuros no banco de dados ou esquema.

Você pode visualizar o escopo das permissões com escopo em nível de banco de dados em SVV_DATABASE_PRIVILEGES. Você pode visualizar o escopo das permissões com escopo em nível de esquema em SVV_SCHEMA_PRIVILEGES.

Para ter mais informações sobre permissões com escopo definido, consulte Permissões em escopo.

Esta é a sintaxe para revogação de permissões em escopo de usuários ou perfis.

```
REVOKE [ GRANT OPTION ]
{ CREATE | USAGE | ALTER | DROP } [,...] | ALL [ PRIVILEGES ] }
FOR SCHEMAS IN
DATABASE db_name
FROM { username | ROLE role_name } [, ...]

REVOKE [ GRANT OPTION ]
{ { SELECT | INSERT | UPDATE | DELETE | DROP | ALTER | TRUNCATE | REFERENCES }
 [, ...] } | ALL [PRIVILEGES] } }
FOR TABLES IN
{ SCHEMA schema_name [ DATABASE db_name ] | DATABASE db_name }
FROM { username | ROLE role_name } [, ...]

REVOKE [ GRANT OPTION ] { EXECUTE | ALL [ PRIVILEGES ] }
FOR FUNCTIONS IN
{ SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
FROM { username | ROLE role_name } [, ...]

REVOKE [ GRANT OPTION ] { EXECUTE | ALL [ PRIVILEGES ] }
FOR PROCEDURES IN
{ SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
FROM { username | ROLE role_name } [, ...]

REVOKE [ GRANT OPTION ] USAGE
FOR LANGUAGES IN
DATABASE db_name
FROM { username | ROLE role_name } [, ...]

REVOKE [GRANT_OPTION]
{ { CREATE | ALTER | DROP} [,...] | ALL [ PRIVILEGES ] }
FOR COPY JOBS
IN DATABASE db_name
FROM { username [ WITH GRANT OPTION ] | ROLE role_name } [, ...]

REVOKE [ GRANT OPTION ]
{ {ALTER | DROP  | USAGE } [,...] | ALL [ PRIVILEGES ] }
FOR TEMPLATES IN
{ SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
FROM { username | ROLE role_name } [, ...]
```

Observe que as permissões no escopo não fazem distinção entre permissões para funções e procedimentos. Por exemplo, a declaração a seguir revoga as permissões EXECUTE para funções e procedimentos de bob no esquema Sales_schema.

```
REVOKE EXECUTE FOR FUNCTIONS IN SCHEMA Sales_schema FROM bob;
```

Revogar permissões de machine learning

A seguir está a sintaxe para permissões de modelos de machine learning no Amazon Redshift.

```
REVOKE [ GRANT OPTION FOR ]
    CREATE MODEL FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
    [ RESTRICT ]

REVOKE [ GRANT OPTION FOR ]
    { EXECUTE | ALL [ PRIVILEGES ] }
    ON MODEL model_name [, ...]

    FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
    [ RESTRICT ]
```

Revogar permissões de perfil

A sintaxe a seguir revoga permissões de perfil no Amazon Redshift.

```
REVOKE [ ADMIN OPTION FOR ] { ROLE role_name } [, ...] FROM { user_name } [, ...]
```

```
REVOKE { ROLE role_name } [, ...] FROM { ROLE role_name } [, ...]
```

A sintaxe a seguir revoga permissões de sistema a perfis no Amazon Redshift.

```
REVOKE
  {
    { CREATE USER | DROP USER | ALTER USER |
    CREATE SCHEMA | DROP SCHEMA |
    ALTER DEFAULT PRIVILEGES |
    ACCESS CATALOG |
```

```
    CREATE TABLE | DROP TABLE | ALTER TABLE |
    CREATE OR REPLACE FUNCTION | CREATE OR REPLACE EXTERNAL FUNCTION |
    DROP FUNCTION |
    CREATE OR REPLACE PROCEDURE | DROP PROCEDURE |
    CREATE OR REPLACE VIEW | DROP VIEW |
    CREATE MODEL | DROP MODEL |
    CREATE DATASHARE | ALTER DATASHARE | DROP DATASHARE |
    CREATE LIBRARY | DROP LIBRARY |
    CREATE ROLE | DROP ROLE
    TRUNCATE TABLE
    VACUUM | ANALYZE | CANCEL }[, ...]
  }
  | { ALL [ PRIVILEGES ] }
FROM { ROLE role_name } [, ...]
```

Revogar permissões para políticas de segurança

Veja a seguir a sintaxe para revogar permissões para explicar os filtros de política de segurança no nível da linha de uma consulta no plano EXPLAIN. As políticas de segurança possíveis incluem segurança por linha e políticas de mascaramento dinâmico de dados.

```
REVOKE EXPLAIN { RLS | MASKING } FROM ROLE rolename
```

Veja a seguir a sintaxe para revogar permissões para ignorar políticas de segurança por linha para uma consulta.

```
REVOKE IGNORE RLS FROM ROLE rolename
```

A sintaxe a seguir revoga permissões SELECT da política de segurança especificada. As políticas de segurança possíveis incluem segurança por linha e políticas de mascaramento dinâmico de dados.

```
REVOKE SELECT ON [ TABLE ] table_name [, ...]
            FROM { RLS | MASKING } POLICY policy_name [, ...]
```

Revogar a permissão DEBUG

Veja a seguir a sintaxe para revogar a permissão DEBUG em um banco de dados em um warehouse do Amazon Redshift com permissões federadas. A revogação da permissão DEBUG é apenas prospectiva: ela impede que novos registros de log sejam gerados com o acesso DEBUG, mas os registros de log gerados enquanto a permissão DEBUG estava ativa permanecem visíveis. Para

ter mais informações sobre a permissão DEBUG, consulte Observações de uso para conceder a permissão DEBUG.

```
REVOKE DEBUG
    ON DATABASE federated_db_name
    FROM { <prefix>:<name>
         | ROLE <idc_prefix>:<group_name>
         | ADMINS OF ACCOUNT 'account_id' }
```

Parâmetros

GRANT OPTION FOR

Revoga somente a opção de conceder uma permissão especificada a outros usuários e não revoga a própria permissão. Não é possível revogar GRANT OPTION de um grupo ou de PUBLIC.

SELECT

Revoga a permissão de selecionar dados de uma tabela ou visualização usando uma instrução SELECT.

INSERT

Revoga a permissão de carregar dados em uma tabela usando uma instrução INSERT ou COPY.

UPDATE

Revoga a permissão de atualizar uma coluna de tabela usando a instrução UPDATE.

DELETE

Revoga a permissão de excluir uma linha de dados de uma tabela.

REFERENCES

Revoga a permissão de criar uma restrição de chave externa. Você deve revogar essa permissão tanto na tabela referenciada quanto na tabela de referência.

TRUNCATE

Revoga a permissão para truncar uma tabela. Sem essa permissão, somente o proprietário de uma tabela ou um superusuário pode truncar uma tabela. Para obter mais informações sobre o comando TRUNCATE, consulte the section called “TRUNCATE”.

ALL [PRIVILEGES]

Revoga de um vez todas as permissões disponíveis do usuário ou grupo especificado. A palavra-chave PRIVILEGES é opcional.

> **Note**
>
> O Amazon Redshift não é compatível com as permissões RULE e TRIGGER. Para obter mais informações, acesse Recursos incompatíveis do PostgreSQL.

ALTER

Dependendo do objeto do banco de dados, as seguintes permissões são revogadas do usuário ou grupo de usuários:

- Para tabelas, ALTER revoga a permissão para alterar uma tabela ou visão. Para obter mais informações, consulte ALTER TABLE.
- Para bancos de dados, ALTER revoga a permissão para alterar um banco de dados. Para obter mais informações, consulte ALTER DATABASE.
- Para esquemas, ALTER revoga a permissão para alterar um esquema. Para obter mais informações, consulte ALTER SCHEMA.
- Para tabelas externas, ALTER revoga a permissão para alterar uma tabela em um AWS Glue Data Catalog habilitado para o Lake Formation. Essa permissão só é aplicada ao usar o Lake Formation.

DROP

Dependendo do objeto do banco de dados, revoga as seguintes permissões do usuário ou perfil:

- Para tabelas, DROP revoga a permissão para remover uma tabela ou visualização. Para obter mais informações, consulte DESCARTAR TABELA.
- Para bancos de dados, DROP revoga a permissão para remover um banco de dados. Para obter mais informações, consulte DROP DATABASE.
- Para esquemas, DROP revoga a permissão para remover um esquema. Para obter mais informações, consulte DROP SCHEMA.

ASSUMEROLE

Revoga a permissão de executar os comandos COPY, UNLOAD, EXTERNAL FUNCTION ou CREATE MODEL de usuários, perfis ou grupos com um perfil especificado.

ON [TABLE] table_name

Revoga as permissões especificadas em uma tabela ou visualização. A palavra-chave TABLE é opcional.

ON ALL TABLES IN SCHEMA schema_name

Revoga as permissões especificadas em todas as tabelas no esquema referenciado.

(column_name [,...]) ON TABLE table_name

Revoga as permissões especificadas de usuários, grupos ou PUBLIC nas colunas especificadas da tabela ou visualização do Amazon Redshift.

(column_list) ON EXTERNAL TABLE schema_name.table_name

Revoga as permissões especificadas de um perfil do IAM nas colunas especificadas da tabela do Lake Formation no esquema referenciado.

ON EXTERNAL TABLE schema_name.table_name

Revoga as permissões especificadas de um perfil do IAM nas tabelas especificadas do Lake Formation no esquema referenciado.

ON EXTERNAL SCHEMA schema_name

Revoga as permissões especificadas de um perfil do IAM no esquema referenciado.

FROM IAM_ROLE iam_role

Indica o perfil do IAM que perde as permissões.

ROLE role_name

Revoga as permissões do perfil especificado.

GROUP nome_grupo

Revoga as permissões do grupo de usuários especificado.

PUBLIC

Revoga as permissões de todos os usuários. PUBLIC representa um grupo que inclui sempre todos os usuários. As permissões de um usuário individual consistem na soma das permissões concedidas a PUBLIC, das permissões concedidas a todos os grupos aos quais o usuário pertence e de quaisquer permissões concedidas ao usuário individualmente.

A revogação de PUBLIC para uma tabela externa do Lake Formation resulta na revogação da permissão para o grupo todos do Lake Formation.

CREATE

Dependendo do objeto do banco de dados, as seguintes permissões são revogadas do usuário ou grupo de usuários:

- Para bancos de dados, usar a cláusula CREATE para revogar impede que os usuários criem esquemas no banco de dados.
- Para esquemas, usar a cláusula CREATE para revogar impede que os usuários criem objetos em um esquema. Para renomear um objeto, o usuário deve ter a permissão CREATE e ser proprietário do objeto a ser renomeado.

> **Note**
>
> Por padrão, todos os usuários têm permissões CREATE e USAGE no esquema PUBLIC.

TEMPORARY | TEMP

Revoga a permissão para criar tabelas temporárias no banco de dados especificado.

> **Note**
>
> Por padrão, os usuários recebem permissão para criar tabelas temporárias por associação automática ao grupo PUBLIC. Para remover a permissão para todos os usuários criarem tabelas temporárias, revogue a permissão TEMP do grupo PUBLIC. Depois, conceda explicitamente a permissão para criar tabelas temporárias a usuários ou grupos de usuários específicos.

ON DATABASE nome_bd

Revoga as permissões no banco de dados especificado.

DEBUG

Concede permissão para ver registros em log de registro seguros não suprimidos para um banco de dados em um warehouse do Amazon Redshift com permissões federadas. A revogação da

permissão DEBUG é apenas prospectiva: ela impede que novos registros de log sejam gerados com o acesso DEBUG, mas os registros gerados enquanto a permissão DEBUG estava ativa permanecem visíveis. Para obter mais informações, consulte Observações de uso para conceder a permissão DEBUG.

ADMINS OF ACCOUNT 'account_id'

Para a permissão DEBUG, revoga a permissão dos administradores da conta de consumidor especificada. Este formulário de beneficiário é aceito somente para a permissão DEBUG.

USAGE

Revoga as permissões USAGE em objetos de um esquema específico, que torna esses objetos inacessíveis aos usuários. Ações específicas nesses objetos devem ser revogadas separadamente (como a permissão EXECUTE em funções).

> **Note**
>
> Por padrão, todos os usuários têm permissões CREATE e USAGE no esquema PUBLIC.

ON SCHEMA schema_name

Revoga as permissões no esquema especificado. Você pode usar permissões de esquema para gerenciar a criação de tabelas. A permissão CREATE para um banco de dados controla somente a criação de esquemas.

RESTRICT

Revoga somente as permissões que o usuário concedeu diretamente. Esse comportamento é a configuração padrão.

EXECUTE ON PROCEDURE procedure_name

Revoga a permissão EXECUTE em um procedimento armazenado específico. Como os nomes de procedimento armazenado podem ser sobrecarregados, você deverá incluir a lista de argumentos para o procedimento. Para obter mais informações, consulte Nomeação de procedimentos armazenados.

EXECUTE ON ALL PROCEDURES IN SCHEMA procedure_name

Revoga as permissões especificadas em todos os procedimentos no esquema referenciado.

USAGE ON LANGUAGE nome_linguagem

Revoga a permissão USAGE em um idioma. Para funções definidas pelo usuário (UDFs) em Python, use `plpythonu`. Para UDFs SQL, use `sql`. Para procedimentos armazenados, use `plpgsql`.

Para criar uma UDF, é necessário ter permissão de uso na linguagem para SQL ou `plpythonu` (Python). Por padrão, USAGE ON LANGUAGE SQL é concedido para PUBLIC. No entanto, é necessário conceder explicitamente USAGE ON LANGUAGE PLPYTHONU a usuários ou grupos específicos.

Para revogar o uso na SQL, revogue primeiro o uso em PUBLIC. Depois, conceda o uso na SQL somente a usuários ou grupos específicos que tenham permissão para criar UDFs SQL. O exemplo a seguir revoga o uso na SQL em PUBLIC e concede o uso ao grupo de usuários `udf_devs`.

```
revoke usage on language sql from PUBLIC;
grant usage on language sql to group udf_devs;
```

Para obter mais informações, consulte Segurança e permissões de UDFs.

Para revogar o uso para procedimentos armazenados, primeiro revogue o uso em PUBLIC. Depois, conceda o uso na `plpgsql` somente a usuários ou grupos específicos que tenham permissão para criar procedimentos armazenados. Para obter mais informações, consulte Segurança e privilégios para procedimentos armazenados.

ON COPY JOB job_name

Revoga as permissões especificadas em um trabalho de cópia.

FOR { ALL | COPY | UNLOAD | EXTERNAL FUNCTION | CREATE MODEL } [, ...]

Especifica o comando SQL para o qual a permissão é revogada. Você pode especificar ALL para revogar a permissão nas instruções COPY, UNLOAD, EXTERN FUNCTION e CREATE MODEL. Esta cláusula aplica-se apenas à revogação da permissão ASSUMEROLE.

ALTER

Revoga a permissão ALTER para usuários ou grupos de usuários que permite que aqueles que não possuem uma unidade de compartilhamento de dados alterem a unidade de compartilhamento de dados. Esta permissão é necessária para adicionar ou remover objetos de

uma unidade de compartilhamento de dados ou para definir a propriedade PUBLICACCESSIBLE. Para obter mais informações, consulte ALTER DATASHARE.

SHARE

Revoga as permissões para que usuários e grupos de usuários adicionem consumidores a uma unidade de compartilhamento de dados. A revogação dessas permissões é necessária para impedir que o consumidor específico acesse a unidade de compartilhamento de dados de seus clusters.

ON DATASHARE datashare_name

Concede as permissões especificadas na unidade de compartilhamento de dados de referência.

FROM username

Indica o usuário que perde as permissões.

FROM GROUP group_name

Indica o grupo de usuários que perde as permissões.

WITH GRANT OPTION

Indica que o usuário que perde as permissões pode, por sua vez, revogar as mesmas permissões para outros. Não é possível revogar WITH GRANT OPTION de um grupo ou de PUBLIC.

USAGE

Quando USAGE é revogado para uma conta de consumidor ou namespace dentro da mesma conta, a conta de consumidor especificada ou namespace dentro de uma conta não pode acessar o datashare e os objetos do datashare de forma somente leitura.

Revogar a permissão USAGE revoga o acesso a uma unidade de compartilhamento de dados para os consumidores.

FROM NAMESPACE 'clusternamespace GUID'

Indica o namespace na mesma conta que faz com que os consumidores percam as permissões para a unidade de compartilhamento de dados. Os namespaces usam um identificador exclusivo global (GUID) alfanumérico de 128 bits.

FROM ACCOUNT 'accountnumber' [VIA DATA CATALOG]

Indica o número de outra conta que remove as permissões dos consumidores para a unidade de compartilhamento de dados. Especificar “VIA DATA CATALOG” indica que você está revogando

o uso da unidade de compartilhamento de dados para uma conta do Lake Formation. Omitir o número da conta significa que você está revogando da conta que detém o cluster.

ON DATABASE shared_database_name> [, ...]

Revoga as permissões de uso especificadas no banco de dados especificado que foi criado na unidade de compartilhamento de dados especificada.

ON SCHEMA shared_schema

Revoga as permissões especificadas no esquema especificado que foi criado na unidade de compartilhamento de dados especificada.

FOR { SCHEMAS | TABLES | FUNCTIONS | PROCEDURES | LANGUAGES | COPY JOBS} IN

Especifica os objetos de banco de dados dos quais revogar permissão. Os parâmetros após IN definem o escopo da permissão revogada.

CREATE MODEL

Revoga a permissão CREATE MODEL para criar modelos de machine learning no banco de dados especificado.

ON MODEL model_name

Revoga a permissão EXECUTE em um modelo específico.

ACCESS CATALOG

Revoga a permissão para exibir metadados relevantes de objetos aos quais a função tem acesso.

[ADMIN OPTION FOR] { role } [, ...]

A função que você revoga de um usuário especificado que tem a WITH ADMIN OPTION.

FROM { role } [, ...]

A função da qual você revoga a função especificada.

EXPLAIN { RLS | MASKING } FROM ROLE rolename

Revoga a permissão para explicar os filtros de política de segurança de uma consulta no plano EXPLAIN de um perfil. RLS revoga a permissão para explicar filtros de política de segurança por linha. MASKING revoga permissão a permissão para explicar os filtros da política de mascaramento dinâmico de dados.

IGNORE RLS FROM ROLE rolename

Revoga a permissão para ignorar políticas de segurança por linha para uma consulta de um perfil.

FROM { RLS | MASKING } POLICY policy_name

Indica a política de segurança que está perdendo as permissões. TO RLS POLICY indica uma política de segurança por linha. TO MASKING POLICY indica uma política de mascaramento dinâmico de dados.

Observações de uso

Para saber mais sobre as observações de uso de REVOKE, consulte the section called “Observações de uso”.

Exemplos

Para conferir exemplos de como usar REVOKE, consulte the section called “Exemplos”.

Observações de uso

Para revogar privilégios de um objeto, você deve atender a um dos seguintes critérios:

- Ser o proprietário do objeto.
- Ser um superusuário.
- Ter um privilégio concedido para o objeto e privilégio.

  Por exemplo, o comando a seguir fornece ao usuário HR a capacidade de executar comandos SELECT na tabela de funcionários e conceder e revogar o mesmo privilégio para outros usuários.

  ```
  grant select on table employees to HR with grant option;
  ```

  HR não pode revogar privilégios para qualquer operação além de SELECT ou em qualquer outra tabela que não seja de funcionários.

Superusuários podem acessar todos os objetos, independentemente de comandos GRANT e REVOKE que definem privilégios de objeto.

PUBLIC representa um grupo que inclui sempre todos os usuários. Por padrão, todos os membros de PUBLIC têm privilégios CREATE e USAGE no esquema PUBLIC. Para restringir as permissões de qualquer usuário no esquema PUBLIC, você deve primeiro revogar todas as permissões em PUBLIC no esquema PUBLIC e, depois, conceder privilégios a usuários ou grupos específicos. O exemplo a seguir controla os privilégios de criação de tabela no esquema PUBLIC.

```
revoke create on schema public from public;
```

Para revogar privilégios de uma tabela do Lake Formation, a função do IAM associada ao esquema externo da tabela deve ter permissão para revogar privilégios para a tabela externa. O exemplo a seguir cria um esquema externo com uma função do IAM associada `myGrantor`. A função do IAM `myGrantor` tem permissão para revogar permissões de outros. O comando REVOKE usa a permissão da função do IAM `myGrantor` que está associada ao esquema externo para revogar permissão para a função do IAM `myGrantee`.

```
create external schema mySchema
from data catalog
database 'spectrum_db'
iam_role 'arn:aws:iam::123456789012:role/myGrantor'
create external database if not exists;
```

```
revoke select
on external table mySchema.mytable
from iam_role 'arn:aws:iam::123456789012:role/myGrantee';
```

> **Note**
>
> Se a função do IAM também tiver a permissão `ALL` em um AWS Glue Data Catalog habilitado para o Lake Formation, a permissão `ALL` não será revogada. Somente a permissão `SELECT` é revogada. Você pode exibir as permissões do Lake Formation no console do Lake Formation.

Observações de uso para revogar a permissão ASSUMEROLE

As observações de uso a seguir são aplicáveis à revogação do privilégio ASSUMEROLE no Amazon Redshift.

Somente um superusuário de banco de dados pode revogar o privilégio ASSUMEROLE para usuários e grupos. Um superusuário sempre mantém o privilégio ASSUMEROLE.

Para habilitar o uso do privilégio ASSUMEROLE para usuários e grupos, um superusuário executa a instrução a seguir uma vez no cluster. Antes de conceder o privilégio ASSUMEROLE a usuários e grupos, um superusuário deve executar a instrução a seguir uma vez no cluster.

```
revoke assumerole on all from public for all;
```

Observações de uso para revogar permissões de machine learning

Você não pode conceder ou revogar diretamente as permissões relacionadas a uma função de ML. Uma função de ML pertence a um modelo de ML e as permissões são controladas por meio do modelo. Em vez disso, você pode revogar permissões relacionadas ao modelo de ML. O exemplo a seguir demonstra como revogar a permissão de execução para todos os usuários associados ao modelo `customer_churn`.

```
REVOKE EXECUTE ON MODEL customer_churn FROM PUBLIC;
```

Você também pode revogar todas as permissões de um usuário para o modelo de ML `customer_churn`.

```
REVOKE ALL on MODEL customer_churn FROM ml_user;
```

A concessão ou revogação da permissão `EXECUTE` relacionada a uma função de ML falhará se houver uma função de ML no esquema, mesmo que essa função de ML já tenha a permissão `EXECUTE` por meio de `GRANT EXECUTE ON MODEL`. Recomendamos usar um esquema separado ao usar o comando `CREATE MODEL` para manter as funções de ML em um esquema separado. O exemplo a seguir demonstra como fazer isso.

```
CREATE MODEL ml_schema.customer_churn
FROM customer_data
TARGET churn
FUNCTION ml_schema.customer_churn_prediction
IAM_ROLE default
SETTINGS (
 S3_BUCKET 'amzn-s3-demo-bucket'
);
```

Exemplos

O exemplo a seguir revoga os privilégios INSERT em uma tabela SALES do grupo de usuários GUESTS. Esse comando impede que os membros de GUESTS carreguem dados na tabela SALES usando o comando INSERT.

```
revoke insert on table sales from group guests;
```

O exemplo a seguir revoga o privilégio SELECT em todas as tabelas no esquema QA_TICKIT do usuário `fred`.

```
revoke select on all tables in schema qa_tickit from fred;
```

O exemplo revoga o privilégio de selecionar de uma exibição para o usuário `bobr`.

```
revoke select on table eventview from bobr;
```

O exemplo a seguir revoga o privilégio de criar tabelas temporárias no banco de dados TICKIT de todos os usuários.

```
revoke temporary on database tickit from public;
```

O exemplo a seguir revoga o privilégio SELECT nas colunas `cust_name` `cust_phone` e da tabela `cust_profile` do usuário `user1`.

```
revoke select(cust_name, cust_phone) on cust_profile from user1;
```

O exemplo a seguir revoga o privilégio SELECT nas colunas `cust_name` e `cust_phone`, e o privilégio UPDATE na coluna `cust_contact_preference`, da tabela `cust_profile` do grupo `sales_group`.

```
revoke select(cust_name, cust_phone), update(cust_contact_preference) on cust_profile
 from group sales_group;
```

O exemplo a seguir mostra o uso da palavra-chave ALL para revogar privilégios SELECT e UPDATE em três colunas da tabela `cust_profile` do grupo `sales_admin`.

```
revoke ALL(cust_name, cust_phone,cust_contact_preference) on cust_profile from group
 sales_admin;
```

O exemplo a seguir revoga o privilégio SELECT na coluna `cust_name` da exibição `cust_profile_vw` do usuário `user2`.

```
revoke select(cust_name) on cust_profile_vw from user2;
```

Exemplos de revogação da permissão USAGE de bancos de dados criados a partir de unidades de compartilhamento de dados

O exemplo a seguir revoga o acesso à unidade de compartilhamento de dados `salesshare` do namespace `13b8833d-17c6-4f16-8fe4-1a018f5ed00d`.

```
REVOKE USAGE ON DATASHARE salesshare FROM NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

O exemplo a seguir revoga a permissão USAGE no `sales_db` de `Bob`.

```
REVOKE USAGE ON DATABASE sales_db FROM Bob;
```

O exemplo a seguir REVOGA a permissão USAGE no `sales_schema` de `Analyst_role`.

```
REVOKE USAGE ON SCHEMA sales_schema FROM ROLE Analyst_role;
```

Exemplos de revogação de permissões em escopo

O exemplo a seguir revoga uso para todos os esquemas atuais e futuros no banco de dados `Sales_db` da função `Sales`.

```
REVOKE USAGE FOR SCHEMAS IN DATABASE Sales_db FROM ROLE Sales;
```

O exemplo a seguir revoga a capacidade de conceder a permissão SELECT para todas as tabelas atuais e futuras no banco de dados `Sales_db` do usuário `alice`. `alice` mantém acesso a todas as tabelas em `Sales_db`.

```
REVOKE GRANT OPTION SELECT FOR TABLES IN DATABASE Sales_db FROM alice;
```

O exemplo a seguir revoga a permissão EXECUTE para funções no esquema `Sales_schema` do usuário `bob`.

```
REVOKE EXECUTE FOR FUNCTIONS IN SCHEMA Sales_schema FROM bob;
```

O exemplo a seguir revoga todas as permissões para todas as tabelas no esquema do `ShareSchema` do banco de dados `ShareDb` da função `Sales`. Ao especificar o esquema,

você também pode especificar o banco de dados do esquema usando o formato de duas partes `database.schema`.

```
REVOKE ALL FOR TABLES IN SCHEMA ShareDb.ShareSchema FROM ROLE Sales;
```

O exemplo a seguir é o mesmo do anterior. Você pode especificar o banco de dados do esquema usando a palavra-chave `DATABASE`, em vez de usar um formato de duas partes.

```
REVOKE ALL FOR TABLES IN SCHEMA ShareSchema DATABASE ShareDb FROM ROLE Sales;
```

Exemplos de revogação do privilégio ASSUMEROLE

Veja a seguir exemplos de revogação do privilégio ASSUMEROLE.

Um superusuário deve habilitar o uso do privilégio ASSUMEROLE para usuários e grupos executando a seguinte instrução uma vez no cluster:

```
revoke assumerole on all from public for all;
```

A instrução a seguir revoga o privilégio ASSUMEROLE do usuário reg_user1 em todas as funções para todas as operações.

```
revoke assumerole on all from reg_user1 for all;
```

Exemplos de revogação do privilégio ROLE

O exemplo a seguir revoga a função sample_role1 de sample_role2.

```
CREATE ROLE sample_role2;
GRANT ROLE sample_role1 TO ROLE sample_role2;
REVOKE ROLE sample_role1 FROM ROLE sample_role2;
```

O exemplo a seguir revoga os privilégios do sistema do user1.

```
GRANT ROLE sys:DBA TO user1;
REVOKE ROLE sys:DBA FROM user1;
```

O exemplo a seguir revoga sample_role1 e sample_role2 do user1.

```
CREATE ROLE sample_role1;
CREATE ROLE sample_role2;
GRANT ROLE sample_role1, ROLE sample_role2 TO user1;
REVOKE ROLE sample_role1, ROLE sample_role2 FROM user1;
```

O exemplo a seguir revoga sample_role2 com a ADMIN OPTION do user1.

```
GRANT ROLE sample_role2 TO user1 WITH ADMIN OPTION;
REVOKE ADMIN OPTION FOR ROLE sample_role2 FROM user1;
REVOKE ROLE sample_role2 FROM user1;
```

O exemplo a seguir revoga sample_role1 e sample_role2 de sample_role5.

```
CREATE ROLE sample_role5;
GRANT ROLE sample_role1, ROLE sample_role2 TO ROLE sample_role5;
REVOKE ROLE sample_role1, ROLE sample_role2 FROM ROLE sample_role5;
```

O exemplo a seguir revoga os privilégios do sistema CREATE SCHEMA e DROP SCHEMA de sample_role1.

```
GRANT CREATE SCHEMA, DROP SCHEMA TO ROLE sample_role1;
REVOKE CREATE SCHEMA, DROP SCHEMA FROM ROLE sample_role1;
```

ROLLBACK

Interrompe a transação atual e descarta todas as atualizações feitas por essa transação.

Este comando executa a mesma função que o comando ABORT.

Sintaxe

```
ROLLBACK [ WORK | TRANSACTION ]
```

Parâmetros

WORK

Palavra-chave opcional. Essa palavra-chave não é permitida em um procedimento armazenado.

TRANSACTION

Palavra-chave opcional. WORK e TRANSACTION são sinônimos. Nenhuma delas é permitida em um procedimento armazenado.

Para obter informações sobre como usar ROLLBACK em um procedimento armazenado, consulte Gerenciamento de transações.

Exemplo

O exemplo a seguir cria uma tabela, depois inicia uma transação com a inserção de dados na tabela. O comando ROLLBACK então reverte a inserção de dados para deixar a tabela vazia.

O comando a seguir cria uma tabela de exemplo denominada MOVIE_GROSS:

```
create table movie_gross( name varchar(30), gross bigint );
```

O próximo conjunto de comandos inicia uma transação que insere duas linhas de dados na tabela:

```
begin;

insert into movie_gross values ( 'Raiders of the Lost Ark', 23400000);

insert into movie_gross values ( 'Star Wars', 10000000 );
```

Depois, o comando seleciona os dados da tabela para mostrar que eles foram inseridos com êxito:

```
select * from movie_gross;
```

A saída do comando mostra que ambas as linhas foram inseridas com êxito:

```
name           |  gross
-------------------------+----------
Raiders of the Lost Ark | 23400000
Star Wars               | 10000000
(2 rows)
```

Agora este comando reverte as alterações de dados para onde a transação foi iniciada:

```
rollback;
```

Selecionar dados na tabela agora exibe uma tabela vazia:

```
select * from movie_gross;

name | gross
------+-------
(0 rows)
```

SELECT

Retorna linhas de tabelas, exibições e funções definidas pelo usuário.

> **Note**
>
> O tamanho máximo de uma única instrução SQL é 16 MB.

Sintaxe

```
[ WITH with_subquery [, ...] ]
SELECT
[ TOP number | [ ALL | DISTINCT ]
* | expression [ AS output_name ] [, ...] ]
[ EXCLUDE column_list ]
[ FROM table_reference [, ...] ]
[ WHERE condition ]
[ [ START WITH expression ] CONNECT BY expression ]
[ GROUP BY ALL | expression [, ...] ]
[ HAVING condition ]
[ QUALIFY condition ]
[ { UNION | ALL | INTERSECT | EXCEPT | MINUS } query ]
[ ORDER BY expression [ ASC | DESC ] ]
[ LIMIT { number | ALL } ]
[ OFFSET start ]
```

Tópicos

- Cláusula WITH

- Lista SELECT
- EXCLUDE column_list
- Cláusula FROM
- Cláusula WHERE
- Cláusula GROUP BY
- Cláusula HAVING
- Cláusula QUALIFY
- UNION, INTERSECT e EXCEPT
- Cláusula ORDER BY
- Cláusula CONNECT BY
- Exemplos de subconsulta
- Subconsultas correlacionadas

Cláusula WITH

Uma cláusula WITH é uma cláusula opcional que precede a lista SELECT em uma consulta. A cláusula WITH define um ou mais common_table_expressions. Cada expressão de tabela comum (CTE) define uma tabela temporária, que é semelhante à definição de visualização. Você pode fazer referência a essas tabelas temporárias na cláusula FROM. Eles são usados apenas enquanto a consulta a que pertencem é executada. Cada CTE na cláusula WITH especifica um nome de tabela, uma lista opcional de nomes de coluna e uma expressão de consulta que é avaliada como uma tabela (uma instrução SELECT). Quando você faz referência ao nome da tabela temporária na cláusula FROM da mesma expressão de consulta que a define, o CTE é recursivo.

Subconsultas da cláusula WITH são uma forma eficiente de definir tabelas que podem ser usadas ao longo da execução de uma consulta. Em todos os casos, os mesmos resultados podem ser obtidos usando subconsultas no corpo principal da instrução SELECT, mas pode ser mais simples fazer leituras ou gravações de subconsultas da cláusula WITH. Sempre que possível, subconsultas da cláusula WITH por várias vezes referidas são aperfeiçoadas como subexpressões comuns, ou seja, é possível avaliar uma subconsulta WITH uma vez e reutilizar seus resultados. (Observe que subexpressões comuns não estão limitadas àquelas definidas na cláusula WITH.)

Sintaxe

```
[ WITH [RECURSIVE] common_table_expression [, common_table_expression , ...] ]
```

Onde common_table_expression pode ser não recursivo ou recursivo. Segue-se a forma não recursiva:

```
CTE_table_name [ ( column_name [, ...] ) ] AS ( query )
```

Segue-se a forma recursiva de common_table_expression:

```
CTE_table_name (column_name [, ...] ) AS ( recursive_query )
```

Parâmetros

RECURSIVE

Palavra-chave que identifica a consulta como um CTE recursivo. Esta palavra-chave é necessária se qualquer common_table_expression definido na cláusula WITH for recursivo. Você só pode especificar a palavra-chave RECURSIVE uma vez, imediatamente após a palavra-chave WITH, mesmo quando a cláusula WITH contém várias CTEs recursivas. Em geral, um CTE recursivo é uma subconsulta UNION ALL com duas partes.

common_table_expression

Define uma tabela temporária que você pode fazer referência no Cláusula FROM e é usado somente durante a execução da consulta a qual pertence.

CTE_table_name

Um nome exclusivo para uma tabela temporária que define os resultados da subconsulta de cláusula WITH. Você não pode usar nomes duplicados em uma única cláusula WITH. Cada subconsulta deve ter um nome de tabela que pode mencionado em Cláusula FROM.

column_name

Uma lista de nomes de colunas de saída para a subconsulta da cláusula WITH, separados por vírgulas. O número de nomes de coluna especificados deve ser igual ou menor que o número de colunas definido pela subconsulta. Para um CTE que não é recursivo, a cláusula column_name é opcional. Para um CTE recursivo, a lista column_name é necessária.

query

Qualquer consulta SELECT compatível com o Amazon Redshift. Consulte SELECT.

recursive_query

Uma consulta UNION ALL que consiste em duas subconsultas SELECT:

- A primeira subconsulta SELECT não tem uma referência recursiva para o mesmo CTE_table_name. Ele retorna um conjunto de resultados que é a semente inicial da recursão. Esta parte é chamada de membro inicial ou membro semente.
- A segunda subconsulta SELECT faz referência ao mesmo CTE_table_name em sua cláusula FROM. Isso é chamado de membro recursivo. A recursive_query contém uma condição WHERE para finalizar a recursive_query.

Observações de uso

Você pode usar a cláusula WITH nas seguintes instruções SQL:

- SELECT
- SELECT INTO
- CREATE TABLE AS
- CREATE VIEW
- DECLARE
- EXPLAIN
- INSERT INTO...SELECT
- PREPARE
- UPDATE (em uma subconsulta cláusula WHERE não é possível definir um CTE recursivo na subconsulta. O CTE recursivo deve preceder a cláusula UPDATE.)
- DELETE

Se a cláusula FROM de uma consulta que contém a cláusula WITH não fizer referência a qualquer das tabelas definidas pela cláusula WITH, a cláusula WITH será ignorada e a consulta será executada como normal.

Uma tabela definida por uma subconsulta de cláusula WITH somente pode ser referida no escopo da consulta SELECT iniciada pela cláusula WITH. Por exemplo, você pode fazer referência a essa tabela na cláusula FROM da subconsulta na lista SELECT, na cláusula WHERE ou na cláusula HAVING. Você não pode usar a cláusula WITH em uma subconsulta e fazer referência à sua tabela na cláusula FROM da consulta principal ou de outra subconsulta. Este padrão de consulta resulta em uma mensagem de erro do formulário `relation table_name doesn't exist` para a tabela da cláusula WITH.

Você não pode especificar outra cláusula WITH em uma subconsulta de cláusula WITH.

Você não pode fazer referência antecipada a tabelas definidas por subconsultas da cláusula WITH. Por exemplo, a consulta a seguir retorna um erro devido à referência antecipada para a tabela W2 na definição da tabela W1:

```
with w1 as (select * from w2), w2 as (select * from w1)
select * from sales;
ERROR:  relation "w2" does not exist
```

A subconsulta de cláusula WITH pode não consistir em uma instrução SELECT INTO. No entanto, você pode usar uma cláusula WITH em uma instrução SELECT INTO.

Expressões de tabela comuns recursivas

Uma expressão de tabela comum (CTE) recursiva é um CTE que faz referência a si próprio. Um CTE recursivo é útil na consulta de dados hierárquicos, como organogramas que mostram relações de relatório entre funcionários e gerentes. Consulte Exemplo: CTE recursivo.

Outro uso comum é uma lista de materiais multinível, quando um produto consiste em muitos componentes e cada componente também consiste em outros componentes ou submontagens.

Certifique-se de limitar a profundidade da recursão incluindo uma cláusula WHERE na segunda subconsulta SELECT da consulta recursiva. Para ver um exemplo, consulte Exemplo: CTE recursivo. Caso contrário, um erro pode ocorrer semelhante ao seguinte:

- `Recursive CTE out of working buffers.`
- `Exceeded recursive CTE max rows limit, please add correct CTE termination predicates or change the max_recursion_rows parameter.`

> **Note**
>
> `max_recursion_rows` é um parâmetro que define o número máximo de linhas que um CTE recursivo pode retornar para evitar loops de recursão infinita. Não recomendamos alterar esse parâmetro para um valor maior do que o padrão. Isso impede que problemas de recursão infinita em suas consultas ocupem espaço excessivo em seu cluster.

Você pode especificar uma ordem de classificação e limitar o resultado do CTE recursivo. Você pode incluir opções de grupo por e distintas no resultado final do CTE recursivo.

Você não pode especificar outra cláusula WITH em uma subconsulta de cláusula WITH. A recursive_query não pode incluir uma cláusula de ordem por ou limite.

Exemplos

O exemplo a seguir mostra o caso mais simples possível de uma consulta que contém uma cláusula WITH. A consulta WITH com o nome VENUECOPY seleciona todas as linhas da tabela VENUE. Por sua vez, a consulta principal seleciona todas as linhas de VENUECOPY. A tabela VENUECOPY existe somente durante a consulta.

```
with venuecopy as (select * from venue)
select * from venuecopy order by 1 limit 10;
```

```
 venueid |         venuename          |    venuecity    | venuestate | venueseats
---------+----------------------------+-----------------+------------+------------
1 | Toyota Park               | Bridgeview      | IL         |          0
2 | Columbus Crew Stadium     | Columbus        | OH         |          0
3 | RFK Stadium               | Washington      | DC         |          0
4 | CommunityAmerica Ballpark | Kansas City     | KS         |          0
5 | Gillette Stadium          | Foxborough      | MA         |      68756
6 | New York Giants Stadium   | East Rutherford | NJ         |      80242
7 | BMO Field                 | Toronto         | ON         |          0
8 | The Home Depot Center     | Carson          | CA         |          0
9 | Dick's Sporting Goods Park | Commerce City   | CO         |          0
v     10 | Pizza Hut Park             | Frisco          | TX         |          0
(10 rows)
```

O exemplo a seguir mostra uma cláusula WITH que produz duas tabelas, chamadas VENUE_SALES e TOP_VENUES. A segunda tabela de consulta WITH seleciona a partir da primeira. Por sua vez, a cláusula WHERE do bloco principal de consulta contém um subconsulta que restringe a tabela TOP_VENUES.

```
with venue_sales as
(select venuename, venuecity, sum(pricepaid) as venuename_sales
from sales, venue, event
where venue.venueid=event.venueid and event.eventid=sales.eventid
group by venuename, venuecity),

top_venues as
(select venuename
from venue_sales
```

```
where venuename_sales > 800000)

select venuename, venuecity, venuestate,
sum(qtysold) as venue_qty,
sum(pricepaid) as venue_sales
from sales, venue, event
where venue.venueid=event.venueid and event.eventid=sales.eventid
and venuename in(select venuename from top_venues)
group by venuename, venuecity, venuestate
order by venuename;
```

```
        venuename       |   venuecity   | venuestate | venue_qty | venue_sales
------------------------+---------------+------------+-----------+-------------
August Wilson Theatre   | New York City | NY         |      3187 |  1032156.00
Biltmore Theatre        | New York City | NY         |      2629 |   828981.00
Charles Playhouse       | Boston        | MA         |      2502 |   857031.00
Ethel Barrymore Theatre | New York City | NY         |      2828 |   891172.00
Eugene O'Neill Theatre  | New York City | NY         |      2488 |   828950.00
Greek Theatre           | Los Angeles   | CA         |      2445 |   838918.00
Helen Hayes Theatre     | New York City | NY         |      2948 |   978765.00
Hilton Theatre          | New York City | NY         |      2999 |   885686.00
Imperial Theatre        | New York City | NY         |      2702 |   877993.00
Lunt-Fontanne Theatre   | New York City | NY         |      3326 |  1115182.00
Majestic Theatre        | New York City | NY         |      2549 |   894275.00
Nederlander Theatre     | New York City | NY         |      2934 |   936312.00
Pasadena Playhouse      | Pasadena      | CA         |      2739 |   820435.00
Winter Garden Theatre   | New York City | NY         |      2838 |   939257.00
(14 rows)
```

Os dois exemplos a seguir demonstram as regras para o escopo de referências de tabela com base subconsultas da cláusula WITH. A primeira consulta é executada, mas a segunda falha com um erro esperado. A primeira consulta tem a subconsulta de cláusula WITH na lista SELECT da consulta principal. A tabela definida pela cláusula WITH (HOLIDAYS) é referida na cláusula FROM da subconsulta na lista SELECT:

```
select caldate, sum(pricepaid) as daysales,
(with holidays as (select * from date where holiday ='t')
select sum(pricepaid)
from sales join holidays on sales.dateid=holidays.dateid
where caldate='2008-12-25') as dec25sales
from sales join date on sales.dateid=date.dateid
where caldate in('2008-12-25','2008-12-31')
```

```
group by caldate
order by caldate;

caldate   | daysales | dec25sales
----------+----------+------------
2008-12-25 | 70402.00 |   70402.00
2008-12-31 | 12678.00 |   70402.00
(2 rows)
```

A segunda consulta falha porque tenta fazer referência à tabela HOLIDAYS na consulta principal, assim como na subconsulta da lista SELECT. As referências principais da consulta estão fora do escopo.

```
select caldate, sum(pricepaid) as daysales,
(with holidays as (select * from date where holiday ='t')
select sum(pricepaid)
from sales join holidays on sales.dateid=holidays.dateid
where caldate='2008-12-25') as dec25sales
from sales join holidays on sales.dateid=holidays.dateid
where caldate in('2008-12-25','2008-12-31')
group by caldate
order by caldate;

ERROR:  relation "holidays" does not exist
```

Exemplo: CTE recursivo

Veja a seguir um exemplo de um CTE recursivo que retorna os funcionários que respondem direta ou indiretamente a John. A consulta recursiva contém uma cláusula WHERE para limitar a profundidade da recursão a menos de 4 níveis.

```
--create and populate the sample table
  create table employee (
  id int,
  name varchar (20),
  manager_id int
  );

  insert into employee(id, name, manager_id)  values
(100, 'Carlos', null),
(101, 'John', 100),
(102, 'Jorge', 101),
```

```
(103, 'Kwaku', 101),
(110, 'Liu', 101),
(106, 'Mateo', 102),
(110, 'Nikki', 103),
(104, 'Paulo', 103),
(105, 'Richard', 103),
(120, 'Saanvi', 104),
(200, 'Shirley', 104),
(201, 'Sofía', 102),
(205, 'Zhang', 104);

--run the recursive query
  with recursive john_org(id, name, manager_id, level) as
( select id, name, manager_id, 1 as level
  from employee
  where name = 'John'
  union all
  select e.id, e.name, e.manager_id, level + 1 as next_level
  from employee e, john_org j
  where e.manager_id = j.id and level < 4
  )
 select distinct id, name, manager_id from john_org order by manager_id;
```

A seguir é o resultado da consulta.

```
   id        name      manager_id
  ------+-----------+--------------
   101    John           100
   102    Jorge          101
   103    Kwaku          101
   110    Liu            101
   201    Sofía          102
   106    Mateo          102
   110    Nikki          103
   104    Paulo          103
   105    Richard        103
   120    Saanvi         104
   200    Shirley        104
   205    Zhang          104
```

A seguir está um organograma para o departamento de John.

Lista SELECT

Tópicos

- Sintaxe
- Parâmetros
- Observações de uso
- Exemplos

A lista SELECT nomeia as colunas, funções e expressões que você deseja que a consulta retorne. A lista representa o resultado da consulta.

Para obter mais informações sobre funções SQL, consulte Referência de funções SQL. Para obter mais informações sobre expressões, consulte Expressões condicionais.

Sintaxe

```
SELECT
[ TOP number ]
[ ALL | DISTINCT ] * | expression [ AS column_alias ] [, ...]
```

Parâmetros

TOP número

TOP pega um inteiro positivo como argumento, que define o número de linhas retornadas para o cliente. O comportamento da cláusula TOP é o mesmo da cláusula LIMIT. O número de linhas retornado é fixo, mas o conjunto de linhas não. Para retornar um conjunto consistente de linhas, use TOP ou LIMIT em conjunto com uma cláusula ORDER BY.

ALL

Palavra-chave redundante que define o comportamento padrão se você não especificar DISTINCT. `SELECT ALL *` é o mesmo que `SELECT *` (seleciona todas as linhas para todas as colunas e retém duplicações).

DISTINCT

Opção que elimina linhas duplicadas do conjunto de resultados, com base em valores correspondentes em uma ou mais colunas.

> **Note**
>
> Se sua aplicação permitir chaves primárias ou chaves estrangeiras inválidas, isso pode fazer com que algumas consultas retornem resultados incorretos. Por exemplo, uma consulta SELECT DISTINCT pode retornar linhas duplicadas se a coluna chave primária não contiver todos os valores exclusivos. Para obter mais informações, consulte Definir restrições de tabela.

* (asterisco)

Retorna o conteúdo total da tabela (todas as colunas e todas as linhas).

expressão

Expressão formada por uma ou mais colunas que existem em tabelas referidas pela consulta. Uma expressão pode conter funções SQL. Por exemplo:

```
avg(datediff(day, listtime, saletime))
```

AS alias_coluna

Nome temporário da coluna que é usada no conjunto de resultados finais. A palavra-chave AS é opcional. Por exemplo:

```
avg(datediff(day, listtime, saletime)) as avgwait
```

Se você não especificar um alias para uma expressão que não for um nome de coluna simples, o resultado definido aplicará um nome padrão à coluna.

> **Note**
>
> O alias é reconhecido logo após ser definido na lista de destino. É possível usar um alias em outras expressões definidas depois dele na mesma lista de destino. Isso é ilustrado no exemplo a seguir.

```
select clicks / impressions as probability, round(100 * probability, 1) as
 percentage from raw_data;
```

O benefício da referência do alias lateral é que você não precisa repetir a expressão usada como alias ao criar expressões mais complexas na mesma lista de destino. Quando o Amazon Redshift analisa esse tipo de referência, ele apenas alinha os aliases definidos anteriormente. Se houver uma coluna com o mesmo nome definido na cláusula FROM como a expressão usada como alias anteriormente, a coluna na cláusula FROM terá prioridade. Por exemplo, se na consulta acima houver uma coluna chamada “probabilidade” na tabela raw_data, a “probabilidade” na segunda expressão da lista de destino faz referência àquela coluna, em vez do nome do alias “probabilidade”.

Observações de uso

TOP é uma extensão SQL que fornece uma alternativa ao comportamento de LIMIT. Você não pode usar TOP e LIMIT na mesma consulta.

Exemplos

O exemplo a seguir retorna dez linhas da tabela SALES. Embora a consulta use a cláusula TOP, ela ainda retorna um conjunto imprevisível de linhas porque nenhuma cláusula ORDER BY foi especificada,

```
select top 10 *
from sales;
```

A consulta a seguir é funcionalmente equivalente, mas usa uma cláusula LIMIT em vez de uma cláusula TOP:

```
select *
from sales
limit 10;
```

O exemplo a seguir retorna as dez primeiras linhas da tabela SALES usando a cláusula TOP, classificadas pela coluna QTYSOLD em ordem decrescente.

```
select top 10 qtysold, sellerid
from sales
order by qtysold desc, sellerid;
```

```
qtysold | sellerid
--------+----------
8 |      518
8 |      520
8 |      574
8 |      718
8 |      868
8 |     2663
8 |     3396
8 |     3726
8 |     5250
8 |     6216
(10 rows)
```

O exemplo a seguir retorna os dois primeiros valores de QTYSOLD e SELLERID da tabela SALES, classificados pela coluna QTYSOLD:

```
select top 2 qtysold, sellerid
from sales
order by qtysold desc, sellerid;

qtysold | sellerid
--------+----------
8 |      518
8 |      520
(2 rows)
```

O exemplo a seguir mostra a lista de grupos de categorias distintos da tabela CATEGORY:

```
select distinct catgroup from category
order by 1;

catgroup
----------
Concerts
Shows
Sports
(3 rows)

--the same query, run without distinct
select catgroup from category
order by 1;
```

```
catgroup
----------
Concerts
Concerts
Concerts
Shows
Shows
Shows
Sports
Sports
Sports
Sports
Sports
(11 rows)
```

O exemplo a seguir retorna o conjunto distinto de números da semana de dezembro de 2008. Sem a cláusula DISTINCT, a declaração retornaria 31 linhas, ou uma para cada dia do mês.

```
select distinct week, month, year
from date
where month='DEC' and year=2008
order by 1, 2, 3;

week | month | year
-----+-------+------
49 | DEC   | 2008
50 | DEC   | 2008
51 | DEC   | 2008
52 | DEC   | 2008
53 | DEC   | 2008
(5 rows)
```

EXCLUDE column_list

O EXCLUDE column_list nomeia as colunas excluídas dos resultados da consulta. Usar a opção EXCLUDE é útil quando somente um subconjunto de colunas precisa ser excluído de uma tabela ampla, que é uma tabela que contém muitas colunas.

Tópicos

- Sintaxe

- Parâmetros
- Exemplos

Sintaxe

```
EXCLUDE column_list
```

Parâmetros

column_list

Uma lista separada por vírgulas de um ou mais nomes de coluna que existem nas tabelas usadas como referência pela consulta. A column_list pode ser colocada entre parênteses. Somente nomes de coluna são aceitos na lista de exclusão de nomes de coluna, não expressões, como `upper(col1)`, ou asterisco (*).

```
column-name, ... | ( column-name, ... )
```

Por exemplo:

```
SELECT * EXCLUDE col1, col2 FROM tablea;
```

```
SELECT * EXCLUDE (col1, col2) FROM tablea;
```

Exemplos

Os exemplos a seguir usam a tabela SALES que contém as colunas: salesid, listid, sellerid, buyerid, eventid, dateid, qtysold, pricepaid, commission e saletime. Para ter mais informações sobre a tabela SALES, consulte Banco de dados de exemplo.

O exemplo a seguir exibe linhas da tabela SALES, mas exclui a coluna SALETIME.

```
SELECT * EXCLUDE saletime FROM sales;

salesid | listid  | sellerid | buyerid | eventid | dateid  | qtysold  | pricepaid  |
 commission
--------+---------+----------+---------+---------+---------+----------+------------
+-----------
```

```
150314  | 173969  | 48680    | 816     | 8762    | 1827    | 2        | 688        |
 103.2
8325    | 8942    | 23600    | 1078    | 2557    | 1828    | 5        | 525        |
 78.75
46807   | 52711   | 34388    | 1047    | 2046    | 1828    | 2        | 482        |
 72.3
...
```

O exemplo a seguir exibe linhas da tabela SALES, mas exclui as colunas QTYSOLD e SALETIME.

```
SELECT * EXCLUDE (qtysold, saletime) FROM sales;

salesid | listid  | sellerid | buyerid | eventid | dateid  | pricepaid  | commission
--------+---------+----------+---------+---------+---------+------------+-----------
150314  | 173969  | 48680    | 816     | 8762    | 1827    | 688        | 103.2
8325    | 8942    | 23600    | 1078    | 2557    | 1828    | 525        |  78.75
46807   | 52711   | 34388    | 1047    | 2046    | 1828    | 482        |  72.3
...
```

O exemplo a seguir cria uma visão que exibe linhas da tabela SALES, mas exclui a coluna SALETIME.

```
CREATE VIEW sales_view AS SELECT * EXCLUDE saletime FROM sales;
SELECT * FROM sales_view;

salesid | listid  | sellerid | buyerid | eventid | dateid  | qtysold  | pricepaid  |
 commission
--------+---------+----------+---------+---------+---------+----------+------------
+-----------
150314  | 173969  | 48680    | 816     | 8762    | 1827    | 2        | 688        |
 103.2
8325    | 8942    | 23600    | 1078    | 2557    | 1828    | 5        | 525        |
 78.75
46807   | 52711   | 34388    | 1047    | 2046    | 1828    | 2        | 482        |
 72.3
...
```

O exemplo a seguir seleciona somente as colunas não excluídas em uma tabela temporária.

```
SELECT * EXCLUDE saletime INTO TEMP temp_sales FROM sales;
SELECT * FROM temp_sales;
```

```
salesid | listid  | sellerid | buyerid | eventid | dateid  | qtysold  | pricepaid  |
 commission
--------+---------+----------+---------+---------+---------+----------+------------
+-----------
150314  | 173969  | 48680    | 816     | 8762    | 1827    | 2        | 688        |
 103.2
8325    | 8942    | 23600    | 1078    | 2557    | 1828    | 5        | 525        |
 78.75
46807   | 52711   | 34388    | 1047    | 2046    | 1828    | 2        | 482        |
 72.3
...
```

Cláusula FROM

A cláusula FROM em uma consulta lista as referências de tabela (tabelas, exibições e subconsultas) de onde os dados são selecionados. Se as referências de várias tabelas estiverem listadas, as tabelas devem ser juntadas, usando a sintaxe apropriada na cláusula FROM ou WHERE. Se nenhum critério de junção for especificado, o sistema processará a consulta como uma junção cruzada (produto cartesiano).

Tópicos

- Sintaxe
- Parâmetros
- Observações de uso
- Exemplos de PIVOT e UNPIVOT
- Exemplos de JOIN
- Exemplos de UNNEST

Sintaxe

```
FROM table_reference [, ...]
```

onde referência_tabela é uma das seguintes:

```
with_subquery_table_name [ table_alias ]
table_name [ * ] [ table_alias ]
( subquery ) [ table_alias ]
table_reference [ NATURAL ] join_type table_reference
```

```
   [ ON join_condition | USING ( join_column [, ...] ) ]
table_reference  join_type super_expression
   [ ON join_condition ]
table_reference PIVOT (
   aggregate(expr) [ [ AS ] aggregate_alias ]
   FOR column_name IN ( expression [ AS ] in_alias [, ...] )
) [ table_alias ]
table_reference UNPIVOT [ INCLUDE NULLS | EXCLUDE NULLS ] (
   value_column_name
   FOR name_column_name IN ( column_reference [ [ AS ]
   in_alias ] [, ...] )
) [ table_alias ]
UNPIVOT expression AS value_alias [ AT attribute_alias ]
( super_expression.attribute_name ) AS value_alias [ AT index_alias ]
UNNEST ( column_reference )
  [AS] table_alias ( unnested_column_name )
UNNEST ( column_reference ) WITH OFFSET
  [AS] table_alias ( unnested_column_name, [offset_column_name] )
```

O table_alias opcional pode ser usado para fornecer nomes temporários a tabelas e referências de tabelas complexas e, se desejado, também às respectivas colunas, da forma a seguir:

```
[ AS ] alias [ ( column_alias [, ...] ) ]
```

Parâmetros

com_subconsulta_nome_tabela

Tabela definida por uma subconsulta em Cláusula WITH.

table_name

Nome de uma tabela ou exibição.

alias

Nome alternativo temporário para uma tabela ou exibição. Um alias deve ser fornecido para uma tabela derivada de uma subconsulta. Em outras referências de tabela, os alias são opcionais. A palavra-chave AS é sempre opcional. Os alias de tabela oferecem um atalho conveniente para tabelas de identificação em outras partes de uma consulta, como a cláusula WHERE. Por exemplo:

```
select * from sales s, listing l
```

```
where s.listid=l.listid
```

column_alias

Nome alternativo temporário para uma coluna em uma tabela ou exibição.

subconsulta

Uma expressão de consulta que avalia para uma tabela. A tabela existe somente pela duração da consulta e geralmente recebe um nome ou alias. No entanto, um alias não é necessário. Você também pode definir nomes de colunas para tabelas que derivam de subconsultas. Nomear aliases de coluna é importante quando você deseja participar dos resultados de subconsultas a outras tabelas e quando você deseja selecionar ou restringir essas colunas em outro lugar da consulta.

Uma subconsulta pode conter uma cláusula ORDER BY, mas essa cláusula poderá não ter qualquer efeito se uma cláusula LIMIT ou OFFSET também não estiver especificada.

NATURAL

Define um junção que usa automaticamente todos os pares de colunas com nomes idênticos em duas tabelas como colunas de junção. Nenhuma condição explícita de junção é necessária. Por exemplo, se as tabelas CATEGORY e EVENT apresentam colunas com nome CATID, um junção natural dessas tabelas é um junção pelas colunas CATID.

> **Note**
>
> Se uma junção NATURAL for especificada mas não existirem pares de colunas com o mesmo nome nas tabelas a serem juntadas, a junção padrão da consulta usada será a junção cruzada.

join_type

Especifique um dos seguintes tipos de junção:

- [INNER] JOIN
- LEFT [OUTER] JOIN
- RIGHT [OUTER] JOIN
- FULL [OUTER] JOIN
- CROSS JOIN

As junções cruzadas são junções não qualificadas; elas retornam o produto cartesiano das duas tabelas.

As junções internas e externas são junções qualificadas. Elas podem ser qualificadas implicitamente (em junções naturais); com a sintaxe ON ou USING na cláusula FROM; ou com a condição de cláusula WHERE.

Uma junção interna retorna somente linhas correspondentes, com base na condição de junção ou na lista de colunas de junção. Uma junção externa retorna todas as linhas que a junção interna equivalente deve retornar e linhas não correspondentes da tabela "esquerda", da tabela "direita" ou de ambas. A tabela esquerda é a primeira tabela listada, e a tabela direita é a segunda tabela listada. As linhas não correspondentes contêm valores NULL para preencher lacunas entre as colunas resultantes.

ON condição_junção

Tipo de especificação de junção em que as colunas a serem juntadas são exibidas como uma condição que acompanha a palavra-chave ON. Por exemplo:

```
sales join listing
on sales.listid=listing.listid and sales.eventid=listing.eventid
```

USING (coluna_junção [, ...])

Tipo de especificação de junção em que as colunas a serem juntadas estão listadas entre parênteses. Se várias colunas a serem juntadas forem especificadas, elas serão separadas por vírgulas. A palavra-chave USING deve preceder a lista. Por exemplo:

```
sales join listing
using (listid,eventid)
```

PIVOT

Alterna a saída de linhas para colunas, com a finalidade de representar dados tabulares em um formato de fácil leitura. A saída é representada horizontalmente em várias colunas. PIVOT é semelhante a uma consulta GROUP BY com uma agregação, usando uma expressão agregada para especificar um formato de saída. Porém, diferente de GROUP BY, os resultados são retornados em colunas em vez de linhas.

Para obter exemplos que mostrem como consultar com PIVOT e UNPIVOT, consulte Exemplos de PIVOT e UNPIVOT.

UNPIVOT

Transformar colunas em linhas com UNPIVOT: o operador transforma as colunas de resultados de uma tabela de entrada ou os resultados de uma consulta em linhas, para facilitar a leitura da saída. UNPIVOT combina os dados das colunas de entrada em duas colunas de resultado: uma coluna de nome e uma coluna de valor. A coluna name contém nomes de coluna da entrada, como entradas de linha. A coluna value contém valores das colunas de entrada, como resultados de uma agregação. Por exemplo, as contagens de itens em várias categorias.

Desagregar objetos com UNPIVOT (SUPER): é possível desagregar objetos; nesse caso, a expressão é uma expressão SUPER referente a outro item da cláusula FROM. Para obter mais informações, consulte Transformar colunas em linhas de objetos. Também há exemplos que mostram como consultar dados semiestruturados, como dados formatados em JSON.

super_expression

Uma expressão SUPER válida. O Amazon Redshift exibe uma linha para cada valor no atributo especificado. Para ter mais informações sobre o tipo de dado SUPER, consulte Tipo SUPER. Para ter mais informações sobre valores SUPER desaninhados, consulte Desaninhar consultas.

attribute_name

O nome de um atributo na expressão SUPER.

index_alias

Alias para o índice que significa a posição do valor na expressão SUPER.

UNNEST

Expande uma estrutura aninhada, normalmente uma matriz SUPER, em colunas que contêm os elementos não aninhados. Para ter mais informações sobre desagrupamento de dados SUPER, consulte Consultar dados semiestruturados. Para obter exemplos, consulte Exemplos de UNNEST.

unnested_column_name

O nome da coluna que contém os elementos não aninhados.

UNNEST ... WITH OFFSET

Adiciona uma coluna de deslocamento à saída não aninhada, e o deslocamento representa o índice baseado em zero de cada elemento na matriz. Essa variante é útil quando você deseja ver a posição dos elementos em uma matriz. Para ter mais informações sobre desagrupamento

de dados SUPER, consulte Consultar dados semiestruturados. Para obter exemplos, consulte Exemplos de UNNEST.

offset_column_name

Um nome personalizado para a coluna de deslocamento que você usa para definir explicitamente como a coluna de índice aparecerá na saída. Esse parâmetro é opcional. Por padrão, o nome da coluna de deslocamento é `offset_col`.

Observações de uso

Colunas de junção devem ter tipos de dados comparáveis.

Uma junção NATURAL ou USING retém somente um de cada par de colunas de junção no conjunto de resultados intermediário.

Uma junção com a sintaxe ON retém ambas as colunas de junção em seu conjunto de resultados intermediário.

Consulte também Cláusula WITH.

Exemplos de PIVOT e UNPIVOT

PIVOT e UNPIVOT são parâmetros na cláusula FROM que trocam a saída da consulta de linhas para colunas e colunas para linhas, respectivamente. Eles representam resultados de consultas tabulares em um formato fácil de ler. Os exemplos a seguir usam consultas e dados de teste para mostrar como usá-los.

Para obter mais informações sobre esses parâmetros, consulte FROM clause.

Exemplos de PIVOT

Configure a tabela e os dados de exemplo e use-os para executar as consultas de exemplo subsequentes.

```
CREATE TABLE part (
    partname varchar,
    manufacturer varchar,
    quality int,
    price decimal(12, 2)
);
```

```
INSERT INTO part VALUES ('prop', 'local parts co', 2, 10.00);
INSERT INTO part VALUES ('prop', 'big parts co', NULL, 9.00);
INSERT INTO part VALUES ('prop', 'small parts co', 1, 12.00);

INSERT INTO part VALUES ('rudder', 'local parts co', 1, 2.50);
INSERT INTO part VALUES ('rudder', 'big parts co', 2, 3.75);
INSERT INTO part VALUES ('rudder', 'small parts co', NULL, 1.90);

INSERT INTO part VALUES ('wing', 'local parts co', NULL, 7.50);
INSERT INTO part VALUES ('wing', 'big parts co', 1, 15.20);
INSERT INTO part VALUES ('wing', 'small parts co', NULL, 11.80);
```

PIVOT em `partname` com um agregação de `AVG` em `price`.

```
SELECT *
FROM (SELECT partname, price FROM part) PIVOT (
    AVG(price) FOR partname IN ('prop', 'rudder', 'wing')
);
```

A consulta resulta na saída a seguir.

```
  prop   |  rudder  |  wing
---------+----------+---------
 10.33   | 2.71     | 11.50
```

No exemplo anterior, os resultados são transformados em colunas. O exemplo a seguir mostra uma consulta `GROUP BY` que retorna os preços médios em linhas, em vez de em colunas.

```
SELECT partname, avg(price)
FROM (SELECT partname, price FROM part)
WHERE partname IN ('prop', 'rudder', 'wing')
GROUP BY partname;
```

A consulta resulta na saída a seguir.

```
 partname |  avg
----------+-------
 prop     | 10.33
 rudder   |  2.71
 wing     | 11.50
```

Um exemplo de PIVOT com manufacturer como uma coluna implícita.

```
SELECT *
FROM (SELECT quality, manufacturer FROM part) PIVOT (
    count(*) FOR quality IN (1, 2, NULL)
);
```

A consulta resulta na saída a seguir.

```
 manufacturer      | 1  | 2  | null
-------------------+----+----+------
 local parts co    | 1  | 1  |  1
 big parts co      | 1  | 1  |  1
 small parts co    | 1  | 0  |  2
```

Colunas da tabela de entrada que não são referenciadas na definição PIVOT são adicionadas implicitamente à tabela de resultados. Este é o caso da coluna manufacturer no exemplo anterior. O exemplo também mostra que NULL é um valor válido para o operador IN.

PIVOT no exemplo acima retorna informações semelhantes à consulta a seguir, que inclui GROUP BY. A diferença é que PIVOT retorna o valor 0 para a coluna 2 e o fabricante small parts co. A consulta GROUP BY não contém uma linha correspondente. Na maioria dos casos, PIVOT insere NULL se uma linha não tem dados de entrada para determinada coluna. Porém, o agregado de contagem não retorna NULL e 0 é o valor padrão.

```
SELECT manufacturer, quality, count(*)
FROM (SELECT quality, manufacturer FROM part)
WHERE quality IN (1, 2) OR quality IS NULL
GROUP BY manufacturer, quality
ORDER BY manufacturer;
```

A consulta resulta na saída a seguir.

```
 manufacturer        | quality | count
---------------------+---------+-------
 big parts co        |         |     1
 big parts co        |       2 |     1
 big parts co        |       1 |     1
 local parts co      |       2 |     1
 local parts co      |       1 |     1
```

```
 local parts co      |         |     1
 small parts co      |       1 |     1
 small parts co      |         |     2
```

O operador PIVOT aceita aliases opcionais na expressão agregada e em cada valor para o operador `IN`. Use aliases para personalizar os nomes das colunas. Se não houver um alias agregado, somente os aliases da lista `IN` serão usados. Caso contrário, o alias agregado será anexado ao nome da coluna com um sublinhado para separar os nomes.

```
SELECT *
FROM (SELECT quality, manufacturer FROM part) PIVOT (
    count(*) AS count FOR quality IN (1 AS high, 2 AS low, NULL AS na)
);
```

A consulta resulta na saída a seguir.

```
 manufacturer      | high_count  | low_count | na_count
-------------------+-------------+-----------+----------
 local parts co    |           1 |         1 |        1
 big parts co      |           1 |         1 |        1
 small parts co    |           1 |         0 |        2
```

Configure a tabela e os dados de exemplo a seguir e use-os para executar as consultas de exemplo subsequentes. Os dados representam datas de reserva para um grupo de hotéis.

```
CREATE TABLE bookings (
    booking_id int,
    hotel_code char(8),
    booking_date date,
    price decimal(12, 2)
);

INSERT INTO bookings VALUES (1, 'FOREST_L', '02/01/2023', 75.12);
INSERT INTO bookings VALUES (2, 'FOREST_L', '02/02/2023', 75.00);
INSERT INTO bookings VALUES (3, 'FOREST_L', '02/04/2023', 85.54);

INSERT INTO bookings VALUES (4, 'FOREST_L', '02/08/2023', 75.00);
INSERT INTO bookings VALUES (5, 'FOREST_L', '02/11/2023', 75.00);
INSERT INTO bookings VALUES (6, 'FOREST_L', '02/14/2023', 90.00);

INSERT INTO bookings VALUES (7, 'FOREST_L', '02/21/2023', 60.00);
```

```
INSERT INTO bookings VALUES (8, 'FOREST_L', '02/22/2023', 85.00);
INSERT INTO bookings VALUES (9, 'FOREST_L', '02/27/2023', 90.00);

INSERT INTO bookings VALUES (10, 'DESERT_S', '02/01/2023', 98.00);
INSERT INTO bookings VALUES (11, 'DESERT_S', '02/02/2023', 75.00);
INSERT INTO bookings VALUES (12, 'DESERT_S', '02/04/2023', 85.00);

INSERT INTO bookings VALUES (13, 'DESERT_S', '02/05/2023', 75.00);
INSERT INTO bookings VALUES (14, 'DESERT_S', '02/06/2023', 34.00);
INSERT INTO bookings VALUES (15, 'DESERT_S', '02/09/2023', 85.00);

INSERT INTO bookings VALUES (16, 'DESERT_S', '02/12/2023', 23.00);
INSERT INTO bookings VALUES (17, 'DESERT_S', '02/13/2023', 76.00);
INSERT INTO bookings VALUES (18, 'DESERT_S', '02/14/2023', 85.00);

INSERT INTO bookings VALUES (19, 'OCEAN_WV', '02/01/2023', 98.00);
INSERT INTO bookings VALUES (20, 'OCEAN_WV', '02/02/2023', 75.00);
INSERT INTO bookings VALUES (21, 'OCEAN_WV', '02/04/2023', 85.00);

INSERT INTO bookings VALUES (22, 'OCEAN_WV', '02/06/2023', 75.00);
INSERT INTO bookings VALUES (23, 'OCEAN_WV', '02/09/2023', 34.00);
INSERT INTO bookings VALUES (24, 'OCEAN_WV', '02/12/2023', 85.00);

INSERT INTO bookings VALUES (25, 'OCEAN_WV', '02/13/2023', 23.00);
INSERT INTO bookings VALUES (26, 'OCEAN_WV', '02/14/2023', 76.00);
INSERT INTO bookings VALUES (27, 'OCEAN_WV', '02/16/2023', 85.00);

INSERT INTO bookings VALUES (28, 'CITY_BLD', '02/01/2023', 98.00);
INSERT INTO bookings VALUES (29, 'CITY_BLD', '02/02/2023', 75.00);
INSERT INTO bookings VALUES (30, 'CITY_BLD', '02/04/2023', 85.00);

INSERT INTO bookings VALUES (31, 'CITY_BLD', '02/12/2023', 75.00);
INSERT INTO bookings VALUES (32, 'CITY_BLD', '02/13/2023', 34.00);
INSERT INTO bookings VALUES (33, 'CITY_BLD', '02/17/2023', 85.00);

INSERT INTO bookings VALUES (34, 'CITY_BLD', '02/22/2023', 23.00);
INSERT INTO bookings VALUES (35, 'CITY_BLD', '02/23/2023', 76.00);
INSERT INTO bookings VALUES (36, 'CITY_BLD', '02/24/2023', 85.00);
```

Nesse exemplo de consulta, os registros de reservas são calculados para fornecer um total de cada semana. A data de término de cada semana se torna um nome de coluna.

```
SELECT * FROM
```

```
    (SELECT
       booking_id,
       (date_trunc('week', booking_date::date) + '5 days'::interval)::date as enddate,
       hotel_code AS "hotel code"
FROM bookings
) PIVOT (
    count(booking_id) FOR enddate IN ('2023-02-04','2023-02-11','2023-02-18')
);
```

A consulta resulta na saída a seguir.

```
 hotel code | 2023-02-04  | 2023-02-11 | 2023-02-18
------------+-------------+------------+----------
 FOREST_L   |           3 |          2 |        1
 DESERT_S   |           4 |          3 |        2
 OCEAN_WV   |           3 |          3 |        3
 CITY_BLD   |           3 |          1 |        2
```

O Amazon Redshift não é compatível com CROSSTAB para girar em várias colunas. No entanto, é possível alterar dados de linha em colunas, de forma semelhante a uma agregação com PIVOT, com uma consulta como a seguinte. Isso usa os mesmos dados de exemplo de reserva como no exemplo anterior.

```
SELECT
  booking_date,
  MAX(CASE WHEN hotel_code = 'FOREST_L' THEN 'forest is booked' ELSE '' END) AS
 FOREST_L,
  MAX(CASE WHEN hotel_code = 'DESERT_S' THEN 'desert is booked' ELSE '' END) AS
 DESERT_S,
  MAX(CASE WHEN hotel_code = 'OCEAN_WV' THEN 'ocean is booked' ELSE '' END)  AS
 OCEAN_WV
FROM bookings
GROUP BY booking_date
ORDER BY booking_date asc;
```

O exemplo de consulta resulta em datas de reserva listadas ao lado de frases curtas que indicam quais hotéis estão reservados.

```
 booking_date  | forest_l         | desert_s         | ocean_wv
---------------+------------------+------------------+--------------------
 2023-02-01    | forest is booked | desert is booked |  ocean is booked
```

```
 2023-02-02    | forest is booked | desert is booked |  ocean is booked
 2023-02-04    | forest is booked | desert is booked |  ocean is booked
 2023-02-05    |                  | desert is booked |
 2023-02-06    |                  | desert is booked |
```

Veja a seguir as observações de uso do PIVOT:

- PIVOT pode ser aplicado a tabelas, subconsultas e expressões de tabela comuns (CTEs). PIVOT não pode ser aplicado a expressões JOIN, CTEs recursivos, PIVOT ou expressões UNPIVOT. Também não são compatíveis expressões SUPER não aninhadas e tabelas aninhadas do Redshift Spectrum.
- PIVOT é compatível com funções agregadas COUNT, SUM, MIN, MAX e AVG.
- A expressão agregada PIVOT deve ser uma chamada de uma função agregada compatível. Expressões complexas na parte superior do agregado não são compatíveis. Os argumentos agregados não podem conter referências a tabelas diferentes da tabela de entrada do PIVOT. Referências correlacionadas a uma consulta principal também não são compatíveis. O argumento agregado pode conter subconsultas. Elas podem ser correlacionadas internamente ou na tabela de entrada PIVOT.
- Os valores da lista PIVOT IN não podem ser referências de coluna ou subconsultas. Cada valor deve ser compatível com a referência de coluna FOR.
- Se os valores de lista IN não tiverem aliases, PIVOT gerará nomes de coluna padrão. Por valores IN constantes, como 'abc' ou 5, o nome da coluna padrão é a constante em si. Para qualquer expressão complexa, o nome da coluna é um nome padrão do Amazon Redshift, como ?column?.

Exemplos de UNPIVOT

Configure os dados de exemplo e use-os para executar os exemplos subsequentes.

```
CREATE TABLE count_by_color (quality varchar, red int, green int, blue int);

INSERT INTO count_by_color VALUES ('high', 15, 20, 7);
INSERT INTO count_by_color VALUES ('normal', 35, NULL, 40);
INSERT INTO count_by_color VALUES ('low', 10, 23, NULL);
```

UNPIVOT nas colunas de entrada vermelho, verde e azul.

```
SELECT *
```

```
FROM (SELECT red, green, blue FROM count_by_color) UNPIVOT (
    cnt FOR color IN (red, green, blue)
);
```

A consulta resulta na saída a seguir.

```
 color | cnt
-------+-----
 red   |  15
 red   |  35
 red   |  10
 green |  20
 green |  23
 blue  |   7
 blue  |  40
```

Por padrão, os valores NULL na coluna de entrada são ignorados e não produzem uma linha de resultado.

O exemplo a seguir mostra UNPIVOT com INCLUDE NULLS.

```
SELECT *
FROM (
    SELECT red, green, blue
    FROM count_by_color
) UNPIVOT INCLUDE NULLS (
    cnt FOR color IN (red, green, blue)
);
```

A seguir está a saída resultante.

```
 color | cnt
-------+-----
 red   |  15
 red   |  35
 red   |  10
 green |  20
 green |
 green |  23
 blue  |   7
 blue  |  40
```

```
 blue  |
```

Se o parâmetro `INCLUDING NULLS` estiver definido, os valores de entrada `NULL` geram linhas de resultados.

`The following query shows` `UNPIVOT` com `quality` como uma coluna implícita.

```
SELECT *
FROM count_by_color UNPIVOT (
    cnt FOR color IN (red, green, blue)
);
```

A consulta resulta na saída a seguir.

```
 quality | color | cnt
---------+-------+-----
 high    | red   |  15
 normal  | red   |  35
 low     | red   |  10
 high    | green |  20
 low     | green |  23
 high    | blue  |   7
 normal  | blue  |  40
```

Colunas da tabela de entrada que não são referenciadas na definição `UNPIVOT` são adicionadas implicitamente à tabela de resultados. No exemplo, este é o caso da coluna `quality`.

O exemplo a seguir mostra `UNPIVOT` com aliases para valores na lista `IN`.

```
SELECT *
FROM count_by_color UNPIVOT (
    cnt FOR color IN (red AS r, green AS g, blue AS b)
);
```

A consulta anterior resulta na saída a seguir.

```
 quality | color | cnt
---------+-------+-----
 high    | r     |  15
 normal  | r     |  35
```

```
low     | r     |  10
high    | g     |  20
low     | g     |  23
high    | b     |   7
normal  | b     |  40
```

O operador UNPIVOT aceita aliases opcionais em cada valor de lista IN. Cada alias fornece personalização dos dados em cada coluna value.

Veja a seguir as observações de uso do UNPIVOT.

- UNPIVOT pode ser aplicado a tabelas, subconsultas e expressões de tabela comuns (CTEs). UNPIVOT não pode ser aplicado a expressões JOIN, CTEs recursivos, PIVOT ou expressões UNPIVOT. Também não são compatíveis expressões SUPER não aninhadas e tabelas aninhadas do Redshift Spectrum.
- A lista UNPIVOT IN deve conter apenas referências de coluna da tabela de entrada. As colunas da lista IN devem ter um tipo comum com o qual todas sejam compatíveis. A coluna de valor UNPIVOT tem esse tipo comum. A coluna de nome UNPIVOT é do tipo VARCHAR.
- Se um valor de lista IN não tiver um alias, UNPIVOT usará o nome da coluna como valor padrão.

Exemplos de JOIN

Uma cláusula SQL JOIN é usada para combinar os dados de duas ou mais tabelas com base em campos comuns. Os resultados podem ou não mudar dependendo do método de junção especificado. Para obter mais informações sobre a sintaxe da cláusula JOIN, consulte Parâmetros.

O exemplo a seguir usa dados dos dados de amostra TICKIT. Para obter mais informações sobre o esquema de banco de dados, consulte Banco de dados de exemplo. Para saber como carregar dados de exemplo, consulte Carregamento de dados no Guia de conceitos básicos do Amazon Redshift.

A consulta a seguir é uma junção interna (sem a palavra-chave JOIN) entre a tabela LISTING e a tabela SALES, onde o LISTID da tabela LISTING está entre 1 e 5. Essa consulta corresponde aos valores da coluna LISTID na tabela LISTING (a tabela à esquerda) e na tabela SALES (tabela à direita). Os resultados mostram que LISTID 1, 4 e 5 correspondem aos critérios.

```
select listing.listid, sum(pricepaid) as price, sum(commission) as comm
from listing, sales
where listing.listid = sales.listid
```

```
and listing.listid between 1 and 5
group by 1
order by 1;

listid | price  |  comm
-------+--------+--------
     1 | 728.00 | 109.20
     4 |  76.00 |  11.40
     5 | 525.00 |  78.75
```

A consulta a seguir é uma junção externa à esquerda. Junções externas esquerdas e direitas retêm valores de uma das tabelas de junção quando nenhuma correspondência é encontrada na outra tabela. As tabelas esquerdas e direitas são a primeiras e a segunda listadas na sintaxe. Os valores NULL são usados para preencher "lacunas" no conjunto de resultados. Essa consulta corresponde aos valores da coluna LISTID na tabela LISTING (a tabela à esquerda) e na tabela SALES (tabela à direita). Os resultados mostram que LISTIDs 2 e 3 não resultaram em nenhuma venda.

```
select listing.listid, sum(pricepaid) as price, sum(commission) as comm
from listing left outer join sales on sales.listid = listing.listid
where listing.listid between 1 and 5
group by 1
order by 1;

listid | price  |  comm
-------+--------+--------
     1 | 728.00 | 109.20
     2 | NULL   | NULL
     3 | NULL   | NULL
     4 |  76.00 |  11.40
     5 | 525.00 |  78.75
```

A consulta a seguir é uma junção externa à direita. Essa consulta corresponde aos valores da coluna LISTID na tabela LISTING (a tabela à esquerda) e na tabela SALES (tabela à direita). Os resultados mostram que LISTIDs 1, 4 e 5 correspondem aos critérios.

```
select listing.listid, sum(pricepaid) as price, sum(commission) as comm
from listing right outer join sales on sales.listid = listing.listid
where listing.listid between 1 and 5
group by 1
order by 1;
```

```
listid | price  |  comm
-------+--------+--------
     1 | 728.00 | 109.20
     4 |  76.00 |  11.40
     5 | 525.00 |  78.75
```

A consulta a seguir é uma junção completa. As junções completas retêm valores das tabelas unidas quando nenhuma correspondência é encontrada na outra tabela. As tabelas esquerdas e direitas são a primeiras e a segunda listadas na sintaxe. Os valores NULL são usados para preencher "lacunas" no conjunto de resultados. Essa consulta corresponde aos valores da coluna LISTID na tabela LISTING (a tabela à esquerda) e na tabela SALES (tabela à direita). Os resultados mostram que LISTIDs 2 e 3 não resultaram em nenhuma venda.

```
select listing.listid, sum(pricepaid) as price, sum(commission) as comm
from listing full join sales on sales.listid = listing.listid
where listing.listid between 1 and 5
group by 1
order by 1;

listid | price  |  comm
-------+--------+--------
     1 | 728.00 | 109.20
     2 | NULL   | NULL
     3 | NULL   | NULL
     4 |  76.00 |  11.40
     5 | 525.00 |  78.75
```

A consulta a seguir é uma junção completa. Essa consulta corresponde aos valores da coluna LISTID na tabela LISTING (a tabela à esquerda) e na tabela SALES (tabela à direita). Somente linhas que não resultam em vendas (LISTIDs 2 e 3) estão nos resultados.

```
select listing.listid, sum(pricepaid) as price, sum(commission) as comm
from listing full join sales on sales.listid = listing.listid
where listing.listid between 1 and 5
and (listing.listid IS NULL or sales.listid IS NULL)
group by 1
order by 1;

listid | price  |  comm
-------+--------+--------
     2 | NULL   | NULL
```

```
     3 | NULL   | NULL
```

O exemplo a seguir é uma junção interna com a cláusula ON. Nesse caso, as linhas NULL não são retornadas.

```
select listing.listid, sum(pricepaid) as price, sum(commission) as comm
from sales join listing
on sales.listid=listing.listid and sales.eventid=listing.eventid
where listing.listid between 1 and 5
group by 1
order by 1;

listid | price  |  comm
-------+--------+--------
     1 | 728.00 | 109.20
     4 |  76.00 |  11.40
     5 | 525.00 |  78.75
```

A consulta a seguir é uma junção cruzada ou junção cartesiana da tabela LISTING e da tabela SALES com um predicado para limitar os resultados. Essa consulta corresponde aos valores da coluna LISTID na tabela SALES e na tabela LISTING para LISTIDs 1, 2, 3, 4 e 5 em ambas as tabelas. Os resultados mostram que 20 linhas correspondem aos critérios.

```
select sales.listid as sales_listid, listing.listid as listing_listid
from sales cross join listing
where sales.listid between 1 and 5
and listing.listid between 1 and 5
order by 1,2;

sales_listid | listing_listid
-------------+---------------
1            | 1
1            | 2
1            | 3
1            | 4
1            | 5
4            | 1
4            | 2
4            | 3
4            | 4
4            | 5
5            | 1
```

```
5            | 1
5            | 2
5            | 2
5            | 3
5            | 3
5            | 4
5            | 4
5            | 5
5            | 5
```

O exemplo a seguir é uma junção natural entre duas tabelas. Nesse caso, as colunas listid, sellerid, eventid e dateid têm nomes e tipos de dados idênticos em ambas as tabelas e, portanto, são usadas como colunas de junção. Os resultados são limitados a cinco linhas.

```
select listid, sellerid, eventid, dateid, numtickets
from listing natural join sales
order by 1
limit 5;

listid | sellerid  | eventid | dateid | numtickets
-------+-----------+---------+--------+-----------
113    | 29704     | 4699    | 2075   | 22
115    | 39115     | 3513    | 2062   | 14
116    | 43314     | 8675    | 1910   | 28
118    | 6079      | 1611    | 1862   | 9
163    | 24880     | 8253    | 1888   | 14
```

O exemplo a seguir é uma junção entre duas tabelas com a cláusula USING. Nesse caso, as colunas listid e eventid são usadas como colunas de junção. Os resultados são limitados a cinco linhas.

```
select listid, listing.sellerid, eventid, listing.dateid, numtickets
from listing join sales
using (listid, eventid)
order by 1
limit 5;

listid | sellerid | eventid | dateid | numtickets
-------+----------+---------+--------+-----------
1      | 36861    | 7872    | 1850   | 10
4      | 8117     | 4337    | 1970   | 8
5      | 1616     | 8647    | 1963   | 4
5      | 1616     | 8647    | 1963   | 4
```

```
6      | 47402    | 8240    | 2053   | 18
```

A consulta a seguir é uma junção interna de duas subconsultas na cláusula FROM. A consulta encontra o número de ingressos vendidos e não vendidos para categorias diferentes de eventos (shows e apresentações). As subconsultas da cláusula FROM são subconsultas da tabela. Elas podem retornar várias colunas e linhas.

```
select catgroup1, sold, unsold
from
(select catgroup, sum(qtysold) as sold
from category c, event e, sales s
where c.catid = e.catid and e.eventid = s.eventid
group by catgroup) as a(catgroup1, sold)
join
(select catgroup, sum(numtickets)-sum(qtysold) as unsold
from category c, event e, sales s, listing l
where c.catid = e.catid and e.eventid = s.eventid
and s.listid = l.listid
group by catgroup) as b(catgroup2, unsold)

on a.catgroup1 = b.catgroup2
order by 1;

catgroup1 |  sold  | unsold
----------+--------+--------
Concerts  | 195444 |1067199
Shows     | 149905 | 817736
```

Exemplos de UNNEST

UNNEST é um parâmetro na cláusula FROM que expande os dados aninhados em colunas que contêm os elementos não aninhados dos dados. Para ter informações sobre como desagrupar dados, consulte Consultar dados semiestruturados.

A instrução a seguir cria e preenche a tabela `orders`, que contém uma coluna `products` com matrizes de IDs de produto. Os exemplos desta seção utilizam os dados de amostra nessa tabela.

```
CREATE TABLE orders (
    order_id INT,
    products SUPER
);
```

```
-- Populate table
INSERT INTO orders VALUES
(1001, JSON_PARSE('[
        {
            "product_id": "P456",
            "name": "Monitor",
            "price": 299.99,
            "quantity": 1,
            "specs": {
                "size": "27 inch",
                "resolution": "4K"
            }
        }
    ]
')),
(1002, JSON_PARSE('
    [
        {
            "product_id": "P567",
            "name": "USB Cable",
            "price": 9.99,
            "quantity": 3
        },
        {
            "product_id": "P678",
            "name": "Headphones",
            "price": 159.99,
            "quantity": 1,
            "specs": {
                "type": "Wireless",
                "battery_life": "20 hours"
            }
        }
    ]
'));
```

Veja a seguir alguns exemplos de consulta desaninhada com os dados de amostra usando a sintaxe partiQL.

Desagrupar uma matriz sem uma coluna OFFSET

A consulta a seguir desagrupa as matrizes SUPER na coluna de produtos, e cada linha representa um item do pedido em `order_id`.

```
SELECT o.order_id, unnested_products.product
FROM orders o, UNNEST(o.products) AS unnested_products(product);

 order_id |                                                
```

```
----------
+-----------------------------------------------------------------------------------------------
+------------
     1001 | {"product_id":"P456","name":"Monitor","price":299.99,"quantity":1,"specs":
{"size":"27 inch","resolution":"4K"}}             |          0
     1002 | {"product_id":"P567","name":"USB Cable","price":9.99,"quantity":3}
                                                  |          0
     1002 |
 {"product_id":"P678","name":"Headphones","price":159.99,"quantity":1,"specs":
{"type":"Wireless","battery_life":"20 hours"}} |          1
(3 rows)
```

Como a declaração não especifica um alias para a coluna de deslocamento, o Amazon Redshift a denomina `offset_col` por padrão.

Desagrupar uma matriz com uma coluna OFFSET explícita

A consulta a seguir também usa o parâmetro `UNNEST ... WITH OFFSET` para mostrar os produtos nas respectivas matrizes de pedidos. A diferença nessa consulta em comparação com a consulta no exemplo anterior é que ela nomeia explicitamente a coluna de deslocamento com o alias `idx`.

```
SELECT o.order_id, up.product, up.idx
FROM orders o, UNNEST(o.products) WITH OFFSET AS up(product, idx);

 order_id |                                                                 product
                                                  | idx
----------
+-----------------------------------------------------------------------------------------------
+-----
     1001 | {"product_id":"P456","name":"Monitor","price":299.99,"quantity":1,"specs":
{"size":"27 inch","resolution":"4K"}}             |   0
     1002 | {"product_id":"P567","name":"USB Cable","price":9.99,"quantity":3}
                                                  |   0
     1002 |
 {"product_id":"P678","name":"Headphones","price":159.99,"quantity":1,"specs":
{"type":"Wireless","battery_life":"20 hours"}} |   1
(3 rows)
```

Cláusula WHERE

A cláusula WHERE contém as condições que juntam as tabelas ou aplicam predicados às colunas nas tabelas. Tabelas internas que foram juntadas usando a sintaxe apropriada, seja com a cláusula

WHERE ou com a cláusula FROM. Os critérios de junção externa devem ser especificados na cláusula FROM.

Sintaxe

```
[ WHERE condition ]
```

condição

Qualquer condição de pesquisa com um resultado booleano, como uma condição de junção ou um predicado em uma coluna de tabela. Os exemplos a seguir são condições de junção válidas:

```
sales.listid=listing.listid
sales.listid<>listing.listid
```

Os exemplos a seguir são condições válidas nas colunas em tabelas:

```
catgroup like 'S%'
venueseats between 20000 and 50000
eventname in('Jersey Boys','Spamalot')
year=2008
length(catdesc)>25
date_part(month, caldate)=6
```

As condições podem ser simples ou complexas; para condições complexas, você pode usar parênteses para isolar unidades lógicas. No exemplo a seguir, a condição de junção está entre parênteses.

```
where (category.catid=event.catid) and category.catid in(6,7,8)
```

Observações de uso

É possível usar aliases na cláusula WHERE para fazer referência a expressões da lista de seleção.

Não é possível restringir os resultados de funções agregadas na cláusula WHERE; use a cláusula HAVING para essa finalidade.

Colunas restringidas na cláusula WHERE devem ser derivadas de referências da tabela na cláusula FROM.

Exemplo

A consulta a seguir usa uma combinação de diferentes restrições da cláusula WHERE, incluindo uma condição de junção para as tabelas SALES e EVENT, um predicado na coluna EVENTNAME e dois predicados na coluna STARTTIME.

```
select eventname, starttime, pricepaid/qtysold as costperticket, qtysold
from sales, event
where sales.eventid = event.eventid
and eventname='Hannah Montana'
and date_part(quarter, starttime) in(1,2)
and date_part(year, starttime) = 2008
order by 3 desc, 4, 2, 1 limit 10;

eventname    |      starttime      |   costperticket   | qtysold
---------------+---------------------+-------------------+---------
Hannah Montana | 2008-06-07 14:00:00 |     1706.00000000 |       2
Hannah Montana | 2008-05-01 19:00:00 |     1658.00000000 |       2
Hannah Montana | 2008-06-07 14:00:00 |     1479.00000000 |       1
Hannah Montana | 2008-06-07 14:00:00 |     1479.00000000 |       3
Hannah Montana | 2008-06-07 14:00:00 |     1163.00000000 |       1
Hannah Montana | 2008-06-07 14:00:00 |     1163.00000000 |       2
Hannah Montana | 2008-06-07 14:00:00 |     1163.00000000 |       4
Hannah Montana | 2008-05-01 19:00:00 |      497.00000000 |       1
Hannah Montana | 2008-05-01 19:00:00 |      497.00000000 |       2
Hannah Montana | 2008-05-01 19:00:00 |      497.00000000 |       4
(10 rows)
```

Junções externas do estilo Oracle na cláusula WHERE

Para compatibilidade com a Oracle, o Amazon Redshift oferece suporte ao operador de junção externa da Oracle (+) nas condições de junção da cláusula WHERE. Esse operador deve ser usado apenas para definir condições de junção externas; não tente usar em outros contextos. Outros usos para este operador são ignorados silenciosamente na maioria dos casos.

Uma junção externa retorna todas as linhas que a junção interna equivalente deve retornar e linhas não correspondentes de uma ou de ambas as tabelas. Na cláusula FROM, você pode especificar junções esquerdas, direitas e externas. Na cláusula WHERE, você pode especificar somente junções externas esquerdas e direitas.

Para juntar as tabelas externas TABLE1 e TABLE2 e retornar linhas não correspondentes da TABLE1 (junção externa esquerda), especifique `TABLE1 LEFT OUTER JOIN TABLE2` na cláusula

ou aplique o operador (+) a todas as colunas de junção de TABLE2 na cláusula WHERE. Para todas as linhas na TABLE1 que não têm linhas correspondentes na TABLE2, o resultado da consulta contém nulos para quaisquer expressões de lista de seleção contendo colunas da TABLE2.

Para produzir o mesmo comportamento em todas as linhas na TABLE2 que não têm linhas correspondentes na TABLE1, especifique `TABLE1 RIGHT OUTER JOIN TABLE2` na cláusula FROM ou aplique o operador (+) a todas as colunas de junção da TABLE1 na cláusula WHERE.

Sintaxe básica

```
[ WHERE {
[ table1.column1 = table2.column1(+) ]
[ table1.column1(+) = table2.column1 ]
}
```

A primeira condição equivale a:

```
from table1 left outer join table2
on table1.column1=table2.column1
```

A segunda condição equivale a:

```
from table1 right outer join table2
on table1.column1=table2.column1
```

> **Note**
>
> A sintaxe mostrada aqui abrange o caso simples de uma junção equivalente em um par de colunas de junção. Porém, outros tipos de condições de comparação e diversos pares de colunas de junção também são válidos.

Por exemplo, a cláusula WHERE a seguir define um junção externa em relação a dois pares de colunas. O operador (+) deve ser vinculado à mesma tabela em ambas as condições:

```
where table1.col1 > table2.col1(+)
and table1.col2 = table2.col2(+)
```

Observações de uso

Sempre que possível, use a sintaxe OUTER JOIN da cláusula FROM padrão em vez do operador (+) na cláusula WHERE. Consultas contendo o operador (+) estão sujeitas às seguintes regras:

- Você só pode usar o operador (+) na cláusula WHERE, e somente em referência a colunas de tabelas ou exibições.
- Você não pode aplicar o operador (+) a expressões. No entanto, uma expressão pode conter colunas que usam o operador (+). Por exemplo, a condição de junção a seguir retorna um erro de sintaxe:

  ```
  event.eventid*10(+)=category.catid
  ```

  No entanto, a seguinte condição de junção é válida:

  ```
  event.eventid(+)*10=category.catid
  ```

- Você não pode usar o operador (+) em um bloco de consulta que também contenha a sintaxe de junção da cláusula FROM.
- Se duas tabelas são adicionadas em diversas condições de junção, você deve usar o operador (+) em todas ou em nenhuma dessas condições. Uma junção com estilos mistos de sintaxe é executada como uma junção interna, sem aviso.
- O operador (+) não produzirá um uma junção externa se você juntar uma tabela na consulta externa com uma tabela que resulte de uma consulta interna.
- Para usar o operador (+) para juntar uma tabela externa na própria tabela, você deve definir aliases da tabela na cláusula FROM e fazer referência a eles na condição de junção:

  ```
  select count(*)
  from event a, event b
  where a.eventid(+)=b.catid;

  count
  -------
  8798
  (1 row)
  ```

- Você não pode combinar uma condição de junção que contenha o operador (+) com uma condição OR ou IN. Por exemplo:

```
select count(*) from sales, listing
where sales.listid(+)=listing.listid or sales.salesid=0;
ERROR:  Outer join operator (+) not allowed in operand of OR or IN.
```

- Em uma cláusula WHERE faz junções externas de mais de duas tabelas, o operador (+) pode ser aplicado somente uma vez a uma tabela específica. No exemplo a seguir, não é possível fazer referência à tabela SALES com o operador (+) em duas junções sucessivas.

```
select count(*) from sales, listing, event
where sales.listid(+)=listing.listid and sales.dateid(+)=date.dateid;
ERROR:  A table may be outer joined to at most one other table.
```

- Se a condição de junção externa da cláusula WHERE se comparar a uma coluna da TABLE2 com uma constante, aplique o operador (+) à coluna. Se você não incluir o operador, as linhas de junções externas da TABLE1 que contêm nulos para a coluna restringida serão eliminadas. Consulte a seção de Exemplos abaixo.

Exemplos

A seguinte consulta de junção especifica uma junção esquerda externa das tabelas SALES e LISTING em suas colunas LISTID:

```
select count(*)
from sales, listing
where sales.listid = listing.listid(+);

count
--------
172456
(1 row)
```

A seguinte consulta equivalente produz o mesmo resultado, mas usa a sintaxe de junção da cláusula FROM:

```
select count(*)
from sales left outer join listing on sales.listid = listing.listid;

count
--------
172456
```

```
(1 row)
```

A tabela SALES não contém registros de todas as listagens na tabela LISTING, pois nem todas as listagens resultam em vendas. A consulta a seguir junta externamente as tabelas SALES e LISTING e retorna linhas da tabela LISTING mesmo quando a tabela SALES não retorna vendas para determinado ID de lista. As colunas PRICE e COMM, derivadas da tabela SALES, contêm nulos no conjunto de resultados para linhas não correspondentes.

```
select listing.listid, sum(pricepaid) as price,
sum(commission) as comm
from listing, sales
where sales.listid(+) = listing.listid and listing.listid between 1 and 5
group by 1 order by 1;

listid | price  |  comm
--------+--------+--------
1 | 728.00 | 109.20
2 |        |
3 |        |
4 |  76.00 |  11.40
5 | 525.00 |  78.75
(5 rows)
```

Observe que quando o operador de junção da cláusula WHERE é usado, a ordem das tabelas na cláusula FROM não importa.

Um exemplo de uma condição de junção externa mais complexa na cláusula WHERE é o caso em que a condição consiste em uma comparação entre duas tabelas comuns e uma comparação com uma constante:

```
where category.catid=event.catid(+) and eventid(+)=796;
```

Observe que o operador (+) é usado em dois lugares: primeiro na comparação de correspondência entre as tabelas e depois na condição de comparação para a coluna EVENTID. O resultado da sintaxe é a preservação das linhas de junção externa quando a restrição em EVENTID é avaliada. Se você remover o operador (+) da restrição EVENTID, a consulta trata a restrição como um filtro, não como parte da condição de junção externa. Por sua vez, as colunas de junção externa que contêm nulos para EVENTID são eliminadas do conjunto de resultados.

Veja aqui uma consulta completa que ilustra esse comportamento:

```
select catname, catgroup, eventid
from category, event
where category.catid=event.catid(+) and eventid(+)=796;

catname | catgroup | eventid
-----------+----------+---------
Classical | Concerts |
Jazz | Concerts |
MLB | Sports   |
MLS | Sports   |
Musicals | Shows    | 796
NBA | Sports   |
NFL | Sports   |
NHL | Sports   |
Opera | Shows    |
Plays | Shows    |
Pop | Concerts |
(11 rows)
```

A consulta equivalente usando a sintaxe de cláusula FROM é:

```
select catname, catgroup, eventid
from category left join event
on category.catid=event.catid and eventid=796;
```

Se você remover o segundo operador (+) da versão da cláusula WHERE desta consulta, ela retornará somente 1 linha (a linha de `eventid=796`).

```
select catname, catgroup, eventid
from category, event
where category.catid=event.catid(+) and eventid=796;

catname | catgroup | eventid
-----------+----------+---------
Musicals | Shows    | 796
(1 row)
```

Cláusula GROUP BY

A cláusula GROUP BY identifica as colunas de agrupamento para a consulta. É usada para agrupar as linhas em uma tabela que têm os mesmos valores em todas as colunas listadas. A ordem em

que as colunas são listadas não importa. O resultado é a combinação de cada conjunto de linhas com valores comuns em uma linha que representa todas as linhas do grupo. Use GROUP BY para eliminar a redundância na saída e calcular agregados que se aplicam aos grupos. As colunas de agrupamento devem ser declaradas quando a consulta computa agregadas com funções padrão como SUM, AVG e COUNT. Para obter mais informações, consulte Funções agregadas.

Sintaxe

```
[ GROUP BY  expression [, ...] | ALL | aggregation_extension  ]
```

em que aggregation_extension é uma das seguintes:

```
GROUPING SETS ( () | aggregation_extension [, ...] ) |
ROLLUP ( expr [, ...] ) |
CUBE ( expr [, ...] )
```

Parâmetros

expressão

A lista de colunas ou de expressões deve corresponder à lista de expressões não agregadas na lista de seleção da consulta. Por exemplo, considere a seguinte consulta simples.

```
select listid, eventid, sum(pricepaid) as revenue,
count(qtysold) as numtix
from sales
group by listid, eventid
order by 3, 4, 2, 1
limit 5;

listid | eventid | revenue | numtix
-------+---------+---------+--------
89397  |      47 |   20.00 |      1
106590 |      76 |   20.00 |      1
124683 |     393 |   20.00 |      1
103037 |     403 |   20.00 |      1
147685 |     429 |   20.00 |      1
(5 rows)
```

Nesta consulta, a lista de seleção consiste em duas expressões agregadas. A primeira usa a função SUM e a segunda usa a função COUNT. As duas colunas restantes, LISTID e EVENTID, devem ser declaradas como colunas de agrupamento.

As expressões na cláusula GROUP BY também podem fazer referência à lista de seleção usando números ordinais. O exemplo anterior poderia ser abreviado da seguinte forma.

```
select listid, eventid, sum(pricepaid) as revenue,
count(qtysold) as numtix
from sales
group by 1,2
order by 3, 4, 2, 1
limit 5;

listid | eventid | revenue | numtix
-------+---------+---------+--------
89397  |      47 |   20.00 |      1
106590 |      76 |   20.00 |      1
124683 |     393 |   20.00 |      1
103037 |     403 |   20.00 |      1
147685 |     429 |   20.00 |      1
(5 rows)
```

ALL

ALL indica agrupar por todas as colunas especificadas na lista SELECT, exceto as agregadas. Por exemplo, pense na consulta a seguir, que agrupa por `col1` e `col2` sem precisar especificá-las individualmente na cláusula GROUP BY. A coluna `col3` é o argumento da função `SUM` e, portanto, não está agrupada.

```
SELECT col1, col2 sum(col3) FROM testtable GROUP BY ALL
```

Se você excluir uma coluna na lista SELECT, a cláusula GROUP BY ALL não agrupará os resultados com base nessa coluna específica.

```
SELECT * EXCLUDE col3 FROM testtable GROUP BY ALL
```

aggregation_extension

Você pode usar as extensões de agregação GROUPING SETS, ROLLUP e CUBE para realizar o trabalho de várias operações GROUP BY em uma única instrução. Para obter mais informações sobre extensões de agregação e funções relacionadas, consulte Extensões de agregação.

Exemplos

Os exemplos a seguir usam a tabela SALES que contém as colunas: salesid, listid, sellerid, buyerid, eventid, dateid, qtysold, pricepaid, commission e saletime. Para ter mais informações sobre a tabela SALES, consulte Banco de dados de exemplo.

O exemplo de consulta a seguir agrupa por `salesid` e `listid` sem precisar especificá-las individualmente na cláusula GROUP BY. A coluna `qtysold` é o argumento da função SUM e, portanto, não está agrupada.

```
SELECT salesid, listid, sum(qtysold) FROM sales GROUP BY ALL;

salesid | listid  | sum
--------+---------+------
33095   | 36572   | 2
88268   | 100813  | 4
110917  | 127048  | 1
...
```

O exemplo de consulta a seguir exclui várias colunas na lista SELECT, então GROUP BY ALL agrupa somente salesid e listid.

```
SELECT * EXCLUDE sellerid, buyerid, eventid, dateid, qtysold, pricepaid, commission,
 saletime
FROM sales GROUP BY ALL;

salesid | listid
--------+---------
33095   | 36572
88268   | 100813
110917  | 127048
...
```

Extensões de agregação

O Amazon Redshift oferece suporte a extensões de agregação para realizar o trabalho de várias operações GROUP BY em uma única instrução.

Os exemplos de extensões de agregação usam a tabela `orders`, que contém dados de vendas de uma empresa de dispositivos eletrônicos. Você pode criar `orders` com o seguinte.

```
CREATE TABLE ORDERS (
    ID INT,
    PRODUCT CHAR(20),
    CATEGORY CHAR(20),
    PRE_OWNED CHAR(1),
    COST DECIMAL
);

INSERT INTO ORDERS VALUES
    (0, 'laptop',       'computers',    'T', 1000),
    (1, 'smartphone',   'cellphones',   'T', 800),
    (2, 'smartphone',   'cellphones',   'T', 810),
    (3, 'laptop',       'computers',    'F', 1050),
    (4, 'mouse',        'computers',    'F', 50);
```

GROUPING SETS

Calcula um ou mais conjuntos de agrupamento em uma única instrução. Um conjunto de agrupamento é o conjunto de uma única cláusula GROUP BY, um conjunto de 0 ou mais colunas pelo qual você pode agrupar o conjunto de resultados de uma consulta. GROUP BY GROUPING SETS é equivalente a executar uma consulta UNION ALL em um conjunto de resultados agrupado por colunas diferentes. Por exemplo, GROUP BY GROUPING SETS((a), (b)) é equivalente a GROUP BY a UNION ALL GROUP BY b.

O exemplo a seguir retorna o custo dos produtos da tabela de pedidos agrupados de acordo com as categorias de produtos e o tipo de produto vendido.

```
SELECT category, product, sum(cost) as total
FROM orders
GROUP BY GROUPING SETS(category, product);

       category       |       product        | total
----------------------+----------------------+-------
```

```
 computers            |                      |  2100
 cellphones           |                      |  1610
                      | laptop               |  2050
                      | smartphone           |  1610
                      | mouse                |    50

(5 rows)
```

ROLLUP

Assume uma hierarquia em que as colunas anteriores são consideradas pais das colunas subsequentes. ROLLUP agrupa os dados pelas colunas fornecidas, retornando linhas de subtotal extras representando os totais em todos os níveis de colunas de agrupamento, além das linhas agrupadas. Por exemplo, você pode usar GROUP BY ROLLUP((a), (b)) para retornar um conjunto de resultados agrupado primeiro por a, depois por b, assumindo que b é uma subseção de a. ROLLUP também retorna uma linha com todo o conjunto de resultados sem colunas de agrupamento.

GROUP BY ROLLUP((a), (b)) é equivalente a GROUP BY GROUPING SETS((a,b), (a), ()).

O exemplo a seguir retorna o custo dos produtos da tabela de pedidos agrupados primeiro por categoria, depois por produto, com o produto como uma subdivisão da categoria.

```
SELECT category, product, sum(cost) as total
FROM orders
GROUP BY ROLLUP(category, product) ORDER BY 1,2;

       category       |       product        | total
----------------------+----------------------+-------
 cellphones           | smartphone           |  1610
 cellphones           |                      |  1610
 computers            | laptop               |  2050
 computers            | mouse                |    50
 computers            |                      |  2100
                      |                      |  3710
(6 rows)
```

CUBE

Agrupa os dados pelas colunas fornecidas, retornando linhas de subtotal extras representando os totais em todos os níveis de colunas de agrupamento, além das linhas agrupadas. CUBE retorna as mesmas linhas que ROLLUP, enquanto inclui linhas de subtotal adicionais para cada combinação

de coluna de agrupamento não contemplada por ROLLUP. Por exemplo, você pode usar GROUP BY CUBE((a), (b)) para retornar um conjunto de resultados agrupado primeiro por a, depois por b, assumindo que b é uma subseção de a, depois apenas por b. CUBE também retorna uma linha com todo o conjunto de resultados sem colunas de agrupamento.

GROUP BY CUBE((a), (b)) é equivalente a GROUP BY GROUPING SETS((a,b), (a), (b), ()).

O exemplo a seguir retorna o custo dos produtos da tabela de pedidos agrupados primeiro por categoria, depois por produto, com o produto como uma subdivisão da categoria. Ao contrário do exemplo anterior para ROLLUP, a instrução retorna resultados para cada combinação de coluna de agrupamento.

```
SELECT category, product, sum(cost) as total
FROM orders
GROUP BY CUBE(category, product) ORDER BY 1,2;

       category       |       product        | total
----------------------+----------------------+-------
 cellphones           | smartphone           |  1610
 cellphones           |                      |  1610
 computers            | laptop               |  2050
 computers            | mouse                |    50
 computers            |                      |  2100
                      | laptop               |  2050
                      | mouse                |    50
                      | smartphone           |  1610
                      |                      |  3710
(9 rows)
```

Funções GROUPING/GROUPING_ID

ROLLUP e CUBE adicionam valores NULL ao conjunto de resultados para indicar linhas de subtotal. Por exemplo, GROUP BY ROLLUP((a), (b)) retorna uma ou mais linhas que têm um valor NULL na coluna de agrupamento b para indicar que são subtotais de campos na coluna de agrupamento a. Esses valores NULL servem apenas para satisfazer o formato das tuplas de retorno.

Quando você executa operações GROUP BY com ROLLUP e CUBE em relações que armazenam valores NULL em si, isso pode produzir conjuntos de resultados com linhas que parecem ter colunas de agrupamento idênticas. Voltando ao exemplo anterior, se a coluna de agrupamento b contiver um valor NULL armazenado, GROUP BY ROLLUP((a), (b)) retornará uma linha com um valor NULL na coluna de agrupamento b que não é um subtotal.

Para distinguir entre valores NULL criados por ROLLUP e CUBE e os valores NULL armazenados nas próprias tabelas, você pode usar a função GROUPING ou seu alias GROUPING_ID. GROUPING usa um único conjunto de agrupamento como argumento e, para cada linha no conjunto de resultados, retorna um valor de bit 0 ou 1 correspondente à coluna de agrupamento nessa posição, depois converte esse valor em um inteiro. Se o valor nessa posição for um valor NULL criado por uma extensão de agregação, GROUPING retornará 1. Retornará 0 para todos os outros valores, incluindo valores NULL armazenados.

Por exemplo, GROUPING(category, product) pode retornar os seguintes valores para determinada linha, dependendo dos valores da coluna de agrupamento dessa linha. Neste exemplo, todos os valores NULL na tabela são valores NULL criados por uma extensão de agregação.

Coluna category	Coluna product	Valor de bit da função GROUPING	Valor decimal
não NULL	não NULL	00	0
não NULL	NULL	01	1
NULL	não NULL	10	2
NULL	NULL	11	3

As funções GROUPING aparecem na parte da lista SELECT da consulta no formato a seguir.

```
SELECT ... [GROUPING( expr )...] ...
  GROUP BY ... {CUBE | ROLLUP| GROUPING SETS} ( expr ) ...
```

O exemplo a seguir é igual ao exemplo anterior para CUBE, mas com a adição de funções GROUPING para seus conjuntos de agrupamento.

```
SELECT category, product,
       GROUPING(category) as grouping0,
       GROUPING(product) as grouping1,
       GROUPING(category, product) as grouping2,
       sum(cost) as total
FROM orders
GROUP BY CUBE(category, product) ORDER BY 3,1,2;
```

```
      category       |       product        | grouping0 | grouping1 | grouping2 |
 total
---------------------+----------------------+-----------+-----------+-----------
+-------
 cellphones          | smartphone           |         0 |         0 |         0 |
 1610
 cellphones          |                      |         0 |         1 |         1 |
 1610
 computers           | laptop               |         0 |         0 |         0 |
 2050
 computers           | mouse                |         0 |         0 |         0 |
 50
 computers           |                      |         0 |         1 |         1 |
 2100
                     | laptop               |         1 |         0 |         2 |
 2050
                     | mouse                |         1 |         0 |         2 |
 50
                     | smartphone           |         1 |         0 |         2 |
 1610
                     |                      |         1 |         1 |         3 |
 3710
(9 rows)
```

ROLLUP e CUBE parciais

Você pode executar operações ROLLUP e CUBE com apenas uma parte dos subtotais.

A sintaxe para operações parciais de ROLLUP e CUBE é a seguinte.

```
GROUP BY expr1, { ROLLUP | CUBE }(expr2, [, ...])
```

Aqui, a cláusula GROUP BY cria apenas linhas de subtotal no nível de expr2 e em diante.

Os exemplos a seguir mostram operações parciais de ROLLUP e CUBE na tabela de pedidos, primeiro agrupando os produtos seminovos, depois executando ROLLUP e CUBE nas colunas category e product.

```
SELECT pre_owned, category, product,
       GROUPING(category, product, pre_owned) as group_id,
       sum(cost) as total
```

```
FROM orders
GROUP BY pre_owned, ROLLUP(category, product) ORDER BY 4,1,2,3;

 pre_owned |       category       |       product        | group_id | total
-----------+----------------------+----------------------+----------+-------
 F         | computers            | laptop               |        0 |  1050
 F         | computers            | mouse                |        0 |    50
 T         | cellphones           | smartphone           |        0 |  1610
 T         | computers            | laptop               |        0 |  1000
 F         | computers            |                      |        2 |  1100
 T         | cellphones           |                      |        2 |  1610
 T         | computers            |                      |        2 |  1000
 F         |                      |                      |        6 |  1100
 T         |                      |                      |        6 |  2610
(9 rows)

SELECT pre_owned, category, product,
       GROUPING(category, product, pre_owned) as group_id,
       sum(cost) as total
FROM orders
GROUP BY pre_owned, CUBE(category, product) ORDER BY 4,1,2,3;

 pre_owned |       category       |       product        | group_id | total
-----------+----------------------+----------------------+----------+-------
 F         | computers            | laptop               |        0 |  1050
 F         | computers            | mouse                |        0 |    50
 T         | cellphones           | smartphone           |        0 |  1610
 T         | computers            | laptop               |        0 |  1000
 F         | computers            |                      |        2 |  1100
 T         | cellphones           |                      |        2 |  1610
 T         | computers            |                      |        2 |  1000
 F         |                      | laptop               |        4 |  1050
 F         |                      | mouse                |        4 |    50
 T         |                      | laptop               |        4 |  1000
 T         |                      | smartphone           |        4 |  1610
 F         |                      |                      |        6 |  1100
 T         |                      |                      |        6 |  2610
(13 rows)
```

Como a coluna pre-owned não está incluída nas operações ROLLUP e CUBE, não há uma linha de total geral que inclua todas as outras linhas.

Agrupamento concatenado

Você pode concatenar várias cláusulas GROUPING SETS/ROLLUP/CUBE para calcular diferentes níveis de subtotal. Os agrupamentos concatenados retornam o produto cartesiano dos conjuntos de agrupamento fornecidos.

A sintaxe para concatenar as cláusulas GROUPING SETS/ROLLUP/CUBE é a seguinte.

```
GROUP BY {ROLLUP|CUBE|GROUPING SETS}(expr1[, ...]),
         {ROLLUP|CUBE|GROUPING SETS}(expr1[, ...])[, ...]
```

Considere o exemplo a seguir para ver como um pequeno agrupamento concatenado pode produzir um grande conjunto de resultados finais.

```
SELECT pre_owned, category, product,
       GROUPING(category, product, pre_owned) as group_id,
       sum(cost) as total
FROM orders
GROUP BY CUBE(category, product), GROUPING SETS(pre_owned, ())
ORDER BY 4,1,2,3;

 pre_owned |       category       |       product        | group_id | total
-----------+----------------------+----------------------+----------+-------
 F         | computers            | laptop               |        0 |  1050
 F         | computers            | mouse                |        0 |    50
 T         | cellphones           | smartphone           |        0 |  1610
 T         | computers            | laptop               |        0 |  1000
           | cellphones           | smartphone           |        1 |  1610
           | computers            | laptop               |        1 |  2050
           | computers            | mouse                |        1 |    50
 F         | computers            |                      |        2 |  1100
 T         | cellphones           |                      |        2 |  1610
 T         | computers            |                      |        2 |  1000
           | cellphones           |                      |        3 |  1610
           | computers            |                      |        3 |  2100
 F         |                      | laptop               |        4 |  1050
 F         |                      | mouse                |        4 |    50
 T         |                      | laptop               |        4 |  1000
 T         |                      | smartphone           |        4 |  1610
           |                      | laptop               |        5 |  2050
           |                      | mouse                |        5 |    50
           |                      | smartphone           |        5 |  1610
 F         |                      |                      |        6 |  1100
```

```
 T         |                     |                     |        6 |  2610
           |                     |                     |        7 |  3710
(22 rows)
```

Agrupamento aninhado

Você pode usar as operações GROUPING SETS/ROLLUP/CUBE como expr de seu GROUPING SETS para formar um agrupamento aninhado. O subagrupamento dentro da GROUPING SETS aninhada é nivelado.

A sintaxe para agrupamento aninhado é a seguinte.

```
GROUP BY GROUPING SETS({ROLLUP|CUBE|GROUPING SETS}(expr[, ...])[, ...])
```

Considere o exemplo a seguir.

```
SELECT category, product, pre_owned,
       GROUPING(category, product, pre_owned) as group_id,
       sum(cost) as total
FROM orders
GROUP BY GROUPING SETS(ROLLUP(category), CUBE(product, pre_owned))
ORDER BY 4,1,2,3;

       category       |       product        | pre_owned | group_id | total
----------------------+----------------------+-----------+----------+-------
 cellphones           |                      |           |        3 |  1610
 computers            |                      |           |        3 |  2100
                      | laptop               | F         |        4 |  1050
                      | laptop               | T         |        4 |  1000
                      | mouse                | F         |        4 |    50
                      | smartphone           | T         |        4 |  1610
                      | laptop               |           |        5 |  2050
                      | mouse                |           |        5 |    50
                      | smartphone           |           |        5 |  1610
                      |                      | F         |        6 |  1100
                      |                      | T         |        6 |  2610
                      |                      |           |        7 |  3710
                      |                      |           |        7 |  3710
(13 rows)
```

Observe que, como ROLLUP(category) e CUBE(product, pre_owned) contêm o conjunto de agrupamento (), a linha que representa o total geral é duplicada.

Observações de uso

- A cláusula GROUP BY é compatível com até 64 conjuntos de agrupamento. No caso de ROLLUP e CUBE, ou alguma combinação de GROUPING SETS, ROLLUP e CUBE, essa limitação se aplica ao número implícito de conjuntos de agrupamento. Por exemplo, GROUP BY CUBE((a), (b)) conta como 4 conjuntos de agrupamento, não 2.
- Não é possível usar constantes como colunas de agrupamento ao usar extensões de agregação.
- Não é possível fazer um conjunto de agrupamento que contém colunas duplicadas.

Cláusula HAVING

A cláusula HAVING aplica uma condição a um conjunto de resultados agrupados intermediários retornados por uma consulta.

Sintaxe

```
[ HAVING condition ]
```

Por exemplo, você pode restringir os resultados de uma função SUM:

```
having sum(pricepaid) >10000
```

A condição HAVING é aplicada depois que todas as condições da cláusula WHERE forem aplicadas e as operações GROUP BY concluídas.

A própria condição leva a mesma forma que qualquer condição da cláusula WHERE.

Observações de uso

- Qualquer coluna referida na condição da cláusula HAVING deve ser uma coluna de agrupamento ou uma coluna que faz referência ao resultado de uma função agregada.
- Em uma cláusula HAVING, você não pode especificar:
  - Número ordinal que se refere a um item na lista de seleção. Somente as cláusulas GROUP BY e ORDER BY aceitam números ordinais.

Exemplos

A consulta a seguir calcula as vendas de ingressos globais para todos os eventos por nome e depois elimina eventos em que as vendas globais tenham sido menos de $ 800.000. A condição HAVING é aplicada aos resultados da função agregada na lista de seleção: `sum(pricepaid)`.

```
select eventname, sum(pricepaid)
from sales join event on sales.eventid = event.eventid
group by 1
having sum(pricepaid) > 800000
order by 2 desc, 1;

eventname        |    sum
-----------------+-----------
Mamma Mia!       | 1135454.00
Spring Awakening |  972855.00
The Country Girl |  910563.00
Macbeth          |  862580.00
Jersey Boys      |  811877.00
Legally Blonde   |  804583.00
```

A consulta a seguir calcula um conjunto de resultados semelhante. Nesse caso, no entanto, a condição HAVING é aplicada a um valor agregado não especificado na lista de seleção: `sum(qtysold)`. Os eventos que não tenham vendido mais de 2.000 ingressos são eliminados dos resultados finais.

```
select eventname, sum(pricepaid)
from sales join event on sales.eventid = event.eventid
group by 1
having sum(qtysold) >2000
order by 2 desc, 1;

eventname        |    sum
-----------------+-----------
Mamma Mia!       | 1135454.00
Spring Awakening |  972855.00
The Country Girl |  910563.00
Macbeth          |  862580.00
Jersey Boys      |  811877.00
Legally Blonde   |  804583.00
Chicago          |  790993.00
Spamalot         |  714307.00
```

A consulta a seguir calcula as vendas de ingressos globais para todos os eventos por nome e depois elimina eventos em que as vendas globais tenham sido menos de $ 800.000. A condição HAVING é aplicada aos resultados da função agregada na lista de seleção usando o alias pp para `sum(pricepaid)`.

```
select eventname, sum(pricepaid) as pp
from sales join event on sales.eventid = event.eventid
group by 1
having pp > 800000
order by 2 desc, 1;

eventname        |    pp
-----------------+-----------
Mamma Mia!       | 1135454.00
Spring Awakening |  972855.00
The Country Girl |  910563.00
Macbeth          |  862580.00
Jersey Boys      |  811877.00
Legally Blonde   |  804583.00
```

Cláusula QUALIFY

A cláusula QUALIFY filtra os resultados de uma função de janela previamente calculada de acordo com as condições de pesquisa especificadas pelo usuário. Você pode usar a cláusula para aplicar condições de filtragem ao resultado de uma função de janela sem usar uma subconsulta.

É semelhante à cláusula HAVING, que aplica uma condição para filtrar ainda mais as linhas de uma cláusula WHERE. A diferença entre QUALIFY e HAVING é que os resultados filtrados da cláusula QUALIFY podem ser baseados no resultado da execução de funções de janela nos dados. Você pode usar as cláusulas QUALIFY e HAVING na mesma consulta.

Sintaxe

```
QUALIFY condition
```

> **Note**
>
> Se você estiver usando a cláusula QUALIFY diretamente após a cláusula FROM, o nome da relação FROM deverá ter um alias especificado antes da cláusula QUALIFY.

Exemplos

Os exemplos desta seção utilizam os dados de amostra a seguir.

```
create table store_sales (ss_sold_date date, ss_sold_time time,
               ss_item text, ss_sales_price float);
insert into store_sales values ('2022-01-01', '09:00:00', 'Product 1', 100.0),
                               ('2022-01-01', '11:00:00', 'Product 2', 500.0),
                               ('2022-01-01', '15:00:00', 'Product 3', 20.0),
                               ('2022-01-01', '17:00:00', 'Product 4', 1000.0),
                               ('2022-01-01', '18:00:00', 'Product 5', 30.0),
                               ('2022-01-02', '10:00:00', 'Product 6', 5000.0),
                               ('2022-01-02', '16:00:00', 'Product 7', 5.0);
```

O exemplo a seguir demonstra como encontrar os dois itens mais caros vendidos após as 12h de cada dia.

```
SELECT *
FROM store_sales ss
WHERE ss_sold_time > time '12:00:00'
QUALIFY row_number()
OVER (PARTITION BY ss_sold_date ORDER BY ss_sales_price DESC) <= 2


 ss_sold_date | ss_sold_time |  ss_item  | ss_sales_price
--------------+--------------+-----------+----------------
 2022-01-01   | 17:00:00     | Product 4 |           1000
 2022-01-01   | 18:00:00     | Product 5 |             30
 2022-01-02   | 16:00:00     | Product 7 |              5
```

Depois, você pode encontrar o último item vendido em cada dia.

```
SELECT *
FROM store_sales ss
QUALIFY last_value(ss_item)
OVER (PARTITION BY ss_sold_date ORDER BY ss_sold_time ASC
      ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) = ss_item;

ss_sold_date | ss_sold_time |  ss_item  | ss_sales_price
--------------+--------------+-----------+----------------
 2022-01-01   | 18:00:00     | Product 5 |             30
 2022-01-02   | 16:00:00     | Product 7 |              5
```

O exemplo a seguir retorna os mesmos registros que a consulta anterior, do último item vendido em cada dia, mas não usa a cláusula QUALIFY.

```
SELECT * FROM (
  SELECT *,
  last_value(ss_item)
  OVER (PARTITION BY ss_sold_date ORDER BY ss_sold_time ASC
        ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) ss_last_item
  FROM store_sales ss
)
WHERE ss_last_item = ss_item;

 ss_sold_date | ss_sold_time |  ss_item  | ss_sales_price | ss_last_item
--------------+--------------+-----------+----------------+--------------
 2022-01-02   | 16:00:00     | Product 7 |              5 | Product 7
 2022-01-01   | 18:00:00     | Product 5 |             30 | Product 5
```

UNION, INTERSECT e EXCEPT

Tópicos

- Sintaxe
- Parâmetros
- Ordem de avaliação para operadores de conjunto
- Observações de uso
- Exemplos de consultas UNION
- Exemplos de consultas UNION ALL
- Exemplos de consultas INTERSECT
- Exemplos de consultas EXCEPT

Os operadores de conjunto UNION, INTERSECT e EXCEPT são usados para comparar e mesclar os resultados de duas expressões de consulta separadas. Por exemplo, se você quiser saber quais usuários de um site compram e vendem, mas os nomes de usuários estiverem armazenados em colunas ou tabelas separadas, você pode encontrar a interseção desses dois tipos de usuários. Se você quiser saber quais usuários do site compram, mas não vendem, você pode usar o operador EXCEPT para encontrar a diferença entre as duas listas de usuários. Se quiser criar uma lista com todos os usuários, independentemente da função, use o operador UNION.

Sintaxe

```
query
{ UNION [ ALL ] | INTERSECT | EXCEPT | MINUS }
query
```

Parâmetros

query

Uma expressão de consulta que corresponde, na forma de sua lista de seleção, a uma segunda expressão de consulta que segue o operador UNION, INTERSECT ou EXCEPT. As duas expressões devem conter o mesmo número de colunas de saída com tipos de dados compatíveis. Caso contrário, os dois conjuntos de resultados não poderão ser comparados e mesclados. Operações de conjunto não permitem a conversão implícita entre categorias diferentes de tipos de dados. Para obter mais informações, consulte Compatibilidade e conversão dos tipos.

Você pode criar consultas contendo um número ilimitado de expressões de consulta e conectá-las aos operadores UNION, INTERSECT e EXCEPT em qualquer combinação. Por exemplo, a estrutura de consulta a seguir é válida, pressupondo que as tabelas T1, T2 e T3 contenham conjuntos compatíveis de colunas:

```
select * from t1
union
select * from t2
except
select * from t3
order by c1;
```

UNION

Operação de conjunto que retorna linhas de duas expressões de consulta, independentemente das linhas se derivarem de uma ou ambas as expressões.

INTERSECT

Operação de conjunto que retorna linhas derivadas de duas expressões de consulta. As linhas que não forem retornadas por ambas as expressões serão descartadas.

EXCEPT | MINUS

Operação de conjunto que retorna linhas derivadas de uma das duas expressões de consulta. Para se qualificar para o resultado, as linhas precisam existir na primeira tabela de resultados, mas não na segunda. MINUS e EXCEPT são sinônimos.

ALL

A palavra-chave ALL retém todas as linhas duplicadas produzidas por UNION. O comportamento padrão quando a palavra-chave ALL não é utilizada é descartar essas linhas duplicadas. INTERSECT ALL, EXCEPT ALL e MINUS ALL não são compatíveis.

Ordem de avaliação para operadores de conjunto

Os operadores de conjunto UNION e EXCEPT se associam à esquerda. Se não houver parênteses especificados para influenciar a ordem de precedência, uma combinação desses operadores de conjunto será avaliada da esquerda para a direita. Por exemplo, na consulta a seguir, o operador UNION de T1 e T2 é avaliado primeiro, seguido pela operação EXCEPT, que é executada no resultado de UNION:

```
select * from t1
union
select * from t2
except
select * from t3
order by c1;
```

O operador INTERSECT tem precedência sobre os operadores UNION e EXCEPT quando uma combinação de operadores for usada na mesma consulta. Por exemplo, a consulta a seguir avalia a interseção de T2 e T3, e depois une o resultado com T1:

```
select * from t1
union
select * from t2
intersect
select * from t3
order by c1;
```

Adicionando parênteses, você pode aplicar uma ordem diferente de avaliação. No caso a seguir, o resultado da união de T1 e T2 é cruzado com T3, e a consulta provavelmente produzirá um resultado diferente.

```
(select * from t1
union
select * from t2)
intersect
(select * from t3)
order by c1;
```

Observações de uso

- Os nomes de colunas obtidos no resultado de uma consulta de operação de conjunto são os nomes de colunas (ou aliases) das tabelas na primeira expressão de consulta. Como esses nomes de coluna podem induzir a erros, os valores na coluna derivam de tabelas em ambos os lados do operador de conjunto, você pode querer fornecer aliases significativos para o conjunto de resultados.
- Uma expressão de consulta que preceda um operador de conjunto não deve conter uma cláusula ORDER BY. Uma cláusula ORDER BY produz resultados significativos classificados somente quando é usada no final de uma consulta que contenha operadores de conjunto. Nesse caso, a cláusula ORDER BY se aplica a resultados finais de todas as operações de conjunto. A consulta mais externa também pode conter as cláusulas LIMIT e OFFSET padrão.
- Quando as consultas do operador de conjunto retornam resultados decimais, as colunas de resultados correspondentes são promovidas para retornar a mesma precisão e escala. Por exemplo, na consulta a seguir, em que T1.REVENUE é uma coluna DECIMAL(10,2) e T2.REVENUE é uma coluna DECIMAL(8,4), o resultado decimal é atualizado para DECIMAL(12,4):

  ```
  select t1.revenue union select t2.revenue;
  ```

  A escala é 4 porque é a escala máxima das duas colunas. A precisão é 12 porque T1.REVENUE requer 8 dígitos à esquerda do ponto decimal (12 - 4 = 8). Essa promoção de tipo garante que todos os valores de ambos os lados de UNION se encaixem no resultado. Para valores de 64 bits, a precisão máxima de resultado é 19 e a escala máxima de resultado é 18. Para valores de 128-bits, a precisão máxima de resultado é 38 e a escala máxima de resultado é 37.

  Se o tipo de dados resultante ultrapassar os limites de precisão e escala do Amazon Redshift, a consulta retornará um erro.
- Para operações de conjunto, duas linhas são tratadas como idênticas se, para cada par de colunas correspondente, os dois valores de dados forem iguais ou ambos NULL. Por exemplo, se

as tabelas T1 e T2 contiverem uma coluna e uma linha, e a linha for NULL em ambas as tabelas, uma operação INTERSECT sobre essas tabelas retornará essa linha.

Exemplos de consultas UNION

Na consulta UNION a seguir, as linhas na tabela SALES são mescladas com as linhas na tabela LISTING. Três colunas compatíveis de cada tabela são selecionadas. Nesse caso, as colunas correspondentes têm os mesmos nomes e tipos de dados.

O conjunto de resultados finais é classificado pela primeira coluna na tabela LISTING e limitado a 5 linhas com o valor de LISTID mais alto.

```
select listid, sellerid, eventid from listing
union select listid, sellerid, eventid from sales
order by listid, sellerid, eventid desc limit 5;

listid | sellerid | eventid
--------+----------+---------
1 |    36861 |    7872
2 |    16002 |    4806
3 |    21461 |    4256
4 |     8117 |    4337
5 |     1616 |    8647
(5 rows)
```

O exemplo a seguir mostra como você pode adicionar um valor literal de saída de uma consulta UNION para ver qual expressão de consulta produziu cada linha no conjunto de resultados. A consulta identifica linhas da primeira expressão de consulta como “B” (para compradores) e linhas da segunda expressão de consulta como “S” (para vendedores).

A consulta identifica compradores e vendedores para as transações de ingressos que custem $10.000 ou mais. A única diferença entre as duas expressões de consulta em ambos os lados do operador UNION é a coluna de junção para a tabela SALES.

```
select listid, lastname, firstname, username,
pricepaid as price, 'S' as buyorsell
from sales, users
where sales.sellerid=users.userid
and pricepaid >=10000
union
select listid, lastname, firstname, username, pricepaid,
```

```
'B' as buyorsell
from sales, users
where sales.buyerid=users.userid
and pricepaid >=10000
order by 1, 2, 3, 4, 5;

listid | lastname | firstname | username |   price   | buyorsell
--------+----------+-----------+----------+-----------+-----------
209658 | Lamb     | Colette   | VOR15LYI |  10000.00 | B
209658 | West     | Kato      | ELU81XAA |  10000.00 | S
212395 | Greer    | Harlan    | GXO71KOC |  12624.00 | S
212395 | Perry    | Cora      | YWR73YNZ |  12624.00 | B
215156 | Banks    | Patrick   | ZNQ69CLT |  10000.00 | S
215156 | Hayden   | Malachi   | BBG56AKU |  10000.00 | B
(6 rows)
```

O exemplo a seguir usa um operador UNION ALL porque se forem encontradas linhas duplicadas, elas devem ser mantidas no resultado. Para uma série específica de IDs de evento, a consulta retorna 0 ou mais linhas para cada venda associada a cada evento, e 0 ou 1 linha para cada lista desse evento. Os IDs de evento são exclusivos para cada linha nas tabelas LISTING e EVENT, mas pode haver várias vendas para a mesma combinação de IDs de evento e de lista na tabela SALES.

A terceira coluna no conjunto de resultados identifica a origem da linha. Se vier da tabela SALES, “Yes” é marcado na coluna SALESROW. (SALESROW é um alias para SALES.LISTID.) Se a linha vier da tabela LISTING, “No” é marcado na coluna SALESROW.

Nesse caso, o conjunto de resultados consiste em três linhas de vendas para a lista 500, evento 7787. Em outras palavras, três transações diferentes ocorreram para essa combinação de lista e evento. Outras duas listas, 501 e 502, não produziram vendas. Dessa forma, a única linha que a consulta produz para esses IDs de lista vem de tabela LISTING (SALESROW = "No").

```
select eventid, listid, 'Yes' as salesrow
from sales
where listid in(500,501,502)
union all
select eventid, listid, 'No'
from listing
where listid in(500,501,502)
order by listid asc;

eventid | listid | salesrow
---------+--------+----------
```

```
7787 |    500 | No
7787 |    500 | Yes
7787 |    500 | Yes
7787 |    500 | Yes
6473 |    501 | No
5108 |    502 | No
(6 rows)
```

Se você executar a mesma consulta sem a palavra-chave ALL, o resultado manterá somente uma das transações de vendas.

```
select eventid, listid, 'Yes' as salesrow
from sales
where listid in(500,501,502)
union
select eventid, listid, 'No'
from listing
where listid in(500,501,502)
order by listid asc;

eventid | listid | salesrow
---------+--------+----------
7787 |    500 | No
7787 |    500 | Yes
6473 |    501 | No
5108 |    502 | No
(4 rows)
```

Exemplos de consultas UNION ALL

O exemplo a seguir usa um operador UNION ALL porque se forem encontradas linhas duplicadas, elas devem ser mantidas no resultado. Para uma série específica de IDs de evento, a consulta retorna 0 ou mais linhas para cada venda associada a cada evento, e 0 ou 1 linha para cada lista desse evento. Os IDs de evento são exclusivos para cada linha nas tabelas LISTING e EVENT, mas pode haver várias vendas para a mesma combinação de IDs de evento e de lista na tabela SALES.

A terceira coluna no conjunto de resultados identifica a origem da linha. Se vier da tabela SALES, “Yes” é marcado na coluna SALESROW. (SALESROW é um alias para SALES.LISTID.) Se a linha vier da tabela LISTING, “No” é marcado na coluna SALESROW.

Nesse caso, o conjunto de resultados consiste em três linhas de vendas para a lista 500, evento 7787. Em outras palavras, três transações diferentes ocorreram para essa combinação de lista e

evento. Outras duas listas, 501 e 502, não produziram vendas. Dessa forma, a única linha que a consulta produz para esses IDs de lista vem de tabela LISTING (SALESROW = "No").

```
select eventid, listid, 'Yes' as salesrow
from sales
where listid in(500,501,502)
union all
select eventid, listid, 'No'
from listing
where listid in(500,501,502)
order by listid asc;

eventid | listid | salesrow
---------+--------+----------
7787 |    500 | No
7787 |    500 | Yes
7787 |    500 | Yes
7787 |    500 | Yes
6473 |    501 | No
5108 |    502 | No
(6 rows)
```

Se você executar a mesma consulta sem a palavra-chave ALL, o resultado manterá somente uma das transações de vendas.

```
select eventid, listid, 'Yes' as salesrow
from sales
where listid in(500,501,502)
union
select eventid, listid, 'No'
from listing
where listid in(500,501,502)
order by listid asc;

eventid | listid | salesrow
---------+--------+----------
7787 |    500 | No
7787 |    500 | Yes
6473 |    501 | No
5108 |    502 | No
(4 rows)
```

Exemplos de consultas INTERSECT

Compare o exemplo a seguir com o primeiro exemplo de UNION. A única diferença entre os dois exemplos é o operador de conjunto usado, mas os resultados são muito diferentes. Somente uma das linhas é a mesma:

```
235494 |    23875 |    8771
```

Essa é a única linha no resultado limitado de 5 linhas encontrada em ambas as tabelas.

```
select listid, sellerid, eventid from listing
intersect
select listid, sellerid, eventid from sales
order by listid desc, sellerid, eventid
limit 5;

listid | sellerid | eventid
--------+----------+---------
235494 |    23875 |    8771
235482 |     1067 |    2667
235479 |     1589 |    7303
235476 |    15550 |     793
235475 |    22306 |    7848
(5 rows)
```

A consulta a seguir encontra eventos (em que foram vendidos ingressos) que ocorreram em locais em Nova York e Los Angeles em março. A diferença entre as duas expressões de consulta é a restrição na coluna VENUECITY.

```
select distinct eventname from event, sales, venue
where event.eventid=sales.eventid and event.venueid=venue.venueid
and date_part(month,starttime)=3 and venuecity='Los Angeles'
intersect
select distinct eventname from event, sales, venue
where event.eventid=sales.eventid and event.venueid=venue.venueid
and date_part(month,starttime)=3 and venuecity='New York City'
order by eventname asc;

eventname
----------------------------
A Streetcar Named Desire
Dirty Dancing
```

```
Electra
Running with Annalise
Hairspray
Mary Poppins
November
Oliver!
Return To Forever
Rhinoceros
South Pacific
The 39 Steps
The Bacchae
The Caucasian Chalk Circle
The Country Girl
Wicked
Woyzeck
(16 rows)
```

Exemplos de consultas EXCEPT

A tabela CATEGORY no banco de dados TICKIT contém as seguintes 11 linhas:

```
 catid | catgroup |  catname  |                 catdesc
-------+----------+-----------+--------------------------------------------
   1   | Sports   | MLB       | Major League Baseball
   2   | Sports   | NHL       | National Hockey League
   3   | Sports   | NFL       | National Football League
   4   | Sports   | NBA       | National Basketball Association
   5   | Sports   | MLS       | Major League Soccer
   6   | Shows    | Musicals  | Musical theatre
   7   | Shows    | Plays     | All non-musical theatre
   8   | Shows    | Opera     | All opera and light opera
   9   | Concerts | Pop       | All rock and pop music concerts
  10   | Concerts | Jazz      | All jazz singers and bands
  11   | Concerts | Classical | All symphony, concerto, and choir concerts
(11 rows)
```

Pressuponha que uma tabela CATEGORY_STAGE (tabela de preparação) contém uma linha adicional:

```
 catid | catgroup |  catname  |                 catdesc
-------+----------+-----------+--------------------------------------------
1 | Sports   | MLB       | Major League Baseball
2 | Sports   | NHL       | National Hockey League
```

```
3 | Sports   | NFL       | National Football League
4 | Sports   | NBA       | National Basketball Association
5 | Sports   | MLS       | Major League Soccer
6 | Shows    | Musicals  | Musical theatre
7 | Shows    | Plays     | All non-musical theatre
8 | Shows    | Opera     | All opera and light opera
9 | Concerts | Pop       | All rock and pop music concerts
10 | Concerts | Jazz      | All jazz singers and bands
11 | Concerts | Classical | All symphony, concerto, and choir concerts
12 | Concerts | Comedy    | All stand up comedy performances
(12 rows)
```

Retorne a diferença entre as duas tabelas. Em outras palavras, retorne as linhas que estão na tabela CATEGORY_STAGE, mas não na tabela CATEGORY:

```
select * from category_stage
except
select * from category;

catid | catgroup | catname |             catdesc
-------+----------+---------+----------------------------------
12 | Concerts | Comedy  | All stand up comedy performances
(1 row)
```

A consulta equivalente a seguir usa o sinônimo MINUS.

```
select * from category_stage
minus
select * from category;

catid | catgroup | catname |             catdesc
-------+----------+---------+----------------------------------
12 | Concerts | Comedy  | All stand up comedy performances
(1 row)
```

Se você reverter a ordem das expressões SELECT, a consulta não retornará qualquer linha.

Cláusula ORDER BY

Tópicos

- Sintaxe

- Parâmetros
- Observações de uso
- Exemplos com ORDER BY

A cláusula ORDER BY classifica o conjunto de resultados de uma consulta.

Sintaxe

```
[ ORDER BY expression [ ASC | DESC ] ]
[ NULLS FIRST | NULLS LAST ]
[ LIMIT { count | ALL } ]
[ OFFSET start ]
```

Parâmetros

expressão

Expressão que define a ordem de classificação do conjunto de resultados da consulta, geralmente especificando uma ou mais colunas na lista de seleção. Os resultados são obtidos com base na ordem binária UTF-8. Também é possível especificar o seguinte:

- Colunas que não estiverem na lista de seleção
- Expressões formadas por uma ou mais colunas que existem em tabelas referidas pela consulta
- Números ordinais que representam a posição de entradas da lista de seleção (ou a posição das colunas na tabela se não houver lista de seleção)
- Aliases que definem entradas da lista de seleção

Quando a cláusula ORDER BY tiver várias expressões, o conjunto de resultados será classificado de acordo com a primeira expressão, e a segunda expressão será aplicada a linhas que tenham valores correspondentes com os da primeira expressão, e assim por diante.

ASC | DESC

Opção que define a ordem de classificação para a expressão, da seguinte forma:

- ASC: ascendente (por exemplo, de valores numéricos menores para maiores e de "A" a "Z" para strings de caracteres). Se nenhuma opção é especificada, os dados são classificados na ordem ascendente por padrão.
- DESC: descendente (de valores numéricos maiores para menores; de "Z" a "A" para strings).

NULLS FIRST | NULLS LAST

Opção que especifica se valores NULL devem ser classificados primeiro, antes de valores não nulos, ou por último, depois de valores não nulos. Por padrão, os valores NULL são ordenados e classificados por último na ordem ASC e são ordenados e classificados primeiro na ordem DESC.

LIMIT number | ALL

Opção que controla o número de linhas classificadas que a consulta retorna. O número LIMIT deve ser um inteiro positivo. O valor máximo é 2147483647.

LIMIT 0 não retorna linhas. Você pode usar essa sintaxe para fins de teste: para garantir que uma consulta seja executada (sem exibir qualquer linha) ou obter uma lista de colunas de uma tabela. Uma cláusula ORDER BY é redundante se você estiver usando LIMIT 0 para obter uma lista de colunas. O valor padrão é LIMIT ALL.

OFFSET start

Opção que especifica para ignorar o número de linhas antes de start antes de começar a retornar linhas. O número OFFSET deve ser um inteiro positivo. O valor máximo é 2147483647. Quando usadas com a opção de LIMIT, as linhas OFFSET são ignoradas antes de iniciar a contagem de linhas LIMIT que são retornadas. Se a opção LIMIT não for usada, o número de linhas no conjunto de resultados será reduzido para o número de linhas ignoradas. As linhas ignoradas por uma cláusula OFFSET ainda precisam passar por varredura, e pode não ser eficiente usar um valor OFFSET grande.

Observações de uso

Observe o seguinte comportamento esperado com cláusulas ORDER BY:

- Os valores NULL são considerados "mais altos" que todos os demais valores. Com a ordem de classificação crescente padrão, os valores NULL são classificados no final. Para alterar esse comportamento, use a opção NULLS FIRST.
- Quando uma consulta não tiver uma cláusula ORDER BY, o sistema retornará conjuntos de resultados sem uma classificação previsível das linhas. A mesma consulta executada duas vezes pode retornar o conjunto de resultados em uma ordem diferente.
- As opções LIMIT e OFFSET podem ser usadas sem uma cláusula ORDER BY. No entanto, para obter um conjunto consistente de linhas, use essas opções em conjunto com ORDER BY.

- Em qualquer sistema paralelo como o Amazon Redshift, quando uma cláusula ORDER BY não produz uma classificação exclusiva dos dados, a ordem das linhas não é determinística. Portanto, se a expressão ORDER BY produz valores duplicados, a ordem de retorno dessas linhas pode variar de outros sistemas ou de uma execução do Amazon Redshift para outra.
- O Amazon Redshift não oferece suporte a literais de string nas cláusulas ORDER BY.

Exemplos com ORDER BY

Retorne todas as 11 linhas da tabela CATEGORY, classificada pela segunda coluna, CATGROUP. Para os resultados que têm o mesmo valor de CATGROUP, classifique os valores da coluna CATDESC pelo tamanho da string. Depois, organize pelas colunas CATID e CATNAME.

```
select * from category order by 2, length(catdesc), 1, 3;

catid | catgroup |  catname  |                 catdesc
------+----------+-----------+--------------------------------------
10    | Concerts | Jazz      | All jazz singers and bands
9     | Concerts | Pop       | All rock and pop music concerts
11    | Concerts | Classical | All symphony, concerto, and choir conce
6     | Shows    | Musicals  | Musical theatre
7     | Shows    | Plays     | All non-musical theatre
8     | Shows    | Opera     | All opera and light opera
5     | Sports   | MLS       | Major League Soccer
1     | Sports   | MLB       | Major League Baseball
2     | Sports   | NHL       | National Hockey League
3     | Sports   | NFL       | National Football League
4     | Sports   | NBA       | National Basketball Association
(11 rows)
```

Retorne colunas selecionadas da tabela SALES, classificada pelos valores mais altos de QTYSOLD. Limite o resultado às 10 primeiras linhas:

```
select salesid, qtysold, pricepaid, commission, saletime from sales
order by qtysold, pricepaid, commission, salesid, saletime desc
limit 10;

salesid | qtysold | pricepaid | commission |     saletime
--------+---------+-----------+------------+---------------------
15401   |       8 |    272.00 |      40.80 | 2008-03-18 06:54:56
61683   |       8 |    296.00 |      44.40 | 2008-11-26 04:00:23
90528   |       8 |    328.00 |      49.20 | 2008-06-11 02:38:09
```

```
74549   |       8 |    336.00 |      50.40 | 2008-01-19 12:01:21
130232  |       8 |    352.00 |      52.80 | 2008-05-02 05:52:31
55243   |       8 |    384.00 |      57.60 | 2008-07-12 02:19:53
16004   |       8 |    440.00 |      66.00 | 2008-11-04 07:22:31
489     |       8 |    496.00 |      74.40 | 2008-08-03 05:48:55
4197    |       8 |    512.00 |      76.80 | 2008-03-23 11:35:33
16929   |       8 |    568.00 |      85.20 | 2008-12-19 02:59:33
(10 rows)
```

Retorne uma lista de colunas e nenhuma linha usando a sintaxe LIMIT 0:

```
select * from venue limit 0;
venueid | venuename | venuecity | venuestate | venueseats
---------+-----------+-----------+------------+------------
(0 rows)
```

Cláusula CONNECT BY

A cláusula CONNECT BY especifica a relação entre as linhas em uma hierarquia. Você pode usar CONNECT BY para selecionar linhas em uma ordem hierárquica unindo a tabela a ela mesma e processando os dados hierárquicos. Por exemplo, você pode usá-la para percorrer recursivamente um organograma e listar dados.

As consultas hierárquicas são processadas na seguinte ordem:

1. Se a cláusula FROM tiver uma união, ela será processada primeiro.
2. A cláusula CONNECT BY é avaliada.
3. A cláusula WHERE é avaliada.

Sintaxe

```
[START WITH start_with_conditions]
CONNECT BY connect_by_conditions
```

> **Note**
>
> Embora START e CONNECT não sejam palavras reservadas, use identificadores delimitados (aspas duplas) ou AS se estiver usando START e CONNECT como aliases de tabela em sua consulta para evitar falhas no runtime.

```
SELECT COUNT(*)
FROM Employee "start"
CONNECT BY PRIOR id = manager_id
START WITH name = 'John'
```

```
SELECT COUNT(*)
FROM Employee AS start
CONNECT BY PRIOR id = manager_id
START WITH name = 'John'
```

Parâmetros

start_with_conditions

Condições que especificam a(s) linha(s) raiz da hierarquia

connect_by_conditions

Condições que especificam a relação entre as linhas pais e as linhas filhas da hierarquia. Pelo menos uma condição deve ser qualificada com o operador unário usado para se referir à linha pai.

```
PRIOR column = expression
-- or
expression > PRIOR column
```

Operadores

É possível usar os seguintes operadores na consulta CONNECT BY.

LEVEL

Pseudocoluna que retorna o nível da linha atual na hierarquia. Retorna 1 para a linha raiz, 2 para a filha da linha raiz e assim por diante.

PRIOR

Operador unário que avalia a expressão da linha pai da linha atual na hierarquia.

Exemplos

O exemplo a seguir é uma consulta CONNECT BY que retorna o número de funcionários subordinados direta ou indiretamente a John, até o máximo de 4 níveis.

```
SELECT id, name, manager_id
FROM employee
WHERE LEVEL < 4
START WITH name = 'John'
CONNECT BY PRIOR id = manager_id;
```

A seguir é o resultado da consulta.

```
id      name      manager_id
------+----------+--------------
  101     John        100
  102     Jorge       101
  103     Kwaku       101
  110     Liu         101
  201     Sofía       102
  106     Mateo       102
  110     Nikki       103
  104     Paulo       103
  105     Richard     103
  120     Saanvi      104
  200     Shirley     104
  205     Zhang       104
```

Definição da tabela para esse exemplo:

```
CREATE TABLE employee (
   id INT,
   name VARCHAR(20),
   manager_id INT
   );
```

A seguir estão as linhas inseridas na tabela.

```
INSERT INTO employee(id, name, manager_id)  VALUES
(100, 'Carlos', null),
(101, 'John', 100),
```

```
(102, 'Jorge', 101),
(103, 'Kwaku', 101),
(110, 'Liu', 101),
(106, 'Mateo', 102),
(110, 'Nikki', 103),
(104, 'Paulo', 103),
(105, 'Richard', 103),
(120, 'Saanvi', 104),
(200, 'Shirley', 104),
(201, 'Sofía', 102),
(205, 'Zhang', 104);
```

A seguir está um organograma para o departamento de John.

Exemplos de subconsulta

Os exemplos a seguir mostram diferentes maneiras em que subconsultas se encaixam em consultas SELECT. Consulte Exemplos de JOIN para obter outros exemplos de uso de subconsultas.

Subconsulta da lista SELECT

O exemplo a seguir contém um subconsulta na lista SELECT. Esta subconsulta é escalar: retorna somente uma coluna e um valor, que é repetido nos resultados para cada linha retornada da consulta exterior. A consulta compara o valor Q1SALES que a subconsulta computa com valores de vendas de outros dois trimestres (2 e 3) em 2008, como definido pela consulta externa.

```
select qtr, sum(pricepaid) as qtrsales,
(select sum(pricepaid)
from sales join date on sales.dateid=date.dateid
where qtr='1' and year=2008) as q1sales
from sales join date on sales.dateid=date.dateid
where qtr in('2','3') and year=2008
group by qtr
order by qtr;

qtr  |  qtrsales   |   q1sales
-------+-------------+-------------
2     | 30560050.00 | 24742065.00
3     | 31170237.00 | 24742065.00
(2 rows)
```

Subconsulta da cláusula WHERE

O exemplo a seguir contém um subconsulta de tabela na cláusula WHERE. Essa subconsulta produz várias linhas. Nesse caso, as linhas contêm apenas uma coluna, mas as subconsultas da tabela podem conter várias colunas e linhas, assim como qualquer outra tabela.

A consulta encontra os 10 principais vendedores em termos quantidade máxima de ingressos vendidos. A lista dos 10 principais é restringida pela subconsulta, que remove usuários que vivem em cidades onde há locais de venda de ingressos. Essa consulta pode ser gravada de diferentes maneiras. Por exemplo, a subconsulta pode ser regravada como uma junção na consulta principal.

```
select firstname, lastname, city, max(qtysold) as maxsold
from users join sales on users.userid=sales.sellerid
where users.city not in(select venuecity from venue)
group by firstname, lastname, city
order by maxsold desc, city desc
limit 10;

firstname | lastname  |      city      | maxsold
-----------+-----------+----------------+---------
Noah       | Guerrero | Worcester      |       8
Isadora    | Moss     | Winooski       |       8
Kieran     | Harrison | Westminster    |       8
Heidi      | Davis    | Warwick        |       8
Sara       | Anthony  | Waco           |       8
Bree       | Buck     | Valdez         |       8
Evangeline | Sampson  | Trenton        |       8
Kendall    | Keith    | Stillwater     |       8
Bertha     | Bishop   | Stevens Point  |       8
Patricia   | Anderson | South Portland |       8
(10 rows)
```

Subconsultas da cláusula WITH

Consulte Cláusula WITH.

Subconsultas correlacionadas

O exemplo a seguir contém uma subconsulta correlacionada na cláusula WHERE. Esse tipo de subconsulta contém uma ou mais correlações entre as colunas e as colunas produzidas pela consulta externa. Nesse caso, a correlação é `where s.listid=l.listid`. Para cada linha que a consulta externa produz, a subconsulta é executada para qualificar ou desqualificar a linha.

```
select salesid, listid, sum(pricepaid) from sales s
where qtysold=
(select max(numtickets) from listing l
where s.listid=l.listid)
group by 1,2
order by 1,2
limit 5;

salesid | listid |   sum
--------+--------+----------
 27     |     28 | 111.00
 81     |    103 | 181.00
 142    |    149 | 240.00
 146    |    152 | 231.00
 194    |    210 | 144.00
(5 rows)
```

Padrões de subconsultas correlacionadas não compatíveis

O planejador de consultas usa um método de regravação de consulta chamado decorrelação de subconsultas para otimizar vários padrões de subconsultas correlacionadas para execução em um ambiente de processamento paralelo massivo (MPP). Alguns tipos de subconsultas correlacionadas seguem padrões cuja correlação o Amazon Redshift não pode anular e aos quais ele não oferece suporte. Consultas que contenham erros de retorno das seguintes referências de correlação:

- Referências de correlação que ignoram um bloco de consultas, também conhecidas como "referências de correlação para ignorar consultas". Por exemplo, na consulta a seguir, o bloco contendo a referência de correlação e o bloco ignorado estão conectados por um predicado NOT EXISTS:

  ```
  select event.eventname from event
  where not exists
  (select * from listing
  where not exists
  (select * from sales where event.eventid=sales.eventid));
  ```

  O bloco ignorado nesse caso é a subconsulta na tabela LISTING. A referência de correlação correlaciona as tabelas EVENT e SALES.
- Referências de correlação de uma subconsulta que é parte de uma cláusula ON em uma consulta externa:

```
select * from category
left join event
on category.catid=event.catid and eventid =
(select max(eventid) from sales where sales.eventid=event.eventid);
```

A cláusula ON contém uma referência de correlação de SALES na subconsulta de EVENT na consulta externa.

- Referências de correlação nulo-sensíveis a uma tabela do sistema Amazon Redshift. Por exemplo:

```
select attrelid
from stv_locks sl, pg_attribute
where sl.table_id=pg_attribute.attrelid and 1 not in
(select 1 from pg_opclass where sl.lock_owner = opcowner);
```

- Referências de correlação de dentro de uma subconsulta que contém uma função de janela.

```
select listid, qtysold
from sales s
where qtysold not in
(select sum(numtickets) over() from listing l where s.listid=l.listid);
```

- Referências em uma coluna GROUP BY para os resultados de um subconsulta correlacionada. Por exemplo:

```
select listing.listid,
(select count (sales.listid) from sales where sales.listid=listing.listid) as list
from listing
group by list, listing.listid;
```

- Referências de correlação de uma subconsulta com uma função agregada e uma cláusula GROUP BY, conectada à consulta externa por um predicado IN. (Essa restrição não se aplica a funções agregadas MIN e MAX.) Por exemplo:

```
select * from listing where listid in
(select sum(qtysold)
from sales
where numtickets>4
group by salesid);
```

SELECT INTO

Seleciona as linhas definidas por qualquer consulta e as introduz em uma nova tabela. Você pode especificar se deseja criar uma tabela temporário ou persistente.

Sintaxe

```
[ WITH with_subquery [, ...] ]
SELECT
[ TOP number | [ ALL | DISTINCT ]
* | expression [ AS output_name ] [, ...] ]
[ EXCLUDE column_list ]
INTO [ TEMPORARY | TEMP ] [ TABLE ] new_table
[ FROM table_reference [, ...] ]
[ WHERE condition ]
[ [ START WITH expression ] CONNECT BY expression ]
[ GROUP BY ALL | expression [, ...] ]
[ HAVING condition ]
[ QUALIFY condition ]
[ { UNION | ALL | INTERSECT | EXCEPT | MINUS } query ]
[ ORDER BY expression [ ASC | DESC ] ]
[ LIMIT { number | ALL } ]
[ OFFSET start ]
```

Para obter detalhes sobre os parâmetros deste comando, consulte SELECT.

Exemplos

Selecione todas as linhas da tabela EVENT e crie uma tabela NEWEVENT:

```
select * into newevent from event;
```

Selecione o resultado de uma consulta agregada em uma tabela temporária chamada PROFITS:

```
select username, lastname, sum(pricepaid-commission) as profit
into temp table profits
from sales, users
where sales.sellerid=users.userid
group by 1, 2
order by 3 desc;
```

SET

Define o valor de um parâmetro de configuração de servidor. Use o comando SET para substituir uma configuração para a duração da sessão ou transação atual somente.

Usa o comando RESET para que um parâmetro volte ao seu valor padrão.

É possível alterar os parâmetros de configuração do servidor de várias maneiras. Para obter mais informações, consulte Modificar a configuração do servidor.

Sintaxe

```
SET { [ SESSION | LOCAL ]
{ SEED | parameter_name } { TO | = }
{ value | 'value' | DEFAULT } |
SEED TO value }
```

A instrução a seguir define o valor de uma variável de contexto de sessão.

```
SET { [ SESSION | LOCAL ]
variable_name { TO | = }
{ value | 'value'  }
```

Parâmetros

SESSION

Especifica que a configuração é válida para a sessão atual. Valor padrão.

variable_name

Especifica o nome da variável de contexto definida para a sessão.

A convenção de nomenclatura é um nome de duas partes separado por um ponto; por exemplo identificador.identificador. Só é permitido um separador de ponto. Use um identificador que siga as regras de identificador padrão do Amazon Redshift. Para obter mais informações, consulte Nomes e identificadores. Não são permitidos identificadores delimitados.

LOCAL

Especifica que a configuração é válida para a transação atual.

SEED TO valor

Define uma propagação interna a ser usada pela função RANDOM para a geração de números aleatórios.

SET SEED escolhe um valor numérico entre 0 e 1 e multiplica esse número por (2^{31}-1) para uso com a função Função RANDOM. Se você usar SET SEED antes de fazer diversas chamadas RANDOM, RANDOM gerará números em uma sequência previsível.

parameter_name

Nome do parâmetro a ser definido. Consulte Modificar a configuração do servidor para obter informações sobre parâmetros.

valor

Novo valor do parâmetro. Use aspas simples para definir o valor de uma string específica. Se estiver usando SET SEED, esse parâmetro contém o valor SEED.

DEFAULT

Define o parâmetro para o valor padrão.

Exemplos

Alterar um parâmetro para a sessão atual

O exemplo a seguir define datestyle:

```
set datestyle to 'SQL,DMY';
```

Configurar um grupo de consultas para gerenciamento de workload

Se grupos de consulta estiverem listados em uma definição de fila como parte da configuração de WLM do cluster, configure o parâmetro QUERY_GROUP para um nome de grupo de consultas listado. Consultas subsequentes são atribuídas à fila de consultas associada. A configuração de QUERY_GROUP permanece em vigor pela duração da sessão ou até que o comando RESET QUERY_GROUP seja encontrado.

Este exemplo executa duas consultas como parte de 'priority' do grupo de consultas e, depois, redefine o grupo de consultas.

```
set query_group to 'priority';
select tbl, count(*)from stv_blocklist;
select query, elapsed, substring from svl_qlog order by query desc limit 5;
reset query_group;
```

Para obter mais informações, consulte Gerenciamento do workload.

Alterar o namespace de identidade padrão para a sessão

Um usuário do banco de dados pode definir `default_identity_namespace`. Este exemplo mostra como usar `SET SESSION` para substituir a configuração durante a sessão atual e, depois, mostrar o novo valor do provedor de identidades. Isso é usado com maior frequência quando você utiliza um provedor de identidades com o Redshift e o Centro de Identidade do IAM. Para ter mais informações sobre como usar um provedor de identidades com o Redshift, consulte Conectar o Redshift ao IAM Identity Center para proporcionar aos usuários uma experiência de logon único.

```
SET SESSION default_identity_namespace = 'MYCO';

SHOW default_identity_namespace;
```

Após a execução do comando, é possível executar uma declaração GRANT ou CREATE como a seguinte:

```
GRANT SELECT ON TABLE mytable TO alice;

GRANT UPDATE ON TABLE mytable TO salesrole;

CREATE USER bob password 'md50c983d1a624280812631c5389e60d48c';
```

Nesse caso, o efeito de definir o namespace de identidade padrão é equivalente a prefixar cada identidade com o namespace. Neste exemplo, `alice` é substituído por `MYCO:alice`. Para ter mais informações sobre definições relativas à configuração do Centro de Identidade do IAM, consulte ALTER SYSTEM e ALTER IDENTITY PROVIDER.

Configurar uma identificação para um grupo de consultas

O parâmetro QUERY_GROUP define um rótulo para uma ou mais consultas que são executadas na mesma sessão após um comando SET. Por sua vez, esse rótulo é registrado quando as consultas são executadas e pode ser usado para restringir os resultados retornados das tabelas de sistema STL_QUERY e STV_INFLIGHT e da visualização SVL_QLOG.

```
show query_group;
query_group
-------------
unset
(1 row)

set query_group to '6 p.m.';


show query_group;
query_group
-------------
6 p.m.
(1 row)

select * from sales where salesid=500;
salesid | listid | sellerid | buyerid | eventid | dateid | ...
---------+--------+----------+---------+---------+--------+-----
500 |    504 |     3858 |    2123 |    5871 |   2052 | ...
(1 row)

reset query_group;

select query, trim(label) querygroup, pid, trim(querytxt) sql
from stl_query
where label ='6 p.m.';
query | querygroup |  pid  |                  sql
-------+------------+-------+---------------------------------------
57 | 6 p.m.     | 30711 | select * from sales where salesid=500;
(1 row)
```

As identificações de grupo de consulta são um mecanismo útil para isolar consultas individuais ou grupos de consultas executadas como parte de scripts. Você não precisa identificar e monitorar consultas pelos IDs; você pode acompanhá-las por suas identificações.

Configurar um valor de seed para a geração de números aleatórios

O exemplo a seguir usa a opção SEED com SET para que a função RANDOM gere números em uma sequência previsível.

Primeiro, retorne três inteiros RANDOM sem definir o valor de SEED primeiro:

```
select cast (random() * 100 as int);
```

```
int4
------
6
(1 row)

select cast (random() * 100 as int);
int4
------
68
(1 row)

select cast (random() * 100 as int);
int4
------
56
(1 row)
```

Agora, defina o valor de SEED como `.25` e retorne mais três números RANDOM:

```
set seed to .25;

select cast (random() * 100 as int);
int4
------
21
(1 row)

select cast (random() * 100 as int);
int4
------
79
(1 row)

select cast (random() * 100 as int);
int4
------
12
(1 row)
```

Por fim, redefina o valor de SEED como `.25` e verifique se RANDOM retorna os mesmos resultados que as três chamadas anteriores:

```
set seed to .25;
```

```
select cast (random() * 100 as int);
int4
------
21
(1 row)

select cast (random() * 100 as int);
int4
------
79
(1 row)

select cast (random() * 100 as int);
int4
------
12
(1 row)
```

O exemplo a seguir define uma variável de contexto personalizada.

```
SET app_context.user_id TO 123;
SET app_context.user_id TO 'sample_variable_value';
```

SET SESSION AUTHORIZATION

Define o nome do usuário da sessão atual.

Você pode usar o comando SET SESSION AUTHORIZATION, por exemplo, para testar o acesso ao banco de dados temporariamente executando uma sessão ou uma transação como um usuário sem privilégios. Você deve ser um superusuário do banco de dados para executar este comando.

Sintaxe

```
SET [ LOCAL ] SESSION AUTHORIZATION { user_name | DEFAULT }
```

Parâmetros

LOCAL

Especifica que a configuração é válida para a transação atual. Omitir esse parâmetro especifica que a configuração é válida para a sessão atual.

user_name

Nome do usuário a ser definido. O nome de usuário pode ser gravado como um identificador ou uma string literal.

DEFAULT

Define o nome de usuário da sessão como o valor padrão.

Exemplos

O exemplo a seguir define o nome do usuário da sessão atual como `dwuser`:

```
SET SESSION AUTHORIZATION 'dwuser';
```

O exemplo a seguir define o nome do usuário da transação atual como `dwuser`:

```
SET LOCAL SESSION AUTHORIZATION 'dwuser';
```

Este exemplo define o nome do usuário da sessão atual como o nome de usuário padrão:

```
SET SESSION AUTHORIZATION DEFAULT;
```

SET SESSION CHARACTERISTICS

Este comando está obsoleto.

SHOW

Exibe o valor atual de um parâmetro de configuração de servidor. Esse valor pode ser específico de uma sessão atual se o comando SET estiver em vigor. Para obter uma lista dos parâmetros de configuração, consulte Referência da configuração.

Sintaxe

```
SHOW { parameter_name | ALL }
```

A instrução a seguir exibe o valor atual de uma variável de contexto de sessão. Se a variável não existir, o Amazon Redshift lançará um erro.

```
SHOW variable_name
```

Parâmetros

parameter_name

Exibe o valor atual do parâmetro especificado.

ALL

Exibe os valores atuais de todos os parâmetros.

variable_name

Exibe o valor atual da variável especificada.

Exemplos

O exemplo a seguir exibe o valor para o parâmetro de query_group:

```
show query_group;

query_group

unset
(1 row)
```

O exemplo a seguir exibe uma lista de todos os parâmetros e seus valores:

```
show all;
name        |   setting
--------------------+--------------
datestyle          | ISO, MDY
extra_float_digits | 0
query_group        | unset
search_path        | $user,public
statement_timeout  | 0
```

O exemplo a seguir exibe o valor atual da variável especificada.

```
SHOW app_context.user_id;
```

SHOW USER LOCKOUT

Use SHOW USER LOCKOUT para auditar quais usuários estão bloqueados no momento, determinar se cada bloqueio foi automático ou manual e revisar a atividade recente de login com falha antes de desbloquear um usuário. Usuários federados não são incluídos. Esse comando está disponível somente para superusuários. Para obter mais informações, consulte max_failed_login_attempts.

Privilégios obrigatórios

Para usar SHOW USER LOCKOUT, é necessário ser um superusuário.

Sintaxe

```
SHOW USER LOCKOUT
```

Colunas de saída

user_name

O nome do usuário do banco de dados.

is_locked

Especifica se o usuário está bloqueado no momento.

lock_reason

O motivo pelo qual o usuário foi bloqueado. O valor é `automatic` quando o Amazon Redshift bloqueia o usuário depois de atingir o limite de tentativas com falha ou `manual` quando um superusuário bloqueia o usuário com ALTER USER NOLOGIN. Esse valor é nulo quando o usuário não está bloqueado.

locked_at

O carimbo de data/hora de quando o usuário foi bloqueado.

last_failed_at

O carimbo de data/hora da tentativa de login com falha mais recente.

failed_login_attempts

O número de tentativas consecutivas de login com falha.

locked_by

O ID do usuário que bloqueou esse usuário. Esse valor é nulo para bloqueio automático.

Exemplos

O exemplo a seguir exibe o status de bloqueio para todos os usuários baseados em senha:

```
SHOW USER LOCKOUT;

 user_name    | is_locked | lock_reason      | locked_at           | last_failed_at
  | failed_login_attempts | locked_by
--------------+-----------+------------------+---------------------
+---------------------+-----------------------+-----------
 data_analyst | true      | automatic        | 2026-08-10 14:02:11 | 2026-08-10
 14:02:11 |                     5 |
 sales_user   | true      | manual           | 2026-08-10 09:15:44 |
  |                     0 |       100
 report_user  | false     |                  |                     | 2026-08-09
 22:31:07 |                     2 |
(3 rows)
```

SHOW COLUMN GRANTS

Exibe concessões em uma coluna dentro de uma tabela.

Permissões obrigatórias

SHOW GRANTS para um objeto de destino exibirá somente concessões visíveis para o usuário atual. Uma concessão ficará visível para o usuário atual se o usuário atual atender a um dos seguintes critérios:

- Ser um superusuário.
- Ser o usuário que recebeu a concessão.
- Receber o título de proprietário do perfil concedido.
- Receber o perfil pretendido pela concessão do objeto.

Sintaxe

```
SHOW COLUMN GRANTS ON TABLE
```

```
{ database_name.schema_name.table_name | schema_name.table_name }
[FOR {username | ROLE role_name | PUBLIC}]
[LIMIT row_limit]
```

Parâmetros

database_name

O nome do banco de dados que contém a tabela de destino.

schema_name

O nome do esquema que contém a tabela de destino.

table_name

O nome da tabela de destino.

username

Inclua apenas concessões ao nome de usuário na saída.

role_name

Inclua apenas concessões a role_name na saída.

PUBLIC

Inclua apenas concessões a PUBLIC na saída.

row_limit

O número máximo de linhas a serem retornadas. O row_limit pode ser de 0 a 10.000.

Exemplos

O exemplo a seguir mostra as concessões de colunas na tabela demo_db.demo_schema.t100:

```
SHOW COLUMN GRANTS ON TABLE demo_db.demo_schema.t100;
 database_name | schema_name | table_name | column_name | object_type | privilege_type
 | identity_id | identity_name | identity_type | admin_option | privilege_scope |
 grantor_name
---------------+-------------+------------+-------------+-------------+---------------
+-------------+---------------+---------------+--------------+-----------------
+--------------
```

```
 demo_db       | demo_schema | t100       | b           | COLUMN      | UPDATE
    |         134 | bob           | user          | f            | COLUMN          |
 dbadmin
 demo_db       | demo_schema | t100       | a           | COLUMN      | SELECT
    |         130 | alice         | user          | f            | COLUMN          |
 dbadmin
 demo_db       | demo_schema | t100       | a           | COLUMN      | UPDATE
    |         130 | alice         | user          | f            | COLUMN          |
 dbadmin
```

O exemplo a seguir mostra as concessões de colunas na tabela demo_schema.t100 para o bob do usuário:

```
SHOW COLUMN GRANTS ON TABLE demo_schema.t100 for bob;
 database_name | schema_name | table_name | column_name | object_type | privilege_type
 | identity_id | identity_name | identity_type | admin_option | privilege_scope |
 grantor_name
---------------+-------------+------------+-------------+-------------+---------------
+-------------+---------------+---------------+--------------+-----------------
+--------------
 demo_db       | demo_schema | t100       | b           | COLUMN      | UPDATE
    |         135 | bob           | user          | f            | COLUMN          |
 dbadmin
```

SHOW COLUMNS

Mostra uma lista de colunas em uma tabela, bem como alguns atributos da coluna.

Cada linha de saída é composta dos seguintes elementos: lista separada por vírgula do nome do banco de dados, nome do esquema, nome da tabela, nome da coluna, posição ordinal, padrão da coluna, se anulável, tipo de dados, extensão máxima de caracteres, precisão numérica, comentários, tipo dechave de classificação, ordem de chave de classificação, chave de distribuição, codificação e agrupamento. Para obter mais informações sobre esses atributos, consulte SVV_ALL_COLUMNS.

Se mais de 10.000 colunas resultarem do comando SHOW COLUMNS, será retornado um erro.

Permissões obrigatórias

Para visualizar uma coluna em uma tabela do Amazon Redshift, o usuário atual deve atender a um dos seguintes critérios:

- Ser um superusuário.

- Ser o proprietário da tabela.
- Ter recebido o privilégio USAGE no esquema principal e o privilégio SELECT na tabela ou ter recebido o privilégio SELECT na coluna.

Sintaxe

```
SHOW COLUMNS FROM TABLE database_name.schema_name.table_name [LIKE 'filter_pattern']
 [LIMIT row_limit ]
```

Parâmetros

database_name

O nome do banco de dados que contém as tabelas a serem listadas.

Para mostrar tabelas em um AWS Glue Data Catalog, especifique (`awsdatacatalog`) como o nome do banco de dados e verifique se a configuração do sistema `data_catalog_auto_mount` está definida como `true`. Para obter mais informações, consulte ALTER SYSTEM.

schema_name

O nome do esquema que contém as tabelas a serem listadas.

Para mostrar tabelas do AWS Glue Data Catalog, forneça o nome do banco de dados do AWS Glue como nome do esquema.

table_name

O nome da tabela que contém as colunas a listar.

filter_pattern

Uma expressão de caractere UTF-8 válida com o padrão para estabelecer correspondência com os nomes de coluna. A opção LIKE executa uma correspondência com distinção entre letras maiúsculas e minúsculas compatível com os seguintes metacaracteres de correspondência de padrões:

Metacaractere	Descrição
%	Corresponde a qualquer sequência de zero ou mais caracteres.

Metacaractere	Descrição
_	Corresponde a qualquer caractere único.

Se filter_pattern não contiver metacaracteres, o padrão representará somente a própria string. Nesse caso, LIKE age da mesma forma que o operador de igualdade.

row_limit

O número máximo de linhas a serem retornadas. O row_limit pode ser de 0 a 10.000.

Exemplos

O exemplo a seguir mostra as colunas no banco de dados do Amazon Redshift chamado `sample_data_dev` que estão no esquema `tickit` e na tabela `event`.

```
SHOW COLUMNS FROM TABLE demo_schema.compound_sort_table;

  database_name | schema_name |     table_name      | column_name | ordinal_position
 | column_default | is_nullable |     data_type     | character_maximum_length |
 numeric_precision | numeric_scale | remarks | sort_key_type | sort_key | dist_key |
 encoding | collation
---------------+-------------+--------------------+-------------+------------------
+----------------+-------------+-------------------+--------------------------
+------------------+--------------+---------+--------------+----------+----------
+----------+-----------
 demo_db       | demo_schema | compound_sort_table | id          |                1 |
              | YES         | integer             |                           |
     32 |             0 |         | COMPOUND      |        1 |        1 | delta32k |
 demo_db       | demo_schema | compound_sort_table | name        |                2 |
              | YES         | character varying |                        50 |
          |               |         | COMPOUND      |        2 |          | lzo      |
 default
 demo_db       | demo_schema | compound_sort_table | date_col    |                3 |
              | YES         | date                |                           |
       |               |         |               |        0 |          | delta    |
 demo_db       | demo_schema | compound_sort_table | amount      |                4 |
              | YES         | numeric             |                           |
     10 |             2 |         |               |        0 |          | mostly16 |
```

O exemplo a seguir mostra as tabelas no banco de dados do AWS Glue Data Catalog chamado `awsdatacatalog` que estão no esquema `batman` e na tabela `nation`. A saída é limitada a 2 linhas.

```
SHOW COLUMNS FROM TABLE second_db.public.t22;

 database_name | schema_name | table_name | column_name | ordinal_position |
 column_default | is_nullable |          data_type          | character_maximum_length
 | numeric_precision | numeric_scale | remarks | sort_key_type | sort_key | dist_key |
 encoding | collation
---------------+-------------+------------+-------------+------------------
+----------------+-------------+-----------------------------
+--------------------------+------------------+---------------+---------
+---------------+----------+----------+----------+-----------
 second_db     | public      | t22        | col1        |                1 |
     | YES         | integer                     |                           |
      32 |             0 |         | INTERLEAVED   |       -1 |          | mostly8  |
 second_db     | public      | t22        | col2        |                2 |
      | YES         | character varying           |                     100 |
           |               |         | INTERLEAVED   |        2 |          | text255  |
 default
 second_db     | public      | t22        | col3        |                3 |
     | YES         | timestamp without time zone |                           |
         |               |         |               |        0 |          | raw      |
 second_db     | public      | t22        | col4        |                4 |
     | YES         | numeric                     |                           |
      10 |             2 |         |               |        0 |          | az64     |
```

SHOW CONSTRAINTS

Mostra uma lista de restrições da chave primária e da chave externa em uma tabela.

Permissões obrigatórias

Para executar SHOW CONSTRAINTS em uma tabela, o usuário atual deve atender a um dos seguintes critérios:

- Ser um superusuário.
- Ser o proprietário da tabela.
- Ter recebido o privilégio USAGE no esquema principal e o privilégio USAGE na tabela.

Sintaxe

```
SHOW CONSTRAINTS {PRIMARY KEYS | FOREIGN KEYS [EXPORTED]}
FROM TABLE
{ database_name.schema_name.table_name | schema_name.table_name }
[LIMIT row_limit]
```

Parâmetros

database_name

O nome do banco de dados que contém a tabela de destino.

schema_name

O nome do esquema que contém a tabela de destino.

table_name

O nome da tabela de destino.

EXPORTED

Quando EXPORTED for especificado, liste todas as chaves externas de outras tabelas que fazem referência à tabela de destino.

row_limit

O número máximo de linhas a serem retornadas. O row_limit pode ser de 0 a 10.000.

Exemplos

O exemplo a seguir mostra as restrições da chave primária da tabela demo_db.demo_schema.pk1:

```
SHOW CONSTRAINTS PRIMARY KEYS FROM TABLE demo_db.demo_schema.pk1;
 database_name | schema_name | table_name | pk_name  | column_name | key_seq
---------------+-------------+------------+----------+-------------+---------
 demo_db       | demo_schema | pk1        | pk1_pkey | i           |       1
 demo_db       | demo_schema | pk1        | pk1_pkey | j           |       2
 demo_db       | demo_schema | pk1        | pk1_pkey | c           |       3
```

O exemplo a seguir mostra as restrições da chave externa da tabela demo_schema.fk2:

```
SHOW CONSTRAINTS FOREIGN KEYS FROM TABLE demo_schema.fk2;
```

```
 pk_database_name | pk_schema_name | pk_table_name | pk_column_name | fk_database_name
 | fk_schema_name | fk_table_name | fk_column_name | key_seq |  fk_name   | pk_name  |
 update_rule | delete_rule | deferrability
-----------------+----------------+---------------+----------------+-----------------
+----------------+---------------+----------------+---------+-----------+----------
+-------------+-------------+--------------
 demo_db          | demo_schema    | pk1           | i              | demo_db
 | demo_schema    | fk2           | i              |       1 | fk2_i_fkey | pk1_pkey |
           |             |
 demo_db          | demo_schema    | pk1           | j              | demo_db
 | demo_schema    | fk2           | j              |       2 | fk2_i_fkey | pk1_pkey |
           |             |
 demo_db          | demo_schema    | pk1           | c              | demo_db
 | demo_schema    | fk2           | c              |       3 | fk2_i_fkey | pk1_pkey |
           |             |
```

O exemplo a seguir mostra as restrições da chave externa exportada da tabela demo_schema.pk1:

```
SHOW CONSTRAINTS FOREIGN KEYS EXPORTED FROM TABLE demo_schema.pk1;
 pk_database_name | pk_schema_name | pk_table_name | pk_column_name | fk_database_name
 | fk_schema_name | fk_table_name | fk_column_name | key_seq |     fk_name     |
 pk_name  | update_rule | delete_rule | deferrability
-----------------+----------------+---------------+----------------+-----------------
+----------------+---------------+----------------+---------+-----------------
+----------+-------------+-------------+--------------
 demo_db          | demo_schema    | pk1           | i              | demo_db
    | demo_schema    | fk2           | i              |       1 | fk2_i_fkey      |
 pk1_pkey |             |             |
 demo_db          | demo_schema    | pk1           | j              | demo_db
    | demo_schema    | fk2           | j              |       2 | fk2_i_fkey      |
 pk1_pkey |             |             |
 demo_db          | demo_schema    | pk1           | c              | demo_db
    | demo_schema    | fk2           | c              |       3 | fk2_i_fkey      |
 pk1_pkey |             |             |
 demo_db          | demo_schema    | pk1           | i              | demo_db
    | demo_schema    | other_fk      | i              |       1 | other_fk_i_fkey |
 pk1_pkey |             |             |
 demo_db          | demo_schema    | pk1           | j              | demo_db
    | demo_schema    | other_fk      | j              |       2 | other_fk_i_fkey |
 pk1_pkey |             |             |
 demo_db          | demo_schema    | pk1           | c              | demo_db
    | demo_schema    | other_fk      | c              |       3 | other_fk_i_fkey |
 pk1_pkey |             |             |
```

SHOW EXTERNAL TABLE

Exibe a definição de uma tabela externa, incluindo atributos de tabela e atributos de coluna. Use a saída da instrução SHOW EXTERNAL TABLE para recriar a tabela.

Para obter mais informações sobre a criação de tabelas externas, consulte CREATE EXTERNAL TABLE.

Sintaxe

```
SHOW EXTERNAL TABLE [external_database].external_schema.table_name [ PARTITION ]
```

Parâmetros

external_database

O nome do banco de dados externo associado. Esse parâmetro é opcional.

external_schema

Nome do novo esquema externo associado.

table_name

O nome da tabela a ser exibida.

PARTITION

Exibe instruções ALTER TABLE para adicionar partições à definição de tabela.

Exemplos

Os exemplos a seguir são baseados em uma tabela externa definida da seguinte forma:

```
CREATE EXTERNAL TABLE my_schema.alldatatypes_parquet_test_partitioned (
     csmallint smallint,
     cint int,
     cbigint bigint,
     cfloat float4,
     cdouble float8,
     cchar char(10),
```

```
     cvarchar varchar(255),
     cdecimal_small decimal(18,9),
     cdecimal_big decimal(30,15),
     ctimestamp TIMESTAMP,
     cboolean boolean,
     cstring varchar(16383)
)
PARTITIONED BY (cdate date, ctime TIMESTAMP)
STORED AS PARQUET
LOCATION 's3://amzn-s3-demo-bucket/alldatatypes_parquet_partitioned';
```

Segue-se um exemplo do comando e saída SHOW EXTERNAL TABLE para a tabela `my_schema.alldatatypes_parquet_test_partitioned`.

```
SHOW EXTERNAL TABLE my_schema.alldatatypes_parquet_test_partitioned;
```

```
"CREATE EXTERNAL TABLE my_schema.alldatatypes_parquet_test_partitioned (
    csmallint smallint,
    cint int,
    cbigint bigint,
    cfloat float4,
    cdouble float8,
    cchar char(10),
    cvarchar varchar(255),
    cdecimal_small decimal(18,9),
    cdecimal_big decimal(30,15),
    ctimestamp timestamp,
    cboolean boolean,
    cstring varchar(16383)
)
PARTITIONED BY (cdate date, ctime timestamp)
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS INPUTFORMAT 'org.apache.hadoop.hive.ql.io.parquet.MapredParquetInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat'
LOCATION 's3://amzn-s3-demo-bucket/alldatatypes_parquet_partitioned';"
```

Veja um exemplo do comando SHOW EXTERNAL TABLE e a saída para a mesma tabela definida, mas com o banco de dados também especificado no parâmetro.

```
SHOW EXTERNAL TABLE my_database.my_schema.alldatatypes_parquet_test_partitioned;
```

```
"CREATE EXTERNAL TABLE my_database.my_schema.alldatatypes_parquet_test_partitioned (
    csmallint smallint,
    cint int,
    cbigint bigint,
    cfloat float4,
    cdouble float8,
    cchar char(10),
    cvarchar varchar(255),
    cdecimal_small decimal(18,9),
    cdecimal_big decimal(30,15),
    ctimestamp timestamp,
    cboolean boolean,
    cstring varchar(16383)
)
PARTITIONED BY (cdate date, ctime timestamp)
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS INPUTFORMAT 'org.apache.hadoop.hive.ql.io.parquet.MapredParquetInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat'
LOCATION 's3://amzn-s3-demo-bucket/alldatatypes_parquet_partitioned';"
```

Segue-se um exemplo do comando e saída SHOW EXTERNAL TABLE ao usar o parâmetro PARTITION. A saída contém instruções ALTER TABLE para adicionar partições à definição de tabela.

```
SHOW EXTERNAL TABLE my_schema.alldatatypes_parquet_test_partitioned PARTITION;
```

```
"CREATE EXTERNAL TABLE my_schema.alldatatypes_parquet_test_partitioned (
    csmallint smallint,
    cint int,
    cbigint bigint,
    cfloat float4,
    cdouble float8,
    cchar char(10),
    cvarchar varchar(255),
    cdecimal_small decimal(18,9),
    cdecimal_big decimal(30,15),
    ctimestamp timestamp,
    cboolean boolean,
    cstring varchar(16383)
)
PARTITIONED BY (cdate date)
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
```

```
STORED AS INPUTFORMAT 'org.apache.hadoop.hive.ql.io.parquet.MapredParquetInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat'
LOCATION 's3://amzn-s3-demo-bucket/alldatatypes_parquet_partitioned';
ALTER TABLE my_schema.alldatatypes_parquet_test_partitioned ADD IF NOT
 EXISTS PARTITION (cdate='2021-01-01') LOCATION 's3://amzn-s3-demo-bucket/
alldatatypes_parquet_partitioned2/cdate=2021-01-01';
ALTER TABLE my_schema.alldatatypes_parquet_test_partitioned ADD IF NOT
 EXISTS PARTITION (cdate='2021-01-02') LOCATION 's3://amzn-s3-demo-bucket/
alldatatypes_parquet_partitioned2/cdate=2021-01-02';"
```

SHOW DATABASES

Exibe bancos de dados de um catálogo de dados ou de um data warehouse do Amazon Redshift. SHOW DATABASES mostra todos os bancos de dados acessíveis; por exemplo, dentro do data warehouse, bancos de dados do AWS Glue Data Catalog (awsdatacatalog), bancos de dados de compartilhamento de dados e bancos de dados do Lake Formation.

Permissões obrigatórias

Todos os bancos de dados ficam visíveis para os usuários, mas há uma exceção:

- Nos bancos de dados criados por meio de uma unidade de compartilhamento de dados com permissão para ficarem visíveis, o usuário atual deve receber a permissão USAGE no banco de dados.

Sintaxe

Para mostrar os bancos de dados de um data warehouse do Amazon Redshift:

```
SHOW DATABASES
[ LIKE '<expression>' ]
[ LIMIT row_limit ]
```

Para mostrar os bancos de dados de um catálogo de dados:

```
SHOW DATABASES FROM DATA CATALOG
[ ACCOUNT  '<id1>', '<id2>', ... ]
[ LIKE '<expression>' ]
[ IAM_ROLE default | 'SESSION' | 'arn:aws:iam::<account-id>:role/<role-name>' ]
[ LIMIT row_limit ]
```

Parâmetros

ACCOUNT '<id1>', '<id2>', ...

As contas AWS Glue Data Catalog das quais deseja listar bancos de dados. Omitir esse parâmetro indica que o Amazon Redshift deve mostrar os bancos de dados da conta que detém o cluster.

LIKE '<expression>'

Filtra a lista de bancos de dados aos correspondentes à expressão especificada por você. Esse parâmetro é compatível com padrões que usam os caracteres curinga % (porcentagem) e _ (sublinhado).

IAM_ROLE default | 'SESSION' | 'arn:aws:iam::<account-id>:role/<role-name>'

Se você especificar um perfil do IAM associado ao cluster ao executar o comando SHOW DATABASES, o Amazon Redshift usará as credenciais do perfil ao executar consultas no banco de dados.

Especificar a palavra-chave `default` significa usar o perfil do IAM que está definido como padrão e associado ao cluster.

Use `'SESSION'` se você se conectar ao cluster do Amazon Redshift usando uma identidade federada e acesse as tabelas do banco de dados externo criado usando o comando the section called “CREATE DATABASE”. Para obter um exemplo do uso de uma identidade federada, consulte Usar uma identidade federada para gerenciar o acesso do Amazon Redshift aos recursos locais e a tabelas externas do Amazon Redshift Spectrum, que explica como configurar uma identidade federada.

Use o nome do recurso da Amazon (ARN) de uma função do IAM que seu cluster usa para autenticação e autorização. No mínimo, a função do IAM deve ter permissão para executar uma operação LIST no bucket do Amazon S3 a ser acessado e uma operação GET nos objetos do Amazon S3 que constam no bucket. Para saber mais sobre bancos de dados criados por meio do AWS Glue Data Catalog para unidades de compartilhamento de dados e usando IAM_ROLE, consulte Trabalhar com unidades de compartilhamento de dados gerenciadas pelo Lake Formation como consumidor.

O exemplo a seguir mostra a sintaxe da string do parâmetro IAM_ROLE para um único ARN.

```
IAM_ROLE 'arn:aws:iam::<aws-account-id>:role/<role-name>'
```

Você pode encadear funções para que seu cluster possa assumir outra função do IAM, possivelmente pertencente a outra conta. Você pode encadear até 10 funções. Para obter mais informações, consulte Encadeamento de funções do IAM no Amazon Redshift Spectrum.

Anexe a essa função do IAM uma política de permissões do IAM semelhante à política descrita a seguir.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AccessSecret",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetResourcePolicy",
                "secretsmanager:GetSecretValue",
                "secretsmanager:DescribeSecret",
                "secretsmanager:ListSecretVersionIds"
            ],
            "Resource": "arn:aws:secretsmanager:us-
west-2:123456789012:secret:my-rds-secret-VNenFy"
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword",
                "secretsmanager:ListSecrets"
            ],
            "Resource": "*"
        }
    ]
}
```

Para obter as etapas para criar uma função do IAM a ser usada com consulta federada, consulte Criar um segredo e uma função do IAM para usar consultas federadas.

> **Note**
>
> Não inclua espaços na lista de funções encadeadas.

O seguinte mostra a sintaxe do encadeamento de três funções.

```
IAM_ROLE 'arn:aws:iam::<aws-account-id>:role/<role-1-name>,arn:aws:iam::<aws-
account-id>:role/<role-2-name>,arn:aws:iam::<aws-account-id>:role/<role-3-name>'
```

LIMIT row_limit

Cláusula para LIMITAR o número de linhas geradas. Em que row_limit é o número máximo de linhas a serem geradas. O row_limit pode ser de 0 a 10.000.

Exemplos

O exemplo a seguir exibe todos os bancos de dados do Catálogo de Dados do ID de conta 123456789012.

```
SHOW DATABASES FROM DATA CATALOG ACCOUNT '123456789012'

  catalog_id  | database_name |                        database_arn
 |     type     |                                          target_database
                            | location | parameters
--------------+---------------+-----------------------------------------------------------
+--------------
+-------------------------------------------------------------------------------------------
+----------+------------
 123456789012 |   database1   | arn:aws:glue:us-east-1:123456789012:database/database1
 | Data Catalog |
                            |          |
 123456789012 |   database2   | arn:aws:glue:us-east-1:123456789012:database/database2
 | Data Catalog | arn:aws:redshift:us-
east-1:123456789012:datashare:035c45ea-61ce-86f0-8b75-19ac6102c3b7/database2 |
 |
```

Veja a seguir exemplos que demonstram como exibir todos os bancos de dados do catálogo de dados do ID de conta 123456789012 usando credenciais de um perfil do IAM.

```
SHOW DATABASES FROM DATA CATALOG ACCOUNT '123456789012' IAM_ROLE default;
```

```
SHOW DATABASES FROM DATA CATALOG ACCOUNT '123456789012' IAM_ROLE <iam-role-arn>;
```

O exemplo a seguir exibe todos os bancos de dados no data warehouse do Amazon Redshift conectado.

```
SHOW DATABASES

database_name  | database_owner | database_type        | database_acl | parameters |
 database_isolation_level
---------------+----------------+----------------------+--------------+------------
+--------------------
awsdatacatalog | 1              | auto mounted catalog | NULL         | UNKNOWN    |
 UNKNOWN
dev            | 1              | local                | NULL         | NULL       |
 Snapshot Isolation
```

SHOW FUNCTIONS

Mostra uma lista de funções em um esquema, bem como informações sobre os objetos listados.

Cada linha de saída tem as colunas database_name, schema_name, function_name, number_of_arguments, argument_list, return_type, remarks.

Se SHOW FUNCTIONS produzir mais de 10 mil linhas, o comando vai gerar um erro.

Permissões obrigatórias

Para visualizar uma função em um esquema do Redshift, o usuário atual deve atender a um dos seguintes critérios:

- Ser um superusuário.
- Ser o proprietário da função.
- Ter recebido o privilégio USAGE no esquema principal e o privilégio EXECUTE na função.

Sintaxe

```
SHOW FUNCTIONS FROM SCHEMA
[database_name.]schema_name
[LIKE 'filter_pattern'] [LIMIT row_limit]
```

Parâmetros

database_name

O nome do banco de dados que contém as funções a serem listadas.

schema_name

O nome do esquema que contém as funções a serem listadas.

filter_pattern

Uma expressão de caractere UTF-8 válida com um padrão para estabelecer correspondência com os nomes de função. A opção LIKE executa uma correspondência com distinção entre letras maiúsculas e minúsculas compatível com os seguintes metacaracteres de correspondência de padrões:

Metacaractere	Descrição
%	Corresponde a qualquer sequência de zero ou mais caracteres.
_	Corresponde a qualquer caractere único

Observe que o filter_pattern corresponde apenas ao nome da função.

row_limit

O número máximo de linhas a serem retornadas. O row_limit pode ser de 0 a 10.000.

Exemplos

O seguinte exemplo mostra funções do esquema demo_db.demo_schema:

```
SHOW FUNCTIONS FROM SCHEMA demo_db.demo_schema;
```

```
 database_name | schema_name |    function_name     | number_of_arguments |
                      argument_list                                    |    return_type
  | remarks
--------------+-------------+----------------------+--------------------
+-----------------------------------------------------------------------------
+-------------------+---------
 demo_db       | demo_schema | f2                   |                   6 | integer,
 character varying, numeric, date, timestamp without time zone, boolean | character
 varying |
 demo_db       | demo_schema | f_calculate_discount |                   2 | numeric,
 integer                                                                  | numeric
     |
 demo_db       | demo_schema | f_days_between       |                   2 | date, date
                                                                        | integer
  |
```

O seguinte exemplo mostra funções do esquema demo_schema com nomes que terminam em "discount":

```
SHOW FUNCTIONS FROM SCHEMA demo_schema like '%discount';
 database_name | schema_name |    function_name     | number_of_arguments |
 argument_list   | return_type | remarks
--------------+-------------+----------------------+--------------------
+------------------+-------------+---------
 demo_db       | demo_schema | f_calculate_discount |                   2 | numeric,
 integer | numeric     |
```

SHOW GRANTS

Exibe concessões para um usuário, perfil ou objeto. O objeto pode ser um banco de dados, um esquema, uma tabela, uma função ou um modelo. Ao especificar um objeto, como uma tabela ou função, é necessário qualificá-lo com a notação de duas ou três partes. Por exemplo, o `schema_name.table_name` ou o `database_name.schema_name.table_name`.

Se SHOW GRANTS produzir mais de 10 mil linhas, o comando vai gerar um erro.

Para um banco de dados em um warehouse do Amazon Redshift com permissões federadas, SHOW GRANTS também exibe concessões de permissão DEBUG, incluindo concessões a uma identidade global, concessões a um grupo do IdC e concessões aos administradores de uma conta de consumidor. Para ter mais informações sobre a permissão DEBUG, consulte Observações de uso para conceder a permissão DEBUG.

Permissões obrigatórias

Para executar SHOW GRANTS para um usuário ou perfil de destino, o usuário atual deve atender a um dos seguintes critérios:

- Ser um superusuário.
- Ser o usuário de destino.
- Ser o proprietário do perfil de destino.
- Receber o perfil.

SHOW GRANTS para um objeto de destino exibirá somente concessões visíveis para o usuário atual. Uma concessão ficará visível para o usuário atual se o usuário atual atender a um dos seguintes critérios:

- Ser um superusuário.
- Ser o usuário de destino.
- Receber o título de proprietário do perfil concedido.
- Receber o perfil pretendido pela concessão do objeto.

Sintaxe

Veja a seguir a sintaxe para mostrar concessões em um objeto. Observe que a segunda forma de especificar uma função só é válida para esquemas externos e bancos de dados criados de uma unidade de compartilhamento de dados.

```
SHOW GRANTS ON
{
 DATABASE database_name |
 FUNCTION {database_name.schema_name.function_name | schema_name.function_name }
 ( [ [ argname ] argtype [, ...] ] ) |
 FUNCTION {database_name.schema_name.function_name | schema_name.function_name } |
 SCHEMA {database_name.schema_name | schema_name} |
 { TABLE {database_name.schema_name.table_name | schema_name.table_name} | table_name }
 TEMPLATE {database_name.schema_name.template_name | template_name}
}
[FOR {username | ROLE role_name | PUBLIC}]
[LIMIT row_limit]
```

Veja a seguir a sintaxe para mostrar concessões para um usuário ou perfil.

```
SHOW GRANTS FOR
{username | ROLE role_name}
[FROM DATABASE database_name]
[LIMIT row_limit]
```

Parâmetros

database_name

O nome do banco de dados no qual serão mostradas as concessões.

function_name

O nome do perfil para mostrar as concessões.

template_name

O nome do modelo no qual serão mostradas as concessões.

schema_name

O nome do esquema no qual serão mostradas as concessões.

table_name

O nome da tabela na qual serão mostradas as concessões.

FOR username

Indica a exibição de concessões para um usuário.

FOR ROLE role_name

Indica a exibição de subsídios para um perfil.

FOR PUBLIC

Indica a exibição de concessões para PUBLIC.

row_limit

O número máximo de linhas a serem retornadas. O row_limit pode ser de 0 a 10.000.

Exemplos

O exemplo a seguir mostra todas as concessões em um banco de dados chamado `dev`.

```
SHOW GRANTS on database demo_db;

  database_name | privilege_type | identity_id | identity_name | identity_type |
 admin_option | privilege_scope | grantor_name
---------------+----------------+-------------+---------------+---------------
+--------------+-----------------+-------------
 demo_db       | ALTER          |         112 | alice         | user          | f
      | TABLES          | dbadmin
 demo_db       | TRUNCATE       |         112 | alice         | user          | f
      | TABLES          | dbadmin
 demo_db       | DROP           |         112 | alice         | user          | f
      | TABLES          | dbadmin
 demo_db       | INSERT         |         112 | alice         | user          | f
      | TABLES          | dbadmin
 demo_db       | TEMP           |           0 | public        | public        | f
      | DATABASE        | dbadmin
 demo_db       | SELECT         |         112 | alice         | user          | f
      | TABLES          | dbadmin
 demo_db       | UPDATE         |         112 | alice         | user          | f
      | TABLES          | dbadmin
 demo_db       | DELETE         |         112 | alice         | user          | f
      | TABLES          | dbadmin
 demo_db       | REFERENCES     |         112 | alice         | user          | f
      | TABLES          | dbadmin
```

O exemplo a seguir mostra as concessões da permissão DEBUG em um banco de dados denominado `sales_db` em um warehouse do Amazon Redshift com permissões federadas. Uma concessão para uma identidade global aparece com um `identity_type` de `global_user`, uma concessão a um grupo do IdC aparece com um `identity_type` de `global_group` e uma concessão a administradores da conta aparece com um `identity_type` de `admins_of_account`.

```
SHOW GRANTS ON DATABASE sales_db;

 database_name | privilege_type | identity_id | identity_name                       |
 identity_type     | admin_option | privilege_scope | grantor_name
---------------+----------------+-------------+-------------------------------------
+------------------+-------------+-----------------+-------------
 sales_db      | DEBUG          |         145 | IAM:sales_auditor                   |
 global_user       | f            | DATABASE        | dbadmin
```

```
 sales_db      | DEBUG          |         178 | myidc:audit_team                    |
 global_group      | f            | DATABASE        | dbadmin
 sales_db      | DEBUG          |         203 | 123456789012                        |
 admins_of_account | f            | DATABASE        | dbadmin
```

O comando a seguir mostra todas as concessões em um esquema chamado demo.

```
SHOW GRANTS ON SCHEMA demo_schema;

 schema_name | object_name | object_type | privilege_type | identity_id | identity_name
 | identity_type | admin_option | privilege_scope | database_name | grantor_name
-------------+-------------+-------------+----------------+-------------
+---------------+---------------+--------------+-----------------+---------------
+--------------
 demo_schema | demo_schema | SCHEMA      | ALTER          |         112 | alice
 | user          | f            | SCHEMA          | db1           | dbadmin
 demo_schema | demo_schema | SCHEMA      | DROP           |         112 | alice
 | user          | f            | SCHEMA          | db1           | dbadmin
 demo_schema | demo_schema | SCHEMA      | USAGE          |         112 | alice
 | user          | f            | SCHEMA          | db1           | dbadmin
 demo_schema | demo_schema | SCHEMA      | CREATE         |         112 | alice
 | user          | f            | SCHEMA          | db1           | dbadmin
```

O comando a seguir mostra todas as concessões para um usuário chamado `alice`.

```
SHOW GRANTS FOR alice;

 database_name | schema_name | object_name | object_type | privilege_type | identity_id
 | identity_name | identity_type | privilege_scope | grantor_name
---------------+-------------+-------------+-------------+----------------
+-------------+---------------+---------------+-----------------+-------------
 demo_db       |             |             | DATABASE    | INSERT         |         124
 | alice         | user          | TABLES          | dbadmin
 demo_db       |             |             | DATABASE    | SELECT         |         124
 | alice         | user          | TABLES          | dbadmin
 demo_db       |             |             | DATABASE    | UPDATE         |         124
 | alice         | user          | TABLES          | dbadmin
 demo_db       |             |             | DATABASE    | DELETE         |         124
 | alice         | user          | TABLES          | dbadmin
 demo_db       |             |             | DATABASE    | REFERENCES     |         124
 | alice         | user          | TABLES          | dbadmin
```

```
 demo_db       |             |             | DATABASE    | DROP           |         124
 | alice         | user          | TABLES          | dbadmin
 demo_db       |             |             | DATABASE    | TRUNCATE       |         124
 | alice         | user          | TABLES          | dbadmin
 demo_db       |             |             | DATABASE    | ALTER          |         124
 | alice         | user          | TABLES          | dbadmin
 demo_db       | demo_schema |             | SCHEMA      | USAGE          |         124
 | alice         | user          | SCHEMA          | dbadmin
 demo_db       | demo_schema |             | SCHEMA      | CREATE         |         124
 | alice         | user          | SCHEMA          | dbadmin
 demo_db       | demo_schema |             | SCHEMA      | DROP           |         124
 | alice         | user          | SCHEMA          | dbadmin
 demo_db       | demo_schema |             | SCHEMA      | ALTER          |         124
 | alice         | user          | SCHEMA          | dbadmin
 demo_db       | demo_schema | t1          | TABLE       | INSERT         |         124
 | alice         | user          | TABLE           | dbadmin
 demo_db       | demo_schema | t1          | TABLE       | SELECT         |         124
 | alice         | user          | TABLE           | dbadmin
 demo_db       | demo_schema | t1          | TABLE       | UPDATE         |         124
 | alice         | user          | TABLE           | dbadmin
 demo_db       | demo_schema | t1          | TABLE       | DELETE         |         124
 | alice         | user          | TABLE           | dbadmin
 demo_db       | demo_schema | t1          | TABLE       | RULE           |         124
 | alice         | user          | TABLE           | dbadmin
 demo_db       | demo_schema | t1          | TABLE       | REFERENCES     |         124
 | alice         | user          | TABLE           | dbadmin
 demo_db       | demo_schema | t1          | TABLE       | TRIGGER        |         124
 | alice         | user          | TABLE           | dbadmin
 demo_db       | demo_schema | t1          | TABLE       | DROP           |         124
 | alice         | user          | TABLE           | dbadmin
 demo_db       | demo_schema | t1          | TABLE       | TRUNCATE       |         124
 | alice         | user          | TABLE           | dbadmin
 demo_db       | demo_schema | t1          | TABLE       | ALTER          |         124
 | alice         | user          | TABLE           | dbadmin
```

```
SHOW GRANTS FOR alice FROM DATABASE second_db;
 database_name | schema_name | object_name | object_type | privilege_type | identity_id
 | identity_name | identity_type | privilege_scope | grantor_name
---------------+-------------+-------------+-------------+----------------
+-------------+---------------+---------------+-----------------+--------------
 second_db     | public      | t22         | TABLE       | SELECT         |         101
 | alice         | user          | TABLE           | dbadmin
```

O comando a seguir mostra todas as concessões em uma tabela denominada `t3` para um usuário chamado `alice`. Observe que é possível usar a notação de duas ou três partes para especificar o nome da tabela.

```
SHOW GRANTS ON TABLE demo_db.demo_schema.t3 FOR ALICE;
 schema_name | object_name | object_type | privilege_type | identity_id | identity_name
 | identity_type | admin_option | privilege_scope | database_name | grantor_name
-------------+-------------+-------------+----------------+-------------
+---------------+---------------+--------------+-----------------+---------------
+--------------
 demo_schema | t3          | TABLE       | ALTER          |         130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
 demo_schema | t3          | TABLE       | TRUNCATE       |         130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
 demo_schema | t3          | TABLE       | DROP           |         130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
 demo_schema | t3          | TABLE       | TRIGGER        |         130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
 demo_schema | t3          | TABLE       | SELECT         |         130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
 demo_schema | t3          | TABLE       | INSERT         |         130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
 demo_schema | t3          | TABLE       | UPDATE         |         130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
 demo_schema | t3          | TABLE       | DELETE         |         130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
 demo_schema | t3          | TABLE       | RULE           |         130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
 demo_schema | t3          | TABLE       | REFERENCES     |         130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin


SHOW GRANTS ON TABLE demo_schema.t3 FOR ALICE;
 schema_name | object_name | object_type | privilege_type | identity_id | identity_name
 | identity_type | admin_option | privilege_scope | database_name | grantor_name
-------------+-------------+-------------+----------------+-------------
+---------------+---------------+--------------+-----------------+---------------
+--------------
 demo_schema | t3          | TABLE       | ALTER          |         130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
 demo_schema | t3          | TABLE       | TRUNCATE       |         130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
```

```
 demo_schema | t3          | TABLE       | DROP           |          130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
 demo_schema | t3          | TABLE       | TRIGGER        |          130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
 demo_schema | t3          | TABLE       | SELECT         |          130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
 demo_schema | t3          | TABLE       | INSERT         |          130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
 demo_schema | t3          | TABLE       | UPDATE         |          130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
 demo_schema | t3          | TABLE       | DELETE         |          130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
 demo_schema | t3          | TABLE       | RULE           |          130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
 demo_schema | t3          | TABLE       | REFERENCES     |          130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
```

O exemplo a seguir mostra todas as concessões em uma tabela denominada `t4`. Observe as diferentes formas de especificar o nome da tabela.

```
SHOW GRANTS ON t4;
 schema_name | object_name | object_type | privilege_type | identity_id | identity_name
 | identity_type | admin_option | privilege_scope | database_name | grantor_name
-------------+-------------+-------------+----------------+-------------
+---------------+---------------+--------------+-----------------+---------------
+--------------
 public      | t4          | TABLE       | ALTER          |          130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
 public      | t4          | TABLE       | TRUNCATE       |          130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
 public      | t4          | TABLE       | DROP           |          130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
 public      | t4          | TABLE       | TRIGGER        |          130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
 public      | t4          | TABLE       | SELECT         |          130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
 public      | t4          | TABLE       | INSERT         |          130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
 public      | t4          | TABLE       | UPDATE         |          130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
 public      | t4          | TABLE       | DELETE         |          130 | alice
 | user          | f            | TABLE           | demo_db       | dbadmin
```

```
 public      | t4          | TABLE       | RULE            |         130 | alice
 | user          | f             | TABLE           | demo_db       | dbadmin
 public      | t4          | TABLE       | REFERENCES      |         130 | alice
 | user          | f             | TABLE           | demo_db       | dbadmin

SHOW GRANTS ON TABLE public.t4;
 schema_name | object_name | object_type | privilege_type | identity_id | identity_name
 | identity_type | admin_option | privilege_scope | database_name | grantor_name
-------------+-------------+-------------+----------------+-------------
+---------------+---------------+--------------+-----------------+---------------
+--------------
 public      | t4          | TABLE       | ALTER           |         130 | alice
 | user          | f             | TABLE           | demo_db       | dbadmin
 public      | t4          | TABLE       | TRUNCATE        |         130 | alice
 | user          | f             | TABLE           | demo_db       | dbadmin
 public      | t4          | TABLE       | DROP            |         130 | alice
 | user          | f             | TABLE           | demo_db       | dbadmin
 public      | t4          | TABLE       | TRIGGER         |         130 | alice
 | user          | f             | TABLE           | demo_db       | dbadmin
 public      | t4          | TABLE       | SELECT          |         130 | alice
 | user          | f             | TABLE           | demo_db       | dbadmin
 public      | t4          | TABLE       | INSERT          |         130 | alice
 | user          | f             | TABLE           | demo_db       | dbadmin
 public      | t4          | TABLE       | UPDATE          |         130 | alice
 | user          | f             | TABLE           | demo_db       | dbadmin
 public      | t4          | TABLE       | DELETE          |         130 | alice
 | user          | f             | TABLE           | demo_db       | dbadmin
 public      | t4          | TABLE       | RULE            |         130 | alice
 | user          | f             | TABLE           | demo_db       | dbadmin
 public      | t4          | TABLE       | REFERENCES      |         130 | alice
 | user          | f             | TABLE           | demo_db       | dbadmin
```

SHOW MODEL

Mostra informações úteis sobre um modelo de Machine Learning, incluindo seu status, os parâmetros usados para criá-lo e a função de previsão com seus tipos de argumento de entrada. Você pode usar as informações do SHOW MODEL para recriar o modelo. Se as tabelas base tiverem sido alteradas, executar CREATE MODEL com a mesma instrução SQL resultará em um modelo diferente. As informações retornadas pelo SHOW MODEL são diferentes para o proprietário do modelo e um usuário com o privilégio EXECUTE. SHOW MODEL mostra diferentes saídas quando um modelo é treinado a partir do Amazon Redshift ou quando o modelo é um modelo BYOM.

Sintaxe

```
SHOW MODEL ( ALL | model_name )
```

Parâmetros

ALL

Retorna todos os modelos que o usuário pode usar e seus esquemas.

model_name

O nome do modelo. O nome do modelo em um esquema deve ser exclusivo.

Observações de uso

O comando SHOW MODEL retorna o seguinte:

- O nome do modelo.
- O esquema em que o modelo foi criado.
- O proprietário do modelo.
- A hora de criação do modelo.
- O status do modelo, como READY, TRAINING ou FAILED.
- A mensagem do motivo para um modelo com falha.
- O erro de validação se o modelo tiver terminado o treinamento.
- O custo estimado necessário para derivar o modelo para uma abordagem não-BYOM. Somente o proprietário do modelo pode visualizar essas informações.
- Uma lista de parâmetros especificados pelo usuário e seus valores, especificamente o seguinte:
  - A coluna TARGET especificada.
  - O tipo de modelo, AUTO ou XGBoost.
  - O tipo de problema, como REGRESSION, BINARY_CLASSIFICATION e MULTICLASS_CLASSIFICATION. Este parâmetro é específico para AUTO.
  - O nome do trabalho de treinamento do Amazon SageMaker AI ou do trabalho do Amazon SageMaker AI Autopilot que criou o modelo. Você pode usar esse nome de trabalho para encontrar mais informações sobre o modelo no Amazon SageMaker AI.
  - O objetivo, como MSE, F1, Precisão. Este parâmetro é específico para AUTO.

- O nome da função criada.
- O tipo de inferência, local ou remota.
- Os argumentos de entrada da função de previsão.
- Os tipos de argumento de entrada da função de previsão para modelos que não são BYOM (traga seu próprio modelo).
- O tipo de retorno da função de previsão. Esse parâmetro é específico para BYOM.
- O nome do endpoint do Amazon SageMaker AI para um modelo BYOM com inferência remota.
- A função do IAM. Somente o proprietário do modelo pode ver isso.
- O nome de um bucket do S3. Somente o proprietário do modelo pode ver isso.
- A chave AWS KMS, caso uma tenha sido fornecida. Somente o proprietário do modelo pode ver isso.
- O tempo máximo que o modelo pode ser executado.
- Se o tipo de modelo não for AUTO, o Amazon Redshift também mostrará a lista de hiperparâmetros fornecidos e seus valores.

Você também pode exibir algumas das informações fornecidas pelo SHOW MODEL em outras tabelas de catálogo, como pg_proc. O Amazon Redshift retorna informações sobre a função de previsão registrada na tabela de catálogo pg_proc. Essas informações incluem os nomes dos argumentos de entrada e seus tipos para a função de previsão. O Amazon Redshift retorna as mesmas informações no comando SHOW MODEL.

```
SELECT * FROM pg_proc WHERE proname ILIKE '%<function_name>%';
```

Exemplos

O exemplo a seguir mostra a saída do show model.

```
SHOW MODEL ALL;

Schema Name |  Model Name
------------+---------------
 public     | customer_churn
```

O proprietário do customer_churn pode ver a saída a seguir. Um usuário com apenas o privilégio EXECUTE não pode ver a função do IAM, o bucket do Amazon S3 e o custo estimado do modo.

```
SHOW MODEL customer_churn;

       Key                  |           Value
----------------------------+-----------------------------------
 Model Name                 | customer_churn
 Schema Name                | public
 Owner                      | 'owner'
 Creation Time              | Sat, 15.01.2000 14:45:20
 Model State                | READY
 validation:F1              | 0.855
 Estimated Cost             | 5.7
                            |
 TRAINING DATA:             |
 Table                      | customer_data
 Target Column              | CHURN
                            |
 PARAMETERS:                |
 Model Type                 | auto
 Problem Type               | binary_classification
 Objective                  | f1
 Function Name              | predict_churn
 Function Parameters        | age zip average_daily_spend average_daily_cases
 Function Parameter Types   | int int float float
 IAM Role                   | 'iam_role'
 KMS Key                    | 'kms_key'
 Max Runtime                | 36000
```

SHOW DATASHARES

Exibe os compartilhamentos de entrada e saída em um cluster da mesma conta ou entre contas. Se você não especificar um nome de datashare, o Amazon Redshift exibirá todos os datashares em todos os bancos de dados no cluster. Os usuários que têm os privilégios ALTER e SHARE podem ver os compartilhamentos para os quais eles têm privilégios.

Sintaxe

```
SHOW DATASHARES [ LIKE 'namepattern' ]
```

Parâmetros

LIKE

Uma cláusula opcional que compara o padrão de nome especificado com a descrição do datashare. Quando essa cláusula é usada, o Amazon Redshift exibe apenas os datashares com nomes que correspondem ao padrão de nome especificado.

namepattern

O nome do datashare solicitado ou parte do nome a ser correspondido usando caracteres curinga.

Exemplos

O exemplo a seguir exibe os compartilhamentos de entrada e saída em um cluster.

```
SHOW DATASHARES;
SHOW DATASHARES LIKE 'sales%';

share_name   | share_owner | source_database | consumer_database | share_type |
 createdate          | is_publicaccessible | share_acl | producer_account |
 producer_namespace
-------------+-------------+-----------------+-------------------+------------
+---------------------+---------------------+-----------+------------------
+--------------------------------------
'salesshare' | 100         | dev             |                   | outbound
 | 2020-12-09 01:22:54.| False               |           |   123456789012   |
 13b8833d-17c6-4f16-8fe4-1a018f5ed00d
```

SHOW PARAMETERS

Mostra uma lista de parâmetros para uma função/procedimento, junto com algumas informações sobre os parâmetros.

Cada linha de saída tem colunas database_name, schema_name, procedure name ou function_name, parameter_name, ordinal_position, parameter_type (IN/OUT), data_type, character_maximum_length, numeric_precision, numeric_scale e remarks.

Permissões obrigatórias

Para visualizar uma função/procedimento em um esquema do Redshift, o usuário atual deve atender a um dos seguintes critérios:

- Ser um superusuário.
- Ser o proprietário da função.
- Ter recebido o privilégio USAGE no esquema principal e o privilégio EXECUTE na função.

Sintaxe

```
SHOW PARAMETERS OF {FUNCTION| PROCEDURE}
[database_name.]schema_name.function_name(argtype [, ...] )
[LIKE 'filter_pattern'];
```

Parâmetros

database_name

O nome do banco de dados que contém a função a ser listada.

schema_name

O nome do esquema que contém a função a ser listada.

filter_pattern

Uma expressão de caractere UTF-8 válida com o padrão para estabelecer correspondência com os nomes de parâmetro. A opção LIKE executa uma correspondência com distinção entre letras maiúsculas e minúsculas compatível com os seguintes metacaracteres de correspondência de padrões:

Metacaractere	Descrição
%	Corresponde a qualquer sequência de zero ou mais caracteres.
_	Corresponde a qualquer caractere único

Exemplos

O seguinte exemplo mostra os parâmetros do procedimento demo_db.demo_schema.f1:

```
SHOW PARAMETERS OF PROCEDURE demo_db.demo_schema.f1(VARCHAR, DECIMAL, DECIMAL,
 DECIMAL);
 database_name | schema_name | procedure_name |  parameter_name  | ordinal_position
 | parameter_type |          data_type          | character_maximum_length |
 numeric_precision | numeric_scale
---------------+-------------+----------------+------------------+-----------------
+----------------+-----------------------------+--------------------------
+-------------------+--------------
 demo_db       | demo_schema | f1             | operation        |                1 |
 IN             | character varying           |                       10 |
      |
 demo_db       | demo_schema | f1             | value1           |                2 |
 IN             | numeric                     |                          |
   18 |            0
 demo_db       | demo_schema | f1             | value2           |                3 |
 IN             | numeric                     |                          |
   18 |            0
 demo_db       | demo_schema | f1             | result           |                4 |
 INOUT          | numeric                     |                          |
   18 |            0
 demo_db       | demo_schema | f1             | operation_status |                5 |
 OUT            | character varying           |                       50 |
      |
 demo_db       | demo_schema | f1             | calculation_time |                6 |
 OUT            | timestamp without time zone |                          |
      |
 demo_db       | demo_schema | f1             | is_successful    |                7 |
 OUT            | boolean                     |                          |
      |
```

O seguinte exemplo mostra os parâmetros do procedimento demo_schema.f1 com nomes começando com 'val':

```
SHOW PARAMETERS OF PROCEDURE demo_schema.f1(VARCHAR, DECIMAL, DECIMAL, DECIMAL) like
 'val%';
 database_name | schema_name | procedure_name | parameter_name | ordinal_position
 | parameter_type | data_type | character_maximum_length | numeric_precision |
 numeric_scale
```

```
--------------+------------+---------------+---------------+-----------------
+---------------+----------+-------------------------+------------------
+---------------
 demo_db       | demo_schema | f1             | value1         |               2 | IN
           | numeric   |                          |               18 |             0
 demo_db       | demo_schema | f1             | value2         |               3 | IN
           | numeric   |                          |               18 |             0
```

O seguinte exemplo mostra os parâmetros da função demo_schema.f2:

```
SHOW PARAMETERS OF FUNCTION demo_schema.f2(INT, VARCHAR, DECIMAL, DATE, TIMESTAMP,
 BOOLEAN);
 database_name | schema_name | function_name | parameter_name  | ordinal_position
 | parameter_type |          data_type          | character_maximum_length |
 numeric_precision | numeric_scale
--------------+------------+--------------+----------------+-----------------
+---------------+-----------------------------+-------------------------
+------------------+--------------
 demo_db       | demo_schema | f2           |                |               0 |
 RETURN         | character varying           |                       -1 |
      |
 demo_db       | demo_schema | f2           | int_param      |               1 | IN
            | integer                    |                          |
 32 |             0
 demo_db       | demo_schema | f2           | varchar_param  |               2 | IN
           | character varying          |                       -1 |
  |
 demo_db       | demo_schema | f2           | decimal_param  |               3 | IN
           | numeric                    |                          |
  |
 demo_db       | demo_schema | f2           | date_param     |               4 | IN
           | date                       |                          |
  |
 demo_db       | demo_schema | f2           | timestamp_param |               5 | IN
           | timestamp without time zone |                          |
  |
 demo_db       | demo_schema | f2           | boolean_param  |               6 | IN
           | boolean                    |                          |
  |
```

SHOW POLICIES

Exibe as políticas de segurança por linha (RLS) e de mascaramento dinâmico de dados (DDM) definidas em um banco de dados, bem como as políticas de RLS e DDM aplicadas a relações específicas. Somente um superusuário ou usuário com o perfil `sys:secadmin` no banco de dados pode visualizar os resultados dessas políticas.

Sintaxe

```
SHOW { RLS | MASKING } POLICIES
[
    ON { database_name.schema_name.relation_name
       | schema_name.relation_name
       }
    [ FOR { user_name | ROLE role_name | PUBLIC } ]
  |
    FROM DATABASE database_name
]
[ LIMIT row_limit ];
```

Parâmetros

database_name

O nome do banco de dados do qual serão mostradas as políticas.

schema_name

Nome do esquema da relação na qual serão mostradas as políticas anexadas.

relation_name

O nome da relação na qual serão mostradas as políticas anexadas.

user_name

O nome do usuário ao qual a política está anexada na relação.

role_name

O nome do perfil ao qual a política está anexada na relação.

row_limit

O número máximo de linhas a serem retornadas. O row_limit pode ser de 0 a 10.000.

> **Note**
>
> Mostrar políticas de um banco de dados diferente do banco de dados conectado é aceito no Catálogo de Permissões Federadas do Amazon Redshift. O comando SHOW POLICIES aceita consultas entre bancos de dados para todos os bancos de dados em warehouses com permissões federadas do Amazon Redshift

Exemplos

O comando a seguir mostra as políticas de RLS do banco de dados conectado.

```
SHOW RLS POLICIES;

  policy_name   | policy_alias |                          policy_atts
                |
 policy_qual                                                                  |
 policy_enabled | policy_modified_by |    policy_modified_time
----------------+--------------
+-------------------------------------------------------------
+-------------------------------------------------------------------------------------------------------
+---------------+--------------------+---------------------------
 policy_america | rls_table    | [{"colname":"region","type":"character varying(10)"}]
           | (("rls_table"."region" = CAST('USA' AS TEXT)) OR ("rls_table"."region" =
 CAST('CANADA' AS TEXT)) OR ("rls_table"."region" = CAST('Mexico' AS TEXT))) | t
       | admin              | 2025-11-07 14:57:27
```

O comando a seguir mostra as políticas de mascaramento do banco de dados “sales_db.finance-catalog”;

```
SHOW MASKING POLICIES FROM DATABASE "sales_db@finance-catalog";

  policy_name  |                          input_columns                          |
                                    policy_expression
                         | policy_modified_by |    policy_modified_time
--------------+-----------------------------------------------------------------
+-------------------------------------------------------------------------------------------------------
+--------------------+---------------------------
 hash_credit   | [{"colname":"credit_card","type":"character varying(256)"}]      |
 [{"expr":"SHA2((\"masked_table\".\"credit_card\" + CAST('testSalt' AS TEXT)), CAST(256
 AS INT4))","type":"text"}]   | admin              | 2025-11-07 16:05:54
```

```
 hash_username | [{"colname":"username","type":"character varying(256)"}]
 | [{"expr":"SHA2((\"masked_table\".\"username\" + CAST('otherTestSalt' AS TEXT)),
 CAST(256 AS INT4))","type":"text"}] | admin              | 2025-11-07 16:07:08
(2 rows)
```

O comando a seguir mostra as políticas de RLS anexadas à relação sales_table;

```
SHOW RLS POLICIES ON sales_schema.sales_table;

  policy_name   | schema_name  | relation_name | relation_kind | grantor  |
 grantee          | grantee_kind | is_policy_on | is_rls_on | rls_conjunction_type
---------------+-------------+--------------+--------------+----------
+--------------------------+-------------+-------------+-----------
+----------------------
 policy_global  | sales_schema | sales_table   | table         | admin    |
 sales_analyst_role_global | role         | t            | t         | and
 policy_america | sales_schema | sales_table   | table         | admin    |
 sales_analyst_usa         | user         | t            | t         | and
```

O comando a seguir mostra as políticas de mascaramento anexadas à relação transaction_table do banco de dados “sales_db.finance-catalog”.

```
SHOW MASKING POLICIES ON "sales_db@finance-catalog".sales_schema.transaction_table
 LIMIT 1;

  policy_name  | schema_name  |   relation_name   | relation_type | grantor  |
 grantee          | grantee_type | priority |   input_columns   |   output_columns
--------------+-------------+------------------+--------------+----------
+-------------------------+-------------+----------+------------------
+------------------
 hash_username | sales_schema | transaction_table | table         | admin    |
 transaction_analyst_role | role         |      100 | ["user_name"]     | ["user_name"]
```

O comando a seguir mostra as políticas de RLS anexadas à relação sales_table do banco de dados “sales_db.finance-catalog” para o usuário “IAMR:sales_analyst_usa”.

```
SHOW RLS POLICIES ON "sales_db@finance-catalog".sales_schema.sales_table FOR
 "IAMR:sales_analyst_usa";

  policy_name   | schema_name  | relation_name | relation_kind | grantor  |
 grantee           | grantee_kind | is_policy_on | is_rls_on | rls_conjunction_type
```

```
---------------+-------------+--------------+--------------+----------
+----------------------+-------------+-------------+----------
+---------------------
 policy_america | sales_schema | sales_table   | table         | admin    |
 IAMR:sales_analyst_usa | user         | t            | t         | and
```

O comando a seguir mostra as políticas de RLS anexadas à relação transaction_table do banco de dados “sales_db.finance-catalog” para o perfil transaction_analyst_role.

```
SHOW MASKING POLICIES ON sales_schema.transaction_table FOR ROLE
 transaction_analyst_role;

  policy_name  | schema_name  |   relation_name   | relation_type | grantor  |
 grantee          | grantee_type | priority | input_columns | output_columns
--------------+-------------+------------------+--------------+----------
+-------------------------+-------------+----------+--------------+---------------
 hash_username | sales_schema | transaction_table | table         | admin    |
 transaction_analyst_role | role         |      100 | ["user_name"] | ["user_name"]
```

SHOW PROCEDURE

Mostra a definição de um procedimento armazenado específico, incluindo sua assinatura. Use a saída de um SHOW PROCEDURE para recriar o procedimento armazenado.

Sintaxe

```
SHOW PROCEDURE sp_name [( [ [ argname ] [ argmode ] argtype [, ...] ] )]
```

Parâmetros

sp_name

O nome do procedimento a ser exibido.

[argname] [argmode] argtype

Tipos de argumento de entrada para identificar o procedimento armazenado. Opcionalmente, você pode incluir todos os tipos de dados de argumentos, incluindo argumentos OUT. Essa parte é opcional se o nome do procedimento armazenado for único (isto é, não sobrecarregado).

Exemplos

O exemplo a seguir mostra a definição do procedimento `test_spl2`.

```
show procedure test_sp2(int, varchar);
                                        Stored Procedure Definition
-------------------------------------------------------------------------------------------------
CREATE OR REPLACE PROCEDURE public.test_sp2(f1 integer, INOUT f2 character varying, OUT
 character varying)
LANGUAGE plpgsql
AS $_$
DECLARE
out_var alias for $3;
loop_var int;
BEGIN
IF f1 is null OR f2 is null THEN
RAISE EXCEPTION 'input cannot be null';
END IF;
CREATE TEMP TABLE etl(a int, b varchar);
FOR loop_var IN 1..f1 LOOP
insert into etl values (loop_var, f2);
f2 := f2 || '+' || f2;
END LOOP;
SELECT INTO out_var count(*) from etl;
END;
$_$

(1 row)
```

SHOW PROCEDURES

Mostra uma lista de procedimentos em um esquema, bem como informações sobre os objetos listados.

Cada linha de saída tem as colunas `database_name`, `schema_name`, `procedure_name`, `number_of_arguments`, `argument_list`, `return_type`, remarks.

Se SHOW PROCEDURES produzir mais de 10 mil linhas, o comando vai gerar um erro.

Permissões obrigatórias

Para visualizar um procedimento em um esquema do Redshift, o usuário atual deve atender a um dos seguintes critérios:

- Ser um superusuário.
- Ser o proprietário do procedimento.
- Ter recebido o privilégio USAGE no esquema principal e o privilégio EXECUTE no procedimento.

Sintaxe

```
SHOW PROCEDURES FROM SCHEMA
[database_name.]schema_name
[LIKE 'filter_pattern'] [LIMIT row_limit]
```

Parâmetros

database_name

O nome do banco de dados que contém os procedimentos a serem listados.

schema_name

O nome do esquema que contém os procedimentos a serem listados.

filter_pattern

Uma expressão de caractere UTF-8 válida com o padrão para estabelecer correspondência com os nomes de procedimento. A opção LIKE executa uma correspondência com distinção entre letras maiúsculas e minúsculas compatível com os seguintes metacaracteres de correspondência de padrões:

Metacaractere	Descrição
%	Corresponde a qualquer sequência de zero ou mais caracteres.
_	Corresponde a qualquer caractere único

Observe que o filter_pattern corresponde apenas ao nome do procedimento.

row_limit

O número máximo de linhas a serem retornadas. O row_limit pode ser de 0 a 10.000.

Exemplos

O seguinte exemplo mostra procedimentos do esquema demo_db.demo_schema:

```
SHOW PROCEDURES FROM SCHEMA demo_db.demo_schema;
 database_name | schema_name |  procedure_name   | number_of_arguments |
 argument_list                 |                          return_type
           | remarks
---------------+-------------+-------------------+---------------------
+-----------------------------------------
+-------------------------------------------------------------------+---------
 demo_db       | demo_schema | f1                |                   4 | character
 varying, numeric, numeric, numeric | numeric, character varying, timestamp without
 time zone, boolean |
 demo_db       | demo_schema | sp_get_result_set |                   2 | integer,
 refcursor                          | refcursor
                |
 demo_db       | demo_schema | sp_process_data   |                   2 | numeric,
 numeric                            | numeric, character varying
                |
```

O seguinte exemplo mostra procedimentos do esquema demo_schema com nomes que terminam com “data”:

```
SHOW PROCEDURES FROM SCHEMA demo_schema like '%data';
 database_name | schema_name | procedure_name  | number_of_arguments |  argument_list
 |        return_type         | remarks
---------------+-------------+-----------------+---------------------
+------------------+-----------------------------+---------
 demo_db       | demo_schema | sp_process_data |                   2 | numeric, numeric
 | numeric, character varying |
```

SHOW SCHEMAS

Mostra uma lista de esquemas em um banco de dados, bem como alguns atributos do esquema.

Cada linha de saída é composta dos seguintes elementos: nome do banco de dados, nome do esquema, proprietário do esquema, tipo de esquema, ACL do esquema, banco de dados de origem e opção de esquema. Para obter mais informações sobre esses atributos, consulte SVV_ALL_SCHEMAS.

Se mais de 10.000 esquemas puderem resultar do comando SHOW SCHEMAS, será retornado um erro.

Permissões obrigatórias

Para visualizar um esquema em uma tabela do Amazon Redshift, o usuário atual deve atender a um dos seguintes critérios:

- Ser um superusuário.
- Ser o proprietário do esquema.
- Ter recebido o privilégio USAGE no esquema.

Sintaxe

```
SHOW SCHEMAS FROM DATABASE database_name [LIKE 'filter_pattern'] [LIMIT row_limit ]
```

Parâmetros

database_name

O nome do banco de dados que contém as tabelas a serem listadas.

Para mostrar tabelas em um AWS Glue Data Catalog, especifique (`awsdatacatalog`) como o nome do banco de dados e verifique se a configuração do sistema `data_catalog_auto_mount` está definida como `true`. Para obter mais informações, consulte ALTER SYSTEM.

filter_pattern

Uma expressão de caractere UTF-8 válida com o padrão para estabelecer correspondência com os nomes do esquema. A opção LIKE executa uma correspondência com distinção entre letras maiúsculas e minúsculas compatível com os seguintes metacaracteres de correspondência de padrões:

Metacaractere	Descrição
%	Corresponde a qualquer sequência de zero ou mais caracteres.
_	Corresponde a qualquer caractere único.

Se filter_pattern não contiver metacaracteres, o padrão representará somente a própria string. Nesse caso, LIKE age da mesma forma que o operador de igualdade.

row_limit

O número máximo de linhas a serem retornadas. O row_limit pode ser de 0 a 10.000.

Exemplos

O exemplo a seguir mostra os esquemas do banco de dados do Amazon Redshift chamado `dev`.

```
SHOW SCHEMAS FROM DATABASE dev;

 database_name |     schema_name      | schema_owner | schema_type |         schema_acl
          | source_database | schema_option
---------------+----------------------+--------------+-------------
+----------------------------+-----------------+---------------
 dev           | pg_automv            |            1 | local       |
          |                 |
 dev           | pg_catalog           |            1 | local       | jpuser=UC/
jpuser~=U/jpuser  |                 |
 dev           | public               |            1 | local       | jpuser=UC/
jpuser~=UC/jpuser |                 |
 dev           | information_schema   |            1 | local       | jpuser=UC/
jpuser~=U/jpuser  |                 |
 dev           | schemad79cd6d93bf043 |            1 | local       |
          |                 |
```

O exemplo a seguir mostra os esquemas no banco de dados do AWS Glue Data Catalog chamado `awsdatacatalog`. O número máximo de linhas de saída é 5.

```
SHOW SCHEMAS FROM DATABASE awsdatacatalog LIMIT 5;

 database_name  |     schema_name      | schema_owner | schema_type | schema_acl |
 source_database | schema_option
----------------+----------------------+--------------+-------------+------------
+-----------------+---------------
 awsdatacatalog | 000_too_many_glue_db |              | EXTERNAL    |            |
            |
 awsdatacatalog | 123_default          |              | EXTERNAL    |            |
            |
 awsdatacatalog | adhoc                |              | EXTERNAL    |            |
            |
```

```
 awsdatacatalog | all_shapes_10mb      |             | EXTERNAL    |           |
           |
 awsdatacatalog | all_shapes_1g        |             | EXTERNAL    |           |
           |
```

SHOW TABLE

Exibe a definição de uma tabela, incluindo atributos e restrições de tabela, atributos e agrupamento de colunas, bem como restrições de coluna. Use a saída da instrução SHOW TABLE para recriar a tabela.

Para obter mais informações sobre criação de tabelas, consulte CRIAR TABELA.

Sintaxe

```
SHOW TABLE [schema_name.]table_name
```

Parâmetros

schema_name

(Opcional) O nome do esquema relacionado.

table_name

O nome da tabela a ser exibida.

Exemplos

Segue-se um exemplo do comando e saída SHOW TABLE para a tabela `sales`.

```
show table sales;
```

```
CREATE TABLE public.sales (
salesid integer NOT NULL ENCODE az64,
listid integer NOT NULL ENCODE az64 distkey,
sellerid integer NOT NULL ENCODE az64,
buyerid integer NOT NULL ENCODE az64,
eventid integer NOT NULL ENCODE az64,
dateid smallint NOT NULL,
```

```
qtysold smallint NOT NULL ENCODE az64,
pricepaid numeric(8,2) ENCODE az64,
commission numeric(8,2) ENCODE az64,
saletime timestamp without time zone ENCODE az64
)
DISTSTYLE KEY SORTKEY ( dateid );
```

Segue-se um exemplo da saída SHOW TABLE para a tabela `category` no esquema `public`. O agrupamento do banco de dados é CASE_SENSITIVE.

```
show table public.category;
```

```
CREATE TABLE public.category (
catid smallint NOT NULL distkey,
catgroup character varying(10) ENCODE lzo COLLATE case_sensitive,
catname character varying(10) ENCODE lzo COLLATE case_sensitive,
catdesc character varying(50) ENCODE lzo COLLATE case_sensitive
)
DISTSTYLE KEY SORTKEY ( catid );
```

O exemplo a seguir cria a tabela `foo` com uma chave primária.

```
create table foo(a int PRIMARY KEY, b int);
```

Os resultados SHOW TABLE exibem a instrução create com todas as propriedades da tabela `foo`.

```
show table foo;
```

```
CREATE TABLE public.foo (
a integer NOT NULL ENCODE az64,
b integer ENCODE az64, PRIMARY KEY (a)
)
DISTSTYLE AUTO;
```

Esse exemplo cria uma tabela em que a coluna a herda a ordenação CASE_SENSITIVE padrão do banco de dados, enquanto b e c são definidas explicitamente com a ordenação CASE_INSENSITIVE.

```
CREATE TABLE public.foo (
```

```
a CHAR,
b VARCHAR(10) COLLATE CASE_INSENSITIVE,
c SUPER COLLATE CASE_INSENSITIVE
);
```

Os resultados SHOW TABLE exibem a instrução create com todas as propriedades da tabela `foo`.

```
show table public.foo;
```

```
CREATE TABLE public.foo (
a character(1) ENCODE lzo COLLATE case_sensitive,
b character varying(10) ENCODE lzo COLLATE case_insensitive,
c super COLLATE case_insensitive
)
DISTSTYLE AUTO;
```

SHOW TABLES

Mostra uma lista de tabelas em um esquema, bem como alguns atributos da tabela.

Cada linha de saída é composta dos seguintes elementos: nome do banco de dados, nome do esquema, nome da tabela, tipo de tabela, ACL da tabela, comentários, proprietário da tabela, hora da última alteração, hora da última modificação, dist_style e subtipo da tabela. Para obter mais informações sobre esses atributos, consulte SVV_ALL_TABLES.

Os carimbos de data/hora de modificação e alteração podem ficar atrasados em relação às atualizações da tabela em um intervalo fixo de aproximadamente 5 minutos.

Se mais de 10.000 tabelas resultarem do comando SHOW TABLES, será retornado um erro.

Permissões obrigatórias

Para visualizar uma tabela em um esquema do Amazon Redshift, o usuário atual deve atender a um dos seguintes critérios:

- Ser um superusuário.
- Ser o proprietário da tabela.
- Ter recebido o privilégio USAGE no esquema principal e o privilégio SELECT na tabela ou ter recebido o privilégio SELECT em qualquer coluna da tabela.

Sintaxe

```
SHOW TABLES FROM SCHEMA database_name.schema_name [LIKE 'filter_pattern']
 [LIMIT row_limit ]
```

Parâmetros

database_name

O nome do banco de dados que contém as tabelas a serem listadas.

Para mostrar tabelas em um AWS Glue Data Catalog, especifique (`awsdatacatalog`) como o nome do banco de dados e verifique se a configuração do sistema `data_catalog_auto_mount` está definida como `true`. Para obter mais informações, consulte ALTER SYSTEM.

schema_name

O nome do esquema que contém as tabelas a serem listadas.

Para mostrar tabelas do AWS Glue Data Catalog, forneça o nome do banco de dados do AWS Glue como nome do esquema.

filter_pattern

Uma expressão de caractere UTF-8 válida com o padrão para estabelecer correspondência com os nomes da tabela. A opção LIKE executa uma correspondência com distinção entre letras maiúsculas e minúsculas compatível com os seguintes metacaracteres de correspondência de padrões:

Metacaractere	Descrição
%	Corresponde a qualquer sequência de zero ou mais caracteres.
_	Corresponde a qualquer caractere único.

Se filter_pattern não contiver metacaracteres, o padrão representará somente a própria string. Nesse caso, LIKE age da mesma forma que o operador de igualdade.

row_limit

O número máximo de linhas a serem retornadas. O row_limit pode ser de 0 a 10.000.

Exemplos

```
SHOW TABLES FROM SCHEMA s1;

 database_name | schema_name |    table_name     | table_type |              table_acl
             | remarks | owner |     last_altered_time      |     last_modified_time
  | dist_style |   table_subtype
---------------+-------------+-------------------+------------
+-------------------------------------+---------+-------+---------------------------
+---------------------------+------------+-------------------
 dev           | s1          | late_binding_view | VIEW       | alice=arwdRxtDPA/
alice~bob=d/alice  |         | alice |                              |
        |            | LATE BINDING VIEW
 dev           | s1          | manual_mv         | VIEW       | alice=arwdRxtDPA/
alice~bob=P/alice  |         | alice |                              |
        |            | MATERIALIZED VIEW
 dev           | s1          | regular_view      | VIEW       | alice=arwdRxtDPA/
alice~bob=r/alice  |         | alice |                              |
        |            | REGULAR VIEW
 dev           | s1          | test_table        | TABLE      | alice=arwdRxtDPA/
alice~bob=rw/alice |         | alice | 2025-11-18 15:52:00.010452 | 2025-11-18
 15:44:34.856073 | AUTO (ALL) | REGULAR TABLE
```

```
SHOW TABLES FROM SCHEMA dev.s1 LIKE '%view' LIMIT 1;

 database_name | schema_name |    table_name     | table_type |              table_acl
              | remarks | owner | last_altered_time | last_modified_time | dist_style |
   table_subtype
---------------+-------------+-------------------+------------
+-------------------------------------+---------+-------+-------------------
+--------------------+------------+-------------------
 dev           | s1          | late_binding_view | VIEW       | {alice=arwdRxtDPA/
alice,bob=d/alice} |         | alice |                   |                    |
    | LATE BINDING VIEW
```

As visualizações materializadas do Iceberg aparecem na saída SHOW TABLES com `table_type = 'EXTERNAL TABLE'` e `table_subtype = 'ICEBERG MATERIALIZED VIEW'`. Isso as distingue das visualizações materializadas padrão (RMS), que têm `table_type = 'VIEW'` e `table_subtype = 'MATERIALIZED VIEW'`.

O exemplo a seguir lista tabelas em um banco de dados AWS Glue, nas quais as visualizações materializadas do Iceberg podem ser identificadas por `table_subtype`:

```
SHOW TABLES FROM SCHEMA awsdatacatalog.mydb LIKE '%';
```

SHOW TEMPLATE

Exibe a definição completa de um modelo, como o nome totalmente qualificado (banco de dados, esquema e nome do modelo) e todos os parâmetros. A saída é uma instrução CREATE TEMPLATE válida que pode ser usada para recriar o modelo ou criar um modelo similar com modificações.

Para ter mais informações sobre como criar um modelo, consulte CREATE TEMPLATE.

Permissões obrigatórias

Para visualizar a definição de um modelo, é necessário ter uma das seguintes opções:

- Privilégios de superusuário.
- Privilégio USAGE no modelo e privilégio USAGE no esquema que contém o modelo.

Sintaxe

```
SHOW TEMPLATE [database_name.][schema_name.]template_name;
```

Parâmetros

database_name

(Opcional) O nome do banco de dados no qual o modelo é criado. Se não especificado, será usado o banco de dados atual.

schema_name

(Opcional) O nome do esquema no qual o modelo é criado. Se não for especificado, o modelo será pesquisado no caminho de pesquisa atual.

template_name

O nome do modelo.

Exemplos

Veja abaixo um exemplo da saída SHOW TEMPLATE para o modelo`test_template`:

```
CREATE TEMPLATE test_template FOR COPY AS NOLOAD DELIMITER ',' ENCODING UTF16
 ENCRYPTED;
```

```
SHOW TEMPLATE test_template;

CREATE OR REPLACE TEMPLATE dev.public.test_template FOR COPY AS ENCRYPTED NOLOAD
 ENCODING UTF16 DELIMITER ',';
```

O exemplo a seguir cria um modelo `demo_template` no esquema `demo_schema`.

```
CREATE OR REPLACE TEMPLATE demo_schema.demo_template FOR COPY AS
ACCEPTANYDATE ACCEPTINVCHARS DATEFORMAT 'DD-MM-YYYY' EXPLICIT_IDS ROUNDEC
TIMEFORMAT  AS 'DD.MM.YYYY HH:MI:SS' TRUNCATECOLUMNS NULL  AS 'null_string';
```

```
SHOW TEMPLATE demo_schema.demo_template;

CREATE OR REPLACE TEMPLATE dev.demo_schema.demo_template FOR COPY AS TRUNCATECOLUMNS
 NULL 'null_string' EXPLICIT_IDS TIMEFORMAT 'DD.MM.YYYY HH:MI:SS' ACCEPTANYDATE ROUNDEC
 ACCEPTINVCHARS DATEFORMAT 'DD-MM-YYYY';
```

SHOW TEMPLATES

Mostra uma lista de modelos em um esquema, bem como os respectivos atributos.

Cada linha de saída é composta dos seguintes elementos: nome do modelo, ID do modelo, tipo de modelo, proprietário do modelo, nome do banco de dados, hora de criação, hora da última modificação e nome de quem o modificou pela última vez.

Para ver detalhes completos do modelo, inclusive parâmetros do modelo, consulte SYS_REDSHIFT_TEMPLATE.

Permissões obrigatórias

Para visualizar modelos em um esquema do Amazon Redshift, é necessário ter uma das seguintes opções:

- Privilégios de superusuário.
- Privilégio USAGE no esquema que contém os modelos.

Sintaxe

```
SHOW TEMPLATES FROM SCHEMA [database_name.]schema_name [LIKE 'filter_pattern']
 [LIMIT row_limit ];
```

Parâmetros

database_name

(Opcional) O nome do banco de dados que contém os modelos a serem listados. Se não for informado, será usado o banco de dados atual.

schema_name

O nome do esquema que contém os modelos a serem listados.

filter_pattern

(Opcional) Uma expressão de caractere UTF-8 válida com o padrão para estabelecer correspondência com os nomes de modelo. A opção LIKE executa uma correspondência com distinção entre letras maiúsculas e minúsculas compatível com os seguintes metacaracteres de correspondência de padrões:

Metacaractere	Descrição
%	Corresponde a qualquer sequência de zero ou mais caracteres.
_	Corresponde a qualquer caractere único.

Se filter_pattern não contiver metacaracteres, o padrão representará somente a própria string. Nesse caso, LIKE age da mesma forma que o operador de igualdade.

row_limit

O número máximo de linhas a serem retornadas. O intervalo válido é 0 até o limite do modelo no cluster (o padrão é 1.000).

Exemplos

```
SHOW TEMPLATES FROM SCHEMA s1;
```

```
 template_name          | template_id | template_type | template_owner | database_name
 | schema_name |        create_time         |     last_modified_time     |
 last_modified_by
------------------------+-------------+---------------+----------------+--------------
+-------------+----------------------------+----------------------------
+------------------
 template_maxerror      |      107685 | COPY          | alice          | dev
 | s1          | 2025-12-16 19:31:10.514076 | 2025-12-16 19:31:10.514076 |
 100
 json_template          |      107687 | COPY          | alice          | dev
 | s1          | 2025-12-16 19:31:33.229566 | 2025-12-16 19:31:33.229567 |
 100
 noload_template        |      107686 | COPY          | alice          | dev
 | s1          | 2025-12-16 19:31:17.370547 | 2025-12-16 19:31:17.370547 |
 100
 csv_delimiter_template |      107688 | COPY          | alice          | dev
 | s1          | 2025-12-16 19:31:42.354044 | 2025-12-16 19:31:42.354045 |
 100
```

```
SHOW TEMPLATES FROM SCHEMA dev.s1 LIKE '%template' LIMIT 1;

 template_name  | template_id | template_type | template_owner | database_name
 | schema_name |        create_time         |     last_modified_time     |
 last_modified_by
-----------------+-------------+---------------+----------------+--------------
+-------------+----------------------------+----------------------------
+------------------
 noload_template |      107686 | COPY          | alice          | dev           | s1
       | 2025-12-16 19:31:17.370547 | 2025-12-16 19:31:17.370547 |             100
```

SHOW VIEW

Exibe a definição de uma visualização, inclusive para visões materializadas e visualizações de vinculação tardia. Use a saída da instrução SHOW VIEW para recriar a visualização.

Sintaxe

```
SHOW VIEW [schema_name.]view_name
```

Parâmetros

schema_name

(Opcional) O nome do esquema relacionado.

view_name

O nome da visualização a ser exibida.

Exemplos

Segue-se a definição de visualização para a visualização LA_Venues_v.

```
create view LA_Venues_v as select * from venue where venuecity='Los Angeles';
```

Segue-se um exemplo do comando e saída SHOW VIEW para a visualização definida anteriormente.

```
show view LA_Venues_v;
```

```
SELECT venue.venueid,
venue.venuename,
venue.venuecity,
venue.venuestate,
venue.venueseats
FROM venue WHERE ((venue.venuecity)::text = 'Los Angeles'::text);
```

Segue-se a definição de visualização para a visualização public.Sports_v no esquema public.

```
create view public.Sports_v as select * from category where catgroup='Sports';
```

Segue-se um exemplo do comando e saída SHOW VIEW para a visualização definida anteriormente.

```
show view public.Sports_v;
```

```
SELECT category.catid,
category.catgroup,
category.catname,
category.catdesc
FROM category WHERE ((category.catgroup)::text = 'Sports'::text);
```

START TRANSACTION

Sinônimo da função BEGIN.

Consulte BEGIN.

TRUNCATE

Exclui todas as linhas de uma tabela sem fazer uma varredura de lista: esta operação é uma alternativa mais rápida a uma operação DELETE não qualificada. Para executar um comando TRUNCATE, é necessário ter a permissão TRUNCATE e ser o proprietário da tabela ou um superusuário. Para conceder permissões para truncar uma tabela, use o comando GRANT.

TRUNCATE é muito mais eficiente do que DELETE e não requer VACUUM nem ANALYZE. No entanto, esteja ciente de que TRUNCATE confirma a transação em que é executado.

Sintaxe

```
TRUNCATE [ TABLE ] table_name
```

O comando também funciona em uma visão materializada.

```
TRUNCATE materialized_view_name
```

Parâmetros

TABLE

Palavra-chave opcional.

table_name

Uma tabela temporária ou persistente. Somente o proprietário da tabela ou um superusuário pode truncá-la.

Você pode truncar qualquer tabela, incluindo tabelas com referência em limitações de chave externa.

Você não precisa limpar uma tabela depois de truncá-la.

materialized_view_name

Uma visão materializada.

Você pode truncar uma visão materializada que é usada para Ingestão de streaming para uma visão materializada.

Observações de uso

- O comando TRUNCATE confirma a transação em que é executado. Portanto, você não pode reverter uma operação TRUNCATE, e um comando TRUNCATE pode confirmar outras operações quando confirma a si mesmo.
- As operações TRUNCATE mantêm bloqueios exclusivos quando executadas em visões materializadas de streaming do Amazon Redshift conectadas a qualquer um dos seguintes:
  - Um Amazon Kinesis data stream
  - Um tópico Amazon Managed Streaming for Apache Kafka
  - Um stream externo compatível, como um tópico do Confluent Cloud Kafka

  Para obter mais informações, consulte Ingestão de streaming para uma visão materializada.

Exemplos

Use o comando TRUNCATE para excluir todas as linhas da tabela CATEGORY:

```
truncate category;
```

Tente reverter uma operação TRUNCATE:

```
begin;

truncate date;

rollback;

select count(*) from date;
count
-------
0
(1 row)
```

A tabela DATE permanece vazia após o comando ROLLBACK porque o comando TRUNCATE foi confirmado automaticamente.

O exemplo a seguir usa o comando TRUNCATE para excluir todas as linhas de uma visão materializada.

```
truncate my_materialized_view;
```

Ele exclui todos os registros da visão materializada, deixando a visão materializada e o respectivo esquema intactos. Na consulta, o nome da visão materializada é apenas um exemplo.

UNLOAD

> A criptografia do lado do cliente para os comandos COPY e UNLOAD não estará mais disponível para novos clientes a partir de 30 de abril de 2025. Se você usou a criptografia do lado do cliente com os comandos COPY e UNLOAD nos 12 meses anteriores a 30 de abril de 2025, poderá continuar usando a criptografia do lado do cliente com os comandos COPY ou UNLOAD até 30 de abril de 2026. Após 30 de abril de 2026, você não poderá usar a criptografia do lado do cliente para COPY e UNLOAD. Recomendamos que você passe a usar a criptografia do lado do servidor para COPY e UNLOAD o mais rápido possível. Se você já usa a criptografia do lado do servidor para COPY e UNLOAD, não há nenhuma alteração a fazer e você pode continuar a usá-la sem alterar suas consultas. Para ter mais informações sobre criptografia para COPY e UNLOAD, consulte o parâmetro ENCRYPTED abaixo.

Descarrega o resultado de uma consulta em um ou mais arquivos de texto JSON ou Apache Parquet no Amazon S3, usando a criptografia no servidor Amazon S3 (SSE-S3). Você também pode especificar a criptografia do lado do servidor com uma chave do AWS Key Management Service (SSE-KMS).

Por padrão, o formato do arquivo descarregado é texto delimitado por barra vertical (|).

Você pode gerenciar o tamanho de arquivos no Amazon S3, e, por extensão, o número de arquivos, configurando o parâmetro MAXFILESIZE. Verifique se os intervalos de IP do S3 foram adicionados à sua lista de permissões. Para saber mais sobre os intervalos de IP do S3 necessários, consulte Isolamento de rede.

Você pode descarregar o resultado de uma consulta do Amazon Redshift no data lake do Amazon S3 no Apache Parquet, um formato de armazenamento colunar aberto e eficiente para análise. O formato Parquet é até duas vezes mais rápido de descarregar e consome até seis vezes menos armazenamento no Amazon S3 em comparação com os formatos de texto. Isso permite que você

salve a transformação e o enriquecimento de dados que você fez no Amazon S3 no data lake do Amazon S3 em um formato aberto. Depois, você pode analisar os dados com o Redshift Spectrum e outros serviços da AWS, como Amazon Athena, Amazon EMR e Amazon SageMaker AI.

Para obter mais informações e exemplos de cenários sobre o uso do comando UNLOAD, consulte Descarregar dados no Amazon Redshift.

Permissões e privilégios necessários

Para que o comando UNLOAD seja bem-sucedido, é necessário pelo menos o privilégio SELECT nos dados no banco de dados, com a permissão para gravar no local do Amazon S3. Para obter informações sobre permissões para acessar recursos da AWS para o comando UNLOAD, consulte Permissões para acessar outros recursos da AWS.

Para aplicar permissões com privilégio mínimo, siga estas recomendações para conceder permissões somente ao usuário que está executando o comando, conforme necessário.

- O usuário deve ter o privilégio SELECT nos dados. Para obter informações sobre como limitar privilégios do banco de dados, consulte GRANT.
- O usuário precisa de permissão para assumir o perfil do IAM e gravar no bucket do Amazon S3 na Conta da AWS. Para restringir o acesso de um usuário do banco de dados para assumir um perfil, consulte Restringir acesso a funções do IAM no Guia de gerenciamento do Amazon Redshift.
- O usuário precisa acessar o bucket do Amazon S3. Para restringir a permissão usando uma política de bucket do Amazon S3, consulte Políticas de bucket para o Amazon S3 no Guia do usuário do Amazon Simple Storage Service.

Sintaxe

```
UNLOAD ('select-statement')
TO 's3://object-path/name-prefix'
authorization
[ option, ...]

where authorization is
IAM_ROLE { default | 'SESSION' | 'arn:aws:iam::<Conta da AWS-id-1>:role/<role-
name>[,arn:aws:iam::<Conta da AWS-id-2>:role/<role-name>][,...]' }

where option is
| [ FORMAT [ AS ] ] CSV | PARQUET | JSON
| PARTITION BY ( column_name [, ... ] ) [ INCLUDE ]
```

```
| MANIFEST [ VERBOSE ]
| HEADER
| DELIMITER [ AS ] 'delimiter-char'
| FIXEDWIDTH [ AS ] 'fixedwidth-spec'
| ENCRYPTED [ AUTO ]
| BZIP2
| GZIP
| ZSTD
| ADDQUOTES
| NULL [ AS ] 'null-string'
| ESCAPE
| ALLOWOVERWRITE
| CLEANPATH
| PARALLEL [ { ON | TRUE } | { OFF | FALSE } ]
| MAXFILESIZE [AS] max-size [ MB | GB ]
| ROWGROUPSIZE [AS] size [ MB | GB ]
| REGION [AS] 'aws-region' }
| EXTENSION 'extension-name'
```

Parâmetros

('select-instrução')

Uma consulta SELECT. Os resultados da consulta são descarregados. Na maioria dos casos, vale descarregar dados em ordem de classificação especificando uma cláusula ORDER BY na consulta. Essa abordagem economiza tempo necessário para classificar os dados ao carregar novamente.

A consulta deve estar entre aspas simples como mostrado a seguir:

```
('select * from venue order by venueid')
```

> **Note**
>
> Se sua consulta contiver aspas (por exemplo, para incluir valores literais), coloque o literal entre dois conjuntos de aspas simples — você também deve colocar a consulta entre aspas simples:
>
> ```
> ('select * from venue where venuestate=''NV''')
> ```

TO 's3://object-path/name-prefix'

Caminho completo, incluindo o nome do bucket, para o local no Amazon S3 onde o Amazon Redshift gravará os objetos do arquivo de saída, incluindo o arquivo manifesto se MANIFEST estiver especificado. Os nomes de objetos devem ter o prefixo nome-prefixo. Se você usar `PARTITION BY`, uma barra (/) será automaticamente adicionada ao final do valor name-prefix, se necessário. Para maior segurança, UNLOAD se conecta ao Amazon S3 usando uma conexão HTTPS. Por padrão, UNLOAD grava um ou mais arquivos por fatia. UNLOAD adiciona um número de fatia e um número de peça para o prefixo especificado da seguinte forma:

<object-path>/*<name-prefix><slice-number>*_part_*<part-number>*.

Se MANIFEST for especificado, o arquivo manifesto será redigido conforme o seguinte:

<object_path>/*<name_prefix>*manifest.

Se PARALLEL for especificado como OFF, os arquivos de dados serão gravados da seguinte forma:

<object_path>/*<name_prefix><part-number>*.

UNLOAD cria automaticamente arquivos criptografados usando a criptografia no lado do servidor (SSE) do Amazon S3, incluindo o arquivo manifesto se MANIFEST for usado. O comando COPY lê automaticamente os arquivos criptografados no lado do servidor durante a operação de carregamento. Você pode baixar arquivos criptografados do lado do servidor de forma transparente de seu bucket usando o console do Amazon S3 ou API. Para obter mais informações, consulte Proteção de dados usando criptografia no lado do servidor.

> **Important**
>
> REGION é necessário quando o bucket do Amazon S3 não está na mesma Região da AWS que o banco de dados do Amazon Redshift.

authorization

O comando UNLOAD requer autorização para gravar dados no Amazon S3. O comando UNLOAD usa os mesmos parâmetros do comando COPY para autorização. Para obter mais informações, consulte Parâmetros de autorização na referência de sintaxe do comando COPY.

IAM_ROLE { default | 'SESSION' | 'arn:aws:iam::*<Conta da AWS-id-1>*:role/*<role-name>*' }

Use a palavra-chave padrão para que o Amazon Redshift use a função do IAM definida como padrão e associada ao cluster quando o comando UNLOAD for executado.

Use o nome do recurso da Amazon (ARN) de uma função do IAM que seu cluster usa para autenticação e autorização. O IAM_ROLE pode ser conectado. Para obter mais informações, consulte Encadeamento de funções do IAM no Guia de gerenciamento do Amazon Redshift.

Use a palavra-chave `SESSION` para descarregar dados usando as credenciais da sua sessão atual federada do IAM. Essa opção está disponível somente quando você se conecta ao Amazon Redshift com uma identidade federada do IAM.

Com `SESSION`, o Amazon Redshift usa as mesmas permissões do Amazon S3 que sua identidade federada já tem. O Amazon Redshift não assume um perfil do IAM de cluster, então você não precisa anexar um ao cluster ou grupo de trabalho para esse acesso.

Ao especificar `SESSION`, você não pode combiná-lo com nenhum outro método de autorização.

Para descarregar com `SESSION`, você precisa ter as permissões listadas em Permissões e privilégios necessários. Para ver um exemplo do uso de uma identidade federada, consulte Usar uma identidade federada para gerenciar o acesso do Amazon Redshift aos recursos locais e a tabelas externas do Amazon Redshift Spectrum.

[FORMAT [AS]] CSV | PARQUET | JSON

Palavras-chave para especificar o formato de descarregamento para substituir o formato padrão.

No caso de CSV, descarregue para um arquivo de texto no formato CSV usando um caractere de vírgula (,) como delimitador padrão. Se um campo contiver delimitadores, aspas duplas, caracteres de nova linha ou retornos de carro, o campo no arquivo descarregado será colocado entre aspas duplas. Uma aspas duplas dentro de um campo de dados é recuada por aspas duplas adicionais. Quando nenhuma linha é descarregada, o Amazon Redshift pode gravar objetos vazios do Amazon S3.

No caso de PARQUET, descarregue para um arquivo no formato Apache Parquet versão 1.0. Por padrão, cada grupo de linhas é compactado usando a compactação SNAPPY. Para obter mais informações sobre o formato Apache Parquet, consulte Parquet.

Quando JSON, descarrega para um arquivo JSON com todas as linhas que contêm um objeto JSON, representando um registro completo no resultado da consulta. O Amazon Redshift

é compatível com a gravação de JSON aninhado quando o resultado da consulta contém colunas SUPER. Para criar um objeto JSON válido, o nome de toda coluna da consulta deve ser exclusivo. No arquivo JSON, os valores booleanos são descarregados como `t` ou `f`, e os valores NULL são descarregados como `null`. Quando nenhuma linha é descarregada, o Amazon Redshift não grava objetos do Amazon S3.

As palavras-chave FORMAT e AS são opcionais. Não é possível usar CSV com ESCAPE, FIXEDWIDTH ou ADDQUOTES. Não é possível usar PARQUET com DELIMITER, FIXEDWIDTH, ADDQUOTES, ESCAPE, NULL AS, HEADER, GZIP, BZIP2 ou ZSTD. O comando PARQUET com ENCRYPTED só é compatível com criptografia do lado do servidor com uma chave do AWS Key Management Service (SSE-KMS). Não é possível usar JSON com EADER, FIXEDWIDTH, ADDQUOTES, ESCAPE nem NULL AS.

PARTITION BY (column_name [, ...]) [INCLUDE]

Especifica as chaves de partição para a operação de descarregamento. O comando UNLOAD particiona automaticamente arquivos de saída em pastas de partição com base nos valores de chave de partição, seguindo a convenção Apache Hive. Por exemplo, um arquivo Parquet que pertence à partição setembro de 2019 tem o seguinte prefixo: `s3://amzn-s3-demo-bucket/my_prefix/year=2019/month=September/000.parquet`.

O valor para column_name deve ser uma coluna nos resultados da consulta que está sendo descarregado.

Se você especificar PARTITION BY com a opção INCLUDE, as colunas de partição não serão removidas dos arquivos descarregados.

O Amazon Redshift não oferece suporte a literais de string em cláusulas PARTITION BY.

MANIFEST [VERBOSE]

Cria um arquivo manifesto que lista explicitamente detalhes dos arquivos de dados criados por processo UNLOAD. O manifesto é um arquivo de texto em formato JSON que lista o URL de cada arquivo gravado no Amazon S3.

Se MANIFEST for especificado com a opção VERBOSE, o manifesto incluirá os seguintes detalhes:

- Os nomes das colunas e os tipos de dados, e, para os tipos de dados CHAR, VARCHAR ou NUMERIC, dimensões para cada coluna. Para os tipos de dados CHAR e VARCHAR, a dimensão é o comprimento. Para um tipo de dados DECIMAL ou NUMERIC, as dimensões são precisão e escala.

- A contagem de linhas descarregada para cada arquivo. Se a opção HEADER for especificada, a contagem de linhas incluirá a linha de cabeçalho.
- O tamanho total do arquivo de todos os arquivos descarregados e a contagem total de linhas descarregada para todos os arquivos. Se a opção HEADER for especificada, a contagem de linhas incluirá as linhas de cabeçalho.
- O autor. Autor é sempre "Amazon Redshift".

Você pode especificar VERBOSE somente após MANIFEST.

O arquivo manifesto é gravado no mesmo prefixo de caminho do Amazon S3 que os arquivos descarregados no formato `<object_path_prefix>manifest`. Por exemplo, se UNLOAD especificar o prefixo "`s3://amzn-s3-demo-bucket/venue_`" do caminho do Amazon S3, a localização do arquivo manifesto será "`s3://amzn-s3-demo-bucket/venue_manifest`".

CABEÇALHO

Adiciona uma linha de cabeçalho que contém os nomes de coluna no topo de cada arquivo de saída. Opções de transformação de texto, como CSV, DELIMITER, ADDQUOTES e ESCAPE, também se aplicam à linha de cabeçalho. Não é possível usar HEADER com FIXEDWIDTH.

DELIMITER AS 'delimiter_character'

Especifica o único caractere ASCII usado para separar campos no arquivo de saída, como um caractere de barra vertical (|), uma vírgula (,) ou uma tabulação (\t). O delimitador padrão para arquivos de texto é um caractere de barra. O delimitador padrão para arquivos CSV é uma vírgula. A palavra-chave AS é opcional. Não é possível usar DELIMITER com FIXEDWIDTH. Se os dados contiverem o caractere delimitador, será necessário especificar a opção ESCAPE para ignorar o delimitador, ou usar ADDQUOTES para colocar os dados entre aspas duplas. Como alternativa, especifique um delimitador que não esteja contido nos dados.

FIXEDWIDTH 'fixedwidth_spec'

Descarrega os dados em um arquivo em que a largura de cada coluna tem um tamanho fixo, em vez separadas por um delimitador. A fixedwidth_spec é uma string que especifica o número de colunas e a largura das coluna. A palavra-chave AS é opcional. Como FIXEDWIDTH não trunca dados, a especificação para cada coluna na instrução UNLOAD precisa ser pelo menos do tamanho de entrada mais longa para aquela coluna. O formato de fixedwidth_spec é mostrado a abaixo:

```
'colID1:colWidth1,colID2:colWidth2, ...'
```

Não é possível usar FIXEDWIDTH com DELIMITER ou HEADER.

ENCRYPTED [AUTO]

Especifica que os arquivos de saída no Amazon S3 são criptografados usando a criptografia do lado do servidor do Amazon S3. Se MANIFEST for especificado, o arquivo manifesto também será criptografado. Para obter mais informações, consulte Descarregamento de arquivos de dados criptografados. Se você não especificar o parâmetro ENCRYPTED, UNLOAD criará automaticamente arquivos criptografados usando a criptografia do servidor Amazon S3 com chaves de criptografia gerenciadas (SSE-S3) da AWS.

Para ENCRYPTED, você pode querer descarregar no Amazon S3 usando criptografia do lado do servidor com uma chave do AWS KMS (SSE-KMS). Em caso afirmativo, use o parâmetro KMS_KEY_ID para fornecer o ID da chave. Se você executar um comando UNLOAD para dados usando KMS_KEY_ID, poderá fazer uma operação COPY para os mesmos dados sem especificar uma chave.

Se ENCRYPTED AUTO for usado, o comando UNLOAD busca a chave de criptografia padrão do AWS KMS na propriedade de bucket do Amazon S3 de destino e criptografa os arquivos gravados no Amazon S3 com a chave do AWS KMS. Se o bucket não tiver a chave de criptografia padrão do AWS KMS, UNLOAD criará automaticamente arquivos criptografados usando a criptografia do lado do servidor Amazon Redshift com chaves de criptografia gerenciadas (SSE-S3) da AWS. Você não pode usar essa opção com KMS_KEY_ID ou MASTER_SYMMETRIC_KEY.

KMS_KEY_ID 'key-id'

Especifica o ID de chave para uma chave do AWS Key Management Service (AWS KMS) a ser usada para criptografar arquivos de dados no Amazon S3. Para ter mais informações, consulte O que é o AWS Key Management Service? Se você especificar KMS_KEY_ID, você deve especificar o parâmetro ENCRYPTED também. Para autenticar, use o parâmetro the section called “Usar IAM_ROLE”.

BZIP2

Descarrega dados em um ou mais arquivos compactados bzip2 por fatia. Cada arquivo resultante recebe uma extensão `.bz2`.

GZIP

Descarrega dados em um ou mais arquivos compactados gzip por fatia. Cada arquivo resultante recebe uma extensão `.gz`.

ZSTD

Descarrega dados em um ou mais arquivos compactados Zstandard por fatia. Cada arquivo resultante recebe uma extensão `.zst`.

ADDQUOTES

Coloca aspas em cada campo de dados descarregado para que o Amazon Redshift possa descarregar os valores de dados que contêm o próprio delimitador. Por exemplo, se o delimitador for uma vírgula, você pode descarregar e recarregar os seguintes dados com êxito:

```
"1","Hello, World"
```

Se as aspas não forem adicionadas, a string `Hello, World` será analisada como dois campos separados.

Alguns formatos de saída não são compatíveis com ADDQUOTES.

Se você usar ADDQUOTES, deve especificar REMOVEQUOTES em COPY se recarregar dados.

NULL AS 'string-nula'

Especifica uma string que representa um valor nulo em arquivos de descarregamento. Se essa opção for usada, todos os arquivos de saída contêm a string especificada em vez de todos os valores nulos encontrados nos dados selecionados. Se essa opção não for especificada, os valores nulos serão descarregados como:

- String de tamanho zero para saída delimitada
- String de espaço em branco para saída de largura fixa

Se uma string nula é específica para um tamanho fixo de descarregamento e o tamanho de uma coluna de saída é inferior ao tamanho de string nulo, o seguinte comportamento ocorre:

- Um campo vazio é a saída para colunas sem caracteres
- Um erro é relatado para colunas com caracteres

Ao contrário de outros tipos de dados em que uma string definida pelo usuário representa um valor nulo, o Amazon Redshift exporta as colunas de dados SUPER usando o formato JSON e a representa como nula, conforme determinado pelo formato JSON. Como resultado, as colunas de dados SUPER ignoram a opção NULL [AS] usada nos comandos UNLOAD.

ESCAPE

Para colunas CHAR e VARCHAR em arquivos de descarregamento delimitados, um caractere de escape (\) é posicionado antes de cada ocorrência dos seguintes caracteres:

- Linefeed: `\n`
- Retorno de carro: `\r`
- O caractere delimitador especificado para dados descarregados.
- Caractere de escape: `\`
- Um caractere de aspas: `"` ou `'` (se ambos ESCAPE e ADDQUOTES estiverem especificados no comando UNLOAD).

> **Important**
>
> Se você carregou os dados usando COPY com a opção ESCAPE, também deverá especificar a opção ESCAPE com o comando UNLOAD para gerar o arquivo de saída recíproco. Da mesma maneira, se usar UNLOAD com a opção ESCAPE, você precisará usar ESCAPE quando usar o comando COPY nos mesmos dados.

ALLOWOVERWRITE

Como padrão, UNLOAD apresenta falha se localizar arquivos que possivelmente substituiria. Se ALLOWOVERWRITE for especificado, UNLOAD substituirá arquivos existentes, incluindo o arquivo manifesto.

CLEANPATH

A opção CLEANPATH remove arquivos existentes localizados no caminho do Amazon S3 especificado na cláusula TO antes de descarregar arquivos para o local especificado.

Se você incluir a cláusula PARTITION BY, os arquivos existentes serão removidos somente das pastas de partição para receber novos arquivos gerados pela operação UNLOAD.

Você deve ter a permissão `s3:DeleteObject` no bucket do Amazon S3. Para obter mais informações, consulte Políticas e permissões no Amazon S3 no Guia do usuário do Amazon Simple Storage Service. Os arquivos removidos usando a opção CLEANPATH serão excluídos permanentemente e não podem ser recuperados. Se o bucket de destino do Amazon S3 tiver

o versionamento habilitado, UNLOAD com a opção CLEANPATH não removerá as versões anteriores dos arquivos.

Você não pode especificar a opção CLEANPATH se especificar a opção ALLOWOVERWRITE.

PARALLEL

Por padrão, UNLOAD grava dados em paralelo a vários arquivos, de acordo com o número de fatias no cluster. A opção padrão é ON ou TRUE. Se PARALLEL estiver configurado como OFF ou FALSE, UNLOAD gravará um ou mais arquivos de dados em série, classificado(s) absolutamente de acordo com a cláusula ORDER BY, se houver. O tamanho máximo de um arquivo de dados é 6,2 GB. Dessa forma, por exemplo, se você descarrega 13,4 GB de dados, a opção UNLOAD cria os seguintes três arquivos.

```
s3://amzn-s3-demo-bucket/key000    6.2 GB
s3://amzn-s3-demo-bucket/key001    6.2 GB
s3://amzn-s3-demo-bucket/key002    1.0 GB
```

> **Note**
>
> O comando UNLOAD é projetado para usar processamento paralelo. Recomendamos deixar PARALLEL habilitado na maioria dos casos, especialmente se os arquivos são usados para carregar tabelas usando um comando COPY.

MAXFILESIZE [AS] max-size [MB | GB]

Especifica o tamanho máximo de arquivos criados por UNLOAD no Amazon S3. Especifique um valor decimal entre 5 MB e 6,2 GB. A palavra-chave AS é opcional. A unidade padrão é MB. Se MAXFILESIZE não for especificado, o tamanho de arquivo máximo padrão será 6,2 GB. O tamanho do arquivo de manifesto, se houver, não é afetado por MAXFILESIZE.

ROWGROUPSIZE [AS] size [MB | GB]

Especifica o tamanho dos grupos de linhas. Escolher um tamanho maior pode reduzir o número de grupos de linhas, reduzindo a quantidade de comunicação de rede. Especifique um valor inteiro entre 32 MB e 128 MB. A palavra-chave AS é opcional. A unidade padrão é MB.

Se ROWGROUPSIZE não for especificado, o tamanho padrão será 32 MB. Para usar esse parâmetro, o formato de armazenamento deve ser Parquet, e o tipo de nó deve ser ra3.4xlarge, ra3.16xlarge, rg.4xlarge, rg.12xlarge ou dc2.8xlarge.

REGION [AS] 'aws-region'

Especifica a Região da AWS onde está localizado o bucket do Amazon S3 de destino. O parâmetro REGION é necessário para a operação UNLOAD em um bucket do Amazon S3 que não esteja na mesma Região da AWS que o banco de dados do Amazon Redshift.

O valor de aws_region deve corresponder a uma região da AWS listada na tabela de Regiões e endpoints do Amazon Redshift na Referência geral da AWS.

Por padrão, UNLOAD assume que o bucket do Amazon S3 de destino está localizado na mesma Região da AWS que o banco de dados do Amazon Redshift.

EXTENSÃO “extension-name”

Especifica a extensão do arquivo a ser anexada aos nomes dos arquivos descarregados. O Amazon Redshift não executa nenhuma validação, então você deve verificar se a extensão de arquivo especificada está correta. Se você especificar um método de compactação sem fornecer uma extensão, o Amazon Redshift só adicionará a extensão do método de compactação ao nome do arquivo. Se você não fornecer nenhuma extensão e não especificar um método de compactação, o Amazon Redshift não adicionará nada ao nome do arquivo.

Observações de uso

Usar ESCAPE para todas as operações UNLOAD de texto delimitadas

Quando você usa UNLOAD com um delimitador, seus dados podem incluir esse delimitador ou qualquer um dos caracteres listados na descrição da opção ESCAPE. Nesse caso, você deve usar a opção ESCAPE com o comando UNLOAD. Se você não usar a opção ESCAPE com UNLOAD, poderá ocorrer uma falha nas operações COPY subsequentes que usem os dados descarregados.

> **Important**
>
> É altamente recomendável que você sempre use ESCAPE com os comandos UNLOAD e COPY. A exceção será se você tiver certeza de que seus dados não contêm nenhum delimitador nem outros caracteres que talvez precisem ser ignorados.

Perda de precisão de ponto flutuante

Você pode se deparar com perda de precisão para dados flutuantes sucessivamente descarregados e recarregados.

Cláusula de limitação

A consulta SELECT não pode usar uma cláusula LIMIT na SELECT externa. Por exemplo, o seguinte comando UNLOAD retorna uma falha:

```
unload ('select * from venue limit 10')
to 's3://amzn-s3-demo-bucket/venue_pipe_' iam_role 'arn:aws:iam::0123456789012:role/
MyRedshiftRole';
```

Em vez disso, use uma cláusula LIMIT aninhada, como no exemplo a seguir.

```
unload ('select * from venue where venueid in
(select venueid from venue order by venueid desc limit 10)')
to 's3://amzn-s3-demo-bucket/venue_pipe_' iam_role 'arn:aws:iam::0123456789012:role/
MyRedshiftRole';
```

Você também pode preencher uma tabela usando SELECT…INTO ou CREATE TABLE AS com uma cláusula LIMIT e descarregar da tabela.

Descarregar uma coluna do tipo de dados GEOMETRY

Só é possível descarregar colunas GEOMETRY para texto ou formato CSV. Não é possível descarregar dados GEOMETRY com a opção `FIXEDWIDTH`. Os dados são descarregados na forma hexadecimal do binário bem-conhecido estendido (EWKB). Se o tamanho dos dados EWKB for maior que 4 MB, ocorrerá um aviso porque os dados não poderão ser carregados em uma tabela mais tarde.

Descarregar o tipo de dados HLLSKETCH

Só é possível descarregar colunas HLLSKETCH para texto ou formato CSV. Não é possível descarregar dados HLLSKETCH com a opção `FIXEDWIDTH`. Os dados são descarregados no formato Base64 para esboços de HyperLogLog densos ou no formato JSON para esboços de HyperLogLog esparsos. Para obter mais informações, consulte Funções HyperLogLog.

O exemplo a seguir exporta uma tabela contendo colunas HLLSKETCH para um arquivo.

```
CREATE TABLE a_table(an_int INT, b_int INT);
INSERT INTO a_table VALUES (1,1), (2,1), (3,1), (4,1), (1,2), (2,2), (3,2), (4,2),
 (5,2), (6,2);

CREATE TABLE hll_table (sketch HLLSKETCH);
INSERT INTO hll_table select hll_create_sketch(an_int) from a_table group by b_int;
```

```
UNLOAD ('select * from hll_table') TO 's3://amzn-s3-demo-bucket/unload/'
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole' NULL AS 'null' ALLOWOVERWRITE
 CSV;
```

Descarregar uma coluna do tipo de dados VARBYTE

Só é possível descarregar colunas VARBYTE para texto ou formato CSV. Os dados são descarregados em forma hexadecimal. Não é possível descarregar dados VARBYTE com a opção `FIXEDWIDTH`. Não há suporte para a opção `ADDQUOTES` de UNLOAD para um CSV. Uma coluna VARBYTE não pode ser usada como coluna PARTITIONED BY.

Cláusula FORMAT AS PARQUET

Esteja ciente destas considerações ao usar o FORMAT AS PARQUET:

- O descarregamento para Parquet não usa compactação no nível de arquivo. Cada grupo de linhas é compactado com SNAPPY.
- Se MAXFILESIZE não for especificado, o tamanho de arquivo máximo padrão será 6,2 GB. Você pode usar MAXFILESIZE para especificar um tamanho de arquivo de 5 MB a 6,2 GB. O tamanho real do arquivo é aproximado quando o arquivo está sendo gravado, então ele pode não ser exatamente igual ao número especificado.

  Para maximizar a performance da verificação, o Amazon Redshift tenta criar arquivos Parquet que contenham grupos de linhas de 32 MB igualmente dimensionados. O valor MAXFILESIZE especificado é automaticamente arredondado para baixo para o múltiplo mais próximo de 32 MB. Por exemplo, se você especificar MAXFILESIZE 200 MB, cada arquivo Parquet descarregado será de aproximadamente 192 MB (grupo de linhas de 32 MB x 6 = 192 MB).
- Se uma coluna usar o formato de dados TIMESTAMPTZ, somente os valores de timestamp serão descarregados. As informações de fuso horário não são descarregadas.
- Não especifique prefixos de nome de arquivo que começam com caracteres de sublinhado (_) ou ponto final (.). O Redshift Spectrum trata os arquivos que começam com esses caracteres como arquivos ocultos e os ignora.

Cláusula PARTITION BY

Esteja ciente destas considerações ao usar o PARTITION BY:

- As colunas de partição não são incluídas no arquivo de saída.

- Certifique-se de incluir colunas de partição na consulta SELECT usada na instrução UNLOAD. Você pode especificar qualquer número de colunas de partição no comando UNLOAD. No entanto, há uma limitação de que deve haver pelo menos uma coluna sem partição para fazer parte do arquivo.
- Se a chave-valor de partição for nulo, o Amazon Redshift descarregará automaticamente esses dados em uma partição padrão chamada `partition_column=__HIVE_DEFAULT_PARTITION__`.
- O comando UNLOAD não faz nenhuma chamada para um catálogo externo. Para registrar suas novas partições para fazer parte da tabela externa existente, use um ALTER TABLE ... separado. Comando ADD PARTITION ... Como alternativa, você pode executar um comando CREATE EXTERNAL TABLE para registrar os dados descarregados como uma nova tabela externa. Você também pode usar um crawler do AWS Glue para preencher o catálogo de dados. Para obter mais informações, consulte Definição de crawlers no Guia do desenvolvedor do AWS Glue.
- Se você usar a opção MANIFEST, o Amazon Redshift gerará somente um arquivo manifesto na pasta raiz do Amazon S3.
- Os tipos de dados da coluna que você pode usar como chave de partição são SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, BOOLEAN, CHAR, VARCHAR, DATE e TIMESTAMP.

Usar o privilégio ASSUMEROLE para conceder acesso a uma função do IAM para operações UNLOAD

Para fornecer acesso a usuários e grupos específicos a uma função do IAM para operações UNLOAD, um superusuário pode conceder o privilégio ASSUMEROLE em uma função do IAM a usuários e grupos. Para mais informações, consulte GRANT.

UNLOAD não é compatível com aliases de ponto de acesso do Amazon S3

Não é possível usar os aliases de ponto de acesso do Amazon S3 com comandos UNLOAD.

Exemplos

Para ver exemplos que mostram como usar o comando UNLOAD, consulte Exemplos de UNLOAD.

Exemplos de UNLOAD

Esses exemplos demonstram vários parâmetros do comando UNLOAD. Os dados de amostra de TICKIT são usados em muitos dos exemplos. Para obter mais informações, consulte Banco de dados de exemplo.

> **Note**
>
> Estes exemplos contêm quebras de linha para garantir legibilidade. Não inclua quebras de linha nem espaços na string credentials-args.

Descarregar VENUE para um arquivo delimitado por barras (delimitador padrão)

O exemplo a seguir descarrega a tabela VENUE e grava os dados em `s3://amzn-s3-demo-bucket/unload/`:

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/unload/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Por padrão, UNLOAD grava um ou mais arquivos por fatia. Pressupondo um cluster de dois nós com duas fatias por nó, o exemplo anterior cria estes arquivos em `amzn-s3-demo-bucket`:

```
unload/0000_part_00
unload/0001_part_00
unload/0002_part_00
unload/0003_part_00
```

Para melhor diferenciar os arquivos de saída, você pode incluir um prefixo no local. O exemplo a seguir descarrega a tabela VENUE e grava os dados em `s3://amzn-s3-demo-bucket/unload/venue_pipe_`:

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/unload/venue_pipe_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Como resultado, temos os quatro arquivos na pasta `unload`, mais uma vez pressupondo quatro fatias.

```
venue_pipe_0000_part_00
venue_pipe_0001_part_00
venue_pipe_0002_part_00
venue_pipe_0003_part_00
```

Descarregar tabela LINEITEM para arquivos Parquet particionados

O exemplo a seguir descarrega a tabela LINEITEM no formato Parquet, particionada pela coluna `l_shipdate`.

```
unload ('select * from lineitem')
to 's3://amzn-s3-demo-bucket/lineitem/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
PARQUET
PARTITION BY (l_shipdate);
```

Assumindo que há quatro fatias, os arquivos Parquet resultantes são particionados dinamicamente em várias pastas.

```
s3://amzn-s3-demo-bucket/lineitem/l_shipdate=1992-01-02/0000_part_00.parquet
                                                0001_part_00.parquet
                                                0002_part_00.parquet
                                                0003_part_00.parquet
s3://amzn-s3-demo-bucket/lineitem/l_shipdate=1992-01-03/0000_part_00.parquet
                                                0001_part_00.parquet
                                                0002_part_00.parquet
                                                0003_part_00.parquet
s3://amzn-s3-demo-bucket/lineitem/l_shipdate=1992-01-04/0000_part_00.parquet
                                                0001_part_00.parquet
                                                0002_part_00.parquet
                                                0003_part_00.parquet
...
```

> **Note**
>
> Em alguns casos, o comando UNLOAD usou a opção INCLUDE, conforme mostrado na instrução SQL a seguir.
>
> ```
> unload ('select * from lineitem')
> to 's3://amzn-s3-demo-bucket/lineitem/'
> iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
> PARQUET
> PARTITION BY (l_shipdate) INCLUDE;
> ```

Nesses casos, a coluna `l_shipdate` também está nos dados nos arquivos Parquet. Caso contrário, os dados da coluna `l_shipdate` não estão nos arquivos Parquet.

Descarregue a tabela VENUE para um arquivo JSON

O exemplo a seguir descarrega a tabela VENUE e grava os dados no formato JSON em `s3://amzn-s3-demo-bucket/unload/`.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/unload/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
JSON;
```

As linhas de exemplo são exemplos da tabela VENUE.

```
venueid | venuename                  | venuecity       | venuestate | venueseats
--------+----------------------------+-----------------+------------+-----------
      1 | Pinewood Racetrack         | Akron           | OH         | 0
      2 | Columbus "Crew" Stadium    | Columbus        | OH         | 0
      4 | Community, Ballpark, Arena | Kansas City     | KS         | 0
```

Após o descarregamento para JSON, o formato do arquivo é semelhante ao seguinte.

```
{"venueid":1,"venuename":"Pinewood
 Racetrack","venuecity":"Akron","venuestate":"OH","venueseats":0}
{"venueid":2,"venuename":"Columbus \"Crew\" Stadium
 ","venuecity":"Columbus","venuestate":"OH","venueseats":0}
{"venueid":4,"venuename":"Community, Ballpark, Arena","venuecity":"Kansas
 City","venuestate":"KS","venueseats":0}
```

Descarregar VENUE para um arquivo CSV

O exemplo a seguir descarrega a tabela VENUE e grava os dados no formato CSV em `s3://amzn-s3-demo-bucket/unload/`.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/unload/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
```

```
CSV;
```

Suponha que a tabela VENUE contenha as linhas a seguir.

```
venueid | venuename                  | venuecity       | venuestate | venueseats
--------+----------------------------+-----------------+------------+-----------
      1 | Pinewood Racetrack         | Akron           | OH         | 0
      2 | Columbus "Crew" Stadium    | Columbus        | OH         | 0
      4 | Community, Ballpark, Arena | Kansas City     | KS         | 0
```

O arquivo de descarregamento é semelhante ao seguinte.

```
1,Pinewood Racetrack,Akron,OH,0
2,"Columbus ""Crew"" Stadium",Columbus,OH,0
4,"Community, Ballpark, Arena",Kansas City,KS,0
```

Descarregar VENUE para um arquivo CSV usando um delimitador

O exemplo a seguir descarrega a tabela VENUE e grava os dados no formato CSV usando o caractere de barra vertical (|) como delimitador. O arquivo descarregado é gravado em `s3://amzn-s3-demo-bucket/unload/`. Neste exemplo, a tabela VENUE contém o caractere de barra vertical no valor da primeira linha (`Pinewood Race|track`). Isso é feito para mostrar que o valor no resultado está entre aspas duplas. Aspas duplas são recuadas por aspas duplas e todo o campo é colocado entre aspas duplas.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/unload/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
CSV DELIMITER AS '|';
```

Suponha que a tabela VENUE contenha as linhas a seguir.

```
venueid | venuename                  | venuecity       | venuestate | venueseats
--------+----------------------------+-----------------+------------+-------------
      1 | Pinewood Race|track        | Akron           | OH         | 0
      2 | Columbus "Crew" Stadium    | Columbus        | OH         | 0
      4 | Community, Ballpark, Arena | Kansas City     | KS         | 0
```

O arquivo de descarregamento é semelhante ao seguinte.

```
1|"Pinewood Race|track"|Akron|OH|0
2|"Columbus ""Crew"" Stadium"|Columbus|OH|0
4|Community, Ballpark, Arena|Kansas City|KS|0
```

Descarregar VENUE com um arquivo manifesto

Para criar um arquivo de manifesto, inclua a opção MANIFEST. O exemplo a seguir descarrega a tabela VENUE e grava um arquivo manifesto com arquivos de dados em s3://amzn-s3-demo-bucket/venue_pipe_:

> **Important**
>
> Ao descarregar arquivos com a opção MANIFEST, você deve usar a opção MANIFEST com o comando COPY para carregar os arquivos. Se você usar o mesmo prefixo para carregar os arquivos e não especificar a opção MANIFEST, ocorrerá uma falha na opção COPY porque ela pressupõe que o arquivo de manifesto é um arquivo de dados.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/venue_pipe_' iam_role 'arn:aws:iam::0123456789012:role/
MyRedshiftRole'
manifest;
```

O resultado são estes cinco arquivos:

```
s3://amzn-s3-demo-bucket/venue_pipe_0000_part_00
s3://amzn-s3-demo-bucket/venue_pipe_0001_part_00
s3://amzn-s3-demo-bucket/venue_pipe_0002_part_00
s3://amzn-s3-demo-bucket/venue_pipe_0003_part_00
s3://amzn-s3-demo-bucket/venue_pipe_manifest
```

A seguir, você verá o conteúdo do arquivo manifesto.

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket/tickit/venue_0000_part_00"},
    {"url":"s3://amzn-s3-demo-bucket/tickit/venue_0001_part_00"},
```

```
    {"url":"s3://amzn-s3-demo-bucket/tickit/venue_0002_part_00"},
    {"url":"s3://amzn-s3-demo-bucket/tickit/venue_0003_part_00"}
  ]
}
```

Descarregar VENUE com MANIFEST VERBOSE

Quando você especifica a opção MANIFEST VERBOSE, o arquivo manifesto inclui as seguintes seções:

- A seção `entries` lista o caminho, o tamanho e a contagem de linhas do Amazon S3 para cada arquivo.
- A seção `schema` lista os nomes das colunas, os tipos de dados e a dimensão de cada coluna.
- A seção `meta` mostra o tamanho total do arquivo e a contagem de linhas para todos os arquivos.

O exemplo a seguir descarrega a tabela VENUE usando a opção MANIFEST VERBOSE.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/unload_venue_folder/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
manifest verbose;
```

A seguir, você verá o conteúdo do arquivo manifesto.

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket/venue_pipe_0000_part_00", "meta":
 { "content_length": 32295, "record_count": 10 }},
    {"url":"s3://amzn-s3-demo-bucket/venue_pipe_0001_part_00", "meta":
 { "content_length": 32771, "record_count": 20 }},
    {"url":"s3://amzn-s3-demo-bucket/venue_pipe_0002_part_00", "meta":
 { "content_length": 32302, "record_count": 10 }},
    {"url":"s3://amzn-s3-demo-bucket/venue_pipe_0003_part_00", "meta":
 { "content_length": 31810, "record_count": 15 }}
  ],
  "schema": {
    "elements": [
      {"name": "venueid", "type": { "base": "integer" }},
      {"name": "venuename", "type": { "base": "character varying", 25 }},
      {"name": "venuecity", "type": { "base": "character varying", 25 }},
      {"name": "venuestate", "type": { "base": "character varying", 25 }},
```

```
      {"name": "venueseats", "type": { "base": "character varying", 25 }}
    ]
  },
  "meta": {
    "content_length": 129178,
    "record_count": 55
  },
  "author": {
    "name": "Amazon Redshift",
    "version": "1.0.0"
  }
}
```

Descarregar VENUE com um cabeçalho

O exemplo a seguir descarrega VENUE com uma linha de cabeçalho.

```
unload ('select * from venue where venueseats > 75000')
to 's3://amzn-s3-demo-bucket/unload/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
header
parallel off;
```

O seguinte mostra o conteúdo do arquivo de saída com uma linha de cabeçalho.

```
venueid|venuename|venuecity|venuestate|venueseats
6|New York Giants Stadium|East Rutherford|NJ|80242
78|INVESCO Field|Denver|CO|76125
83|FedExField|Landover|MD|91704
79|Arrowhead Stadium|Kansas City|MO|79451
```

Descarregar VENUE em arquivos menores

Por padrão, o tamanho máximo de arquivo é 6,2 GB. Se os dados do descarregamento forem maiores de 6,2 GB, UNLOAD criará um novo arquivo para cada segmento de dados de 6,2 GB. Para criar arquivos menores, inclua o parâmetro MAXFILESIZE. Pressupondo que o tamanho dos dados no exemplo anterior foi 20 GB, o seguinte comando UNLOAD cria 20 arquivos, cada um com 1 GB.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/unload/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
```

```
maxfilesize 1 gb;
```

Descarregar VENUE em série

Para descarregar em série, especifique PARALLEL OFF. UNLOAD gravará um arquivo por vez, com no máximo 6,2 GB por arquivo.

O exemplo a seguir descarrega a tabela VENUE e grava os dados em série em `s3://amzn-s3-demo-bucket/unload/`.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/unload/venue_serial_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
parallel off;
```

O resultado é um arquivo chamado venue_serial_000.

Se os dados do descarregamento forem maiores de 6,2 GB, UNLOAD criará um novo arquivo para cada segmento de dados de 6,2 GB. O exemplo a seguir descarrega a tabela LINEORDER e grava os dados em série em `s3://amzn-s3-demo-bucket/unload/`.

```
unload ('select * from lineorder')
to 's3://amzn-s3-demo-bucket/unload/lineorder_serial_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
parallel off gzip;
```

O resultado é a seguinte série de arquivos.

```
lineorder_serial_0000.gz
lineorder_serial_0001.gz
lineorder_serial_0002.gz
lineorder_serial_0003.gz
```

Para melhor diferenciar os arquivos de saída, você pode incluir um prefixo no local. O exemplo a seguir descarrega a tabela VENUE e grava os dados em `s3://amzn-s3-demo-bucket/venue_pipe_`:

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/unload/venue_pipe_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Como resultado, temos os quatro arquivos na pasta `unload`, mais uma vez pressupondo quatro fatias.

```
venue_pipe_0000_part_00
venue_pipe_0001_part_00
venue_pipe_0002_part_00
venue_pipe_0003_part_00
```

Carregar VENUE dos arquivos de descarregamento

Para carregar uma tabela de um conjunto de arquivos de descarga, simplesmente inverta o processo usando um comando COPY. O exemplo a seguir cria uma nova tabela, LOADVENUE, e carrega a tabela dos arquivos de dados criados no exemplo anterior.

```
create table loadvenue (like venue);

copy loadvenue from 's3://amzn-s3-demo-bucket/venue_pipe_' iam_role
 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Se tiver usado a opção MANIFEST para criar um arquivo manifesto com seus arquivos de descarga, você poderá carregar os dados usando o mesmo arquivo manifesto. Você faz isso com um comando COPY com a opção MANIFEST. O exemplo a seguir carrega dados usando um arquivo manifesto.

```
copy loadvenue
from 's3://amzn-s3-demo-bucket/venue_pipe_manifest' iam_role
 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
manifest;
```

Descarregar VENUE em arquivos criptografados

O exemplo a seguir descarrega a tabela VENUE para um conjunto de arquivos criptografados usando uma chave do AWS KMS. Se você especificar um arquivo manifesto com a opção ENCRYPTED, o arquivo manifesto também será criptografado. Para obter mais informações, consulte Descarregamento de arquivos de dados criptografados.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/venue_encrypt_kms'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
kms_key_id '1234abcd-12ab-34cd-56ef-1234567890ab'
manifest
```

```
encrypted;
```

O exemplo a seguir descarrega a tabela VENUE para um conjunto de arquivos criptografados usando uma chave simétrica raiz.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/venue_encrypt_cmk'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
master_symmetric_key 'EXAMPLEMASTERKEYtkbjk/OpCwtYSx/M4/t7DMCDIK722'
encrypted;
```

Carregar VENUE dos arquivos criptografados

Para carregar tabelas de um conjunto de arquivos que foram criados usando UNLOAD com a opção ENCRYPT, inverta o processo usando um comando COPY. Com esse comando, use a opção ENCRYPTED e especifique a mesma chave simétrica raiz que foi usada no comando UNLOAD. O exemplo a seguir carrega a tabela LOADVENUE a partir dos arquivos de dados criptografados criados no exemplo anterior.

```
create table loadvenue (like venue);

copy loadvenue
from 's3://amzn-s3-demo-bucket/venue_encrypt_manifest'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
master_symmetric_key 'EXAMPLEMASTERKEYtkbjk/OpCwtYSx/M4/t7DMCDIK722'
manifest
encrypted;
```

Descarregar dados de VENUE para um arquivo delimitado por tabulação

```
unload ('select venueid, venuename, venueseats from venue')
to 's3://amzn-s3-demo-bucket/venue_tab_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter as '\t';
```

Os arquivos de dados de saída são semelhantes a:

```
1 Toyota Park Bridgeview IL 0
2 Columbus Crew Stadium Columbus OH 0
3 RFK Stadium Washington DC 0
```

```
4 CommunityAmerica Ballpark Kansas City KS 0
5 Gillette Stadium Foxborough MA 68756
...
```

Descarregar VENUE para um arquivo de dados de largura fixa

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/venue_fw_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
fixedwidth as 'venueid:3,venuename:39,venuecity:16,venuestate:2,venueseats:6';
```

Os arquivos de dados de saída serão semelhantes a:

```
1  Toyota Park              Bridgeview  IL0
2  Columbus Crew Stadium    Columbus    OH0
3  RFK Stadium              Washington  DC0
4  CommunityAmerica BallparkKansas City KS0
5  Gillette Stadium         Foxborough  MA68756
...
```

Descarregar VENUE para um conjunto de arquivos GZIP compactados delimitados por tabulação

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/venue_tab_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter as '\t'
gzip;
```

Descarregue VENUE em um arquivo de texto compactado com GZIP

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/venue_tab_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
extension 'txt.gz'
gzip;
```

Descarregar dados que contêm um delimitador

Este exemplo usa a opção ADDQUOTES para descarregar dados delimitados por vírgula onde alguns dos campos de dados reais contêm uma vírgula.

Primeiramente, crie uma tabela que contenha aspas.

```
create table location (id int, location char(64));

insert into location values (1,'Phoenix, AZ'),(2,'San Diego, CA'),(3,'Chicago, IL');
```

Em seguida, descarregue os dados usando a opção ADDQUOTES.

```
unload ('select id, location from location')
to 's3://amzn-s3-demo-bucket/location_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter ',' addquotes;
```

Os arquivos de dados descarregados são semelhantes a:

```
1,"Phoenix, AZ"
2,"San Diego, CA"
3,"Chicago, IL"
...
```

Descarregar os resultados de uma consulta de junção

O exemplo a seguir descarrega os resultados de uma consulta de junção que contém uma função de janela.

```
unload ('select venuecity, venuestate, caldate, pricepaid,
sum(pricepaid) over(partition by venuecity, venuestate
order by caldate rows between 3 preceding and 3 following) as winsum
from sales join date on sales.dateid=date.dateid
join event on event.eventid=sales.eventid
join venue on event.venueid=venue.venueid
order by 1,2')
to 's3://amzn-s3-demo-bucket/tickit/winsum'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Os arquivos de saída são semelhantes a:

```
Atlanta|GA|2008-01-04|363.00|1362.00
Atlanta|GA|2008-01-05|233.00|2030.00
Atlanta|GA|2008-01-06|310.00|3135.00
```

```
Atlanta|GA|2008-01-08|166.00|8338.00
Atlanta|GA|2008-01-11|268.00|7630.00
...
```

Descarregar usando NULL AS

UNLOAD apresenta valores nulos de saída como strings vazias por padrão. Os exemplos a seguir mostram como usar NULL AS para substituir valores nulos por uma string de texto.

Para estes exemplos, adicione alguns valores nulos à tabela VENUE.

```
update venue set venuestate = NULL
where venuecity = 'Cleveland';
```

Selecione de VENUE onde VENUESTATE for nulo para confirmar que as colunas contêm valores NULL.

```
select * from venue where venuestate is null;

 venueid |        venuename         | venuecity | venuestate | venueseats
---------+--------------------------+-----------+------------+------------
      22 | Quicken Loans Arena      | Cleveland |            |          0
     101 | Progressive Field        | Cleveland |            |      43345
      72 | Cleveland Browns Stadium | Cleveland |            |      73200
```

Agora, use UNLOAD para descarregar a tabela VENUE usando a opção NULL AS para substituir valores nulos pela string de caractere '`fred`'.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/nulls/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
null as 'fred';
```

O exemplo a seguir do arquivo de descarregamento mostra que os valores nulos foram substituídos por `fred`. Na realidade, os valores para VENUESEATS também eram nulos e foram substituídos por `fred`. Apesar de o tipo de dados para VENUESEATS ser 'inteiro', a opção UNLOAD converte os valores em texto nos arquivos do descarregamento, e a opção COPY os converte de volta em inteiro. Se você está descarregando para um arquivo de largura fixa, a string NULL AS não deve ser superior à largura do campo.

```
248|Charles Playhouse|Boston|MA|0
251|Paris Hotel|Las Vegas|NV|fred
258|Tropicana Hotel|Las Vegas|NV|fred
300|Kennedy Center Opera House|Washington|DC|0
306|Lyric Opera House|Baltimore|MD|0
308|Metropolitan Opera|New York City|NY|0
  5|Gillette Stadium|Foxborough|MA|5
 22|Quicken Loans Arena|Cleveland|fred|0
101|Progressive Field|Cleveland|fred|43345
...
```

Para carregar uma tabela de arquivos de descarregamento, use um comando COPY com a mesma opção NULL AS.

> **Note**
>
> Se você tentar carregar nulos em uma coluna definida como NOT NULL, o comando COPY falhará.

```
create table loadvenuenulls (like venue);

copy loadvenuenulls from 's3://amzn-s3-demo-bucket/nulls/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
null as 'fred';
```

Para confirmar que as colunas contêm nulos, e não apenas strings vazias, selecione a opção LOADVENUENULLS e filtre por nulos.

```
select * from loadvenuenulls where venuestate is null or venueseats is null;

 venueid |        venuename         | venuecity | venuestate | venueseats
---------+--------------------------+-----------+------------+------------
      72 | Cleveland Browns Stadium | Cleveland |            |      73200
     253 | Mirage Hotel             | Las Vegas | NV         |
     255 | Venetian Hotel           | Las Vegas | NV         |
      22 | Quicken Loans Arena      | Cleveland |            |          0
     101 | Progressive Field        | Cleveland |            |      43345
     251 | Paris Hotel              | Las Vegas | NV         |
```

```
...
```

Você pode usar a opção UNLOAD em uma tabela que contenha nulos usando o comportamento padrão NULL AS e depois COPY para copiar os dados de volta para uma tabela usando o comportamento NULL AS. No entanto, todos os campos não numéricos na tabela de destino contêm strings vazias, não nulas. Por padrão, UNLOAD converte nulos em strings vazias (espaços em branco ou de tamanho zero). Uma opção COPY converte strings vazias em NULL para colunas numéricas, mas insere strings vazias em colunas não numéricas. O exemplo a seguir mostra como executar a opção UNLOAD seguida por uma opção COPY usando o comportamento padrão NULL AS.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/nulls/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole' allowoverwrite;

truncate loadvenuenulls;
copy loadvenuenulls from 's3://amzn-s3-demo-bucket/nulls/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Nesse caso, quando você filtra por nulos, somente as linhas em que VENUESEATS continha nulos. Nas linhas em que VENUESTATE continha nulos na tabela (VENUE), VENUESTATE contém strings vazias na tabela de destino (LOADVENUENULLS).

```
select * from loadvenuenulls where venuestate is null or venueseats is null;

 venueid |        venuename         | venuecity | venuestate | venueseats
---------+--------------------------+-----------+------------+------------
     253 | Mirage Hotel             | Las Vegas | NV         |
     255 | Venetian Hotel           | Las Vegas | NV         |
     251 | Paris Hotel              | Las Vegas | NV         |
...
```

Para carregar strings vazias em colunas não numéricas como NULL, inclua a opção EMPTYASNULL ou BLANKSASNULL. É possível usar ambas.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/nulls/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole' allowoverwrite;

truncate loadvenuenulls;
```

```
copy loadvenuenulls from 's3://amzn-s3-demo-bucket/nulls/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole' EMPTYASNULL;
```

Para confirmar que as colunas contêm NULL, e não apenas os espaços em branco ou vazios, selecione a opção LOADVENUENULLS e filtre por nulos.

```
select * from loadvenuenulls where venuestate is null or venueseats is null;

 venueid |        venuename         | venuecity | venuestate | venueseats
---------+--------------------------+-----------+------------+------------
      72 | Cleveland Browns Stadium | Cleveland |            |      73200
     253 | Mirage Hotel             | Las Vegas | NV         |
     255 | Venetian Hotel           | Las Vegas | NV         |
      22 | Quicken Loans Arena      | Cleveland |            |          0
     101 | Progressive Field        | Cleveland |            |      43345
     251 | Paris Hotel              | Las Vegas | NV         |
     ...
```

Descarregar usando o parâmetro ALLOWOVERWRITE

Por padrão, UNLOAD não substitui arquivos existentes no bucket de destino. Por exemplo, se você executar a mesma instrução UNLOAD duas vezes sem modificar os arquivos no bucket de destino, a segunda ocorrência de UNLOAD falha. Para substituir arquivos existentes, incluindo o arquivo manifesto, especifique a opção ALLOWOVERWRITE.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/venue_pipe_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
manifest allowoverwrite;
```

Descarregue a tabela EVENT usando os parâmetros PARALLEL e MANIFEST

Você pode usar o parâmetro UNLOAD em uma tabela em paralelo e gerar um arquivo de manifesto. Os arquivos de dados do Amazon S3 são todos criados no mesmo nível e os nomes têm como sufixo o padrão `0000_part_00`. O arquivo de manifesto está no mesmo nível de pasta dos arquivos de dados e tem como sufixo o texto `manifest`. O SQL a seguir descarrega a tabela EVENT e cria arquivos com o nome base `parallel`

```
unload ('select * from mytickit1.event')
to 's3://amzn-s3-demo-bucket/parallel'
```

```
iam_role 'arn:aws:iam::123456789012:role/MyRedshiftRole'
parallel on
manifest;
```

A lista de arquivos do Amazon S3 é semelhante à seguinte:

```
 Name                        Last modified                          Size
 parallel0000_part_00 -   August 2, 2023, 14:54:39 (UTC-07:00) 52.1 KB
 parallel0001_part_00 -   August 2, 2023, 14:54:39 (UTC-07:00) 53.4 KB
 parallel0002_part_00 -   August 2, 2023, 14:54:39 (UTC-07:00) 52.1 KB
 parallel0003_part_00 -   August 2, 2023, 14:54:39 (UTC-07:00) 51.1 KB
 parallel0004_part_00 -   August 2, 2023, 14:54:39 (UTC-07:00) 54.6 KB
 parallel0005_part_00 -   August 2, 2023, 14:54:39 (UTC-07:00) 53.4 KB
 parallel0006_part_00 -   August 2, 2023, 14:54:39 (UTC-07:00) 54.1 KB
 parallel0007_part_00 -   August 2, 2023, 14:54:39 (UTC-07:00) 55.9 KB
 parallelmanifest       -   August 2, 2023, 14:54:39 (UTC-07:00) 886.0 B
```

O conteúdo do arquivo `parallelmanifest` é semelhante ao seguinte:

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket/parallel0000_part_00", "meta": { "content_length":
 53316 }},
    {"url":"s3://amzn-s3-demo-bucket/parallel0001_part_00", "meta": { "content_length":
 54704 }},
    {"url":"s3://amzn-s3-demo-bucket/parallel0002_part_00", "meta": { "content_length":
 53326 }},
    {"url":"s3://amzn-s3-demo-bucket/parallel0003_part_00", "meta": { "content_length":
 52356 }},
    {"url":"s3://amzn-s3-demo-bucket/parallel0004_part_00", "meta": { "content_length":
 55933 }},
    {"url":"s3://amzn-s3-demo-bucket/parallel0005_part_00", "meta": { "content_length":
 54648 }},
    {"url":"s3://amzn-s3-demo-bucket/parallel0006_part_00", "meta": { "content_length":
 55436 }},
    {"url":"s3://amzn-s3-demo-bucket/parallel0007_part_00", "meta": { "content_length":
 57272 }}
  ]
}
```

Descarregue a tabela EVENT usando os parâmetros PARALLEL OFF e MANIFEST

Você pode usar o parâmetro UNLOAD em uma tabela em série (PARALLEL OFF) e gerar um arquivo de manifesto. Os arquivos de dados do Amazon S3 são todos criados no mesmo nível e os nomes têm como sufixo o padrão `0000`. O arquivo de manifesto está no mesmo nível de pasta dos arquivos de dados e tem como sufixo o texto `manifest`.

```
unload ('select * from mytickit1.event')
to 's3://amzn-s3-demo-bucket/serial'
iam_role 'arn:aws:iam::123456789012:role/MyRedshiftRole'
parallel off
manifest;
```

A lista de arquivos do Amazon S3 é semelhante à seguinte:

```
 Name                    Last modified                       Size
 serial0000          -   August 2, 2023, 15:54:39 (UTC-07:00) 426.7 KB
 serialmanifest      -   August 2, 2023, 15:54:39 (UTC-07:00) 120.0 B
```

O conteúdo do arquivo `serialmanifest` é semelhante ao seguinte:

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket/serial000", "meta": { "content_length": 436991 }}
  ]
}
```

Descarregue a tabela EVENT usando os parâmetros PARTITION BY e MANIFEST

Você pode usar o parâmetro UNLOAD em uma tabela por partição e gerar um arquivo de manifesto. Uma nova pasta é criada no Amazon S3 com pastas de partição secundárias e os arquivos de dados nas pastas secundárias com um padrão de nome semelhante a `0000_par_00`. O arquivo de manifesto está no mesmo nível de pasta que as pastas secundárias com o nome `manifest`.

```
unload ('select * from mytickit1.event')
to 's3://amzn-s3-demo-bucket/partition'
iam_role 'arn:aws:iam::123456789012:role/MyRedshiftRole'
partition by (eventname)
```

```
manifest;
```

A lista de arquivos do Amazon S3 é semelhante à seguinte:

```
Name                    Type     Last modified                     Size

partition            Folder
```

Na pasta `partition` encontram-se pastas secundárias com o nome da partição e o arquivo de manifesto. Veja a seguir a parte inferior da lista de pastas na pasta `partition`, semelhante à seguinte:

```
Name                    Type      Last modified                      Size

...
eventname=Zucchero/    Folder
eventname=Zumanity/    Folder
eventname=ZZ Top/      Folder
manifest               -     August 2, 2023, 15:54:39 (UTC-07:00) 467.6 KB
```

Na pasta `eventname=Zucchero/` estão os arquivos de dados semelhantes aos seguintes:

```
Name               Last modified                       Size
0000_part_00 -   August 2, 2023, 15:59:19 (UTC-07:00) 70.0 B
0001_part_00 -   August 2, 2023, 15:59:16 (UTC-07:00) 106.0 B
0002_part_00 -   August 2, 2023, 15:59:15 (UTC-07:00) 70.0 B
0004_part_00 -   August 2, 2023, 15:59:17 (UTC-07:00) 141.0 B
0006_part_00 -   August 2, 2023, 15:59:16 (UTC-07:00) 35.0 B
0007_part_00 -   August 2, 2023, 15:59:19 (UTC-07:00) 108.0 B
```

A parte inferior do conteúdo do arquivo `manifest` é semelhante ao seguinte:

```
{
  "entries": [
    ...
    {"url":"s3://amzn-s3-demo-bucket/partition/eventname=Zucchero/007_part_00", "meta":
 { "content_length": 108 }},
```

```
    {"url":"s3://amzn-s3-demo-bucket/partition/eventname=Zumanity/007_part_00", "meta":
 { "content_length": 72 }}
  ]
}
```

Descarregue a tabela EVENT usando os parâmetros MAXFILESIZE, ROWGROUPSIZE e MANIFEST

Você pode usar o parâmetro UNLOAD em uma tabela em paralelo e gerar um arquivo de manifesto. Os arquivos de dados do Amazon S3 são todos criados no mesmo nível e os nomes têm como sufixo o padrão `0000_part_00`. Os arquivos de dados do Parquet gerados são limitados a 256 MB e o tamanho do grupo de linhas a 128 MB. O arquivo de manifesto está no mesmo nível de pasta dos arquivos de dados e tem como sufixo `manifest`.

```
unload ('select * from mytickit1.event')
to 's3://amzn-s3-demo-bucket/eventsize'
iam_role 'arn:aws:iam::123456789012:role/MyRedshiftRole'
maxfilesize 256 MB
rowgroupsize 128 MB
parallel on
parquet
manifest;
```

A lista de arquivos do Amazon S3 é semelhante à seguinte:

```
 Name                            Type      Last modified                         Size
 eventsize0000_part_00.parquet parquet August 2, 2023, 17:35:21 (UTC-07:00) 24.5 KB
 eventsize0001_part_00.parquet parquet August 2, 2023, 17:35:21 (UTC-07:00) 24.8 KB
 eventsize0002_part_00.parquet parquet August 2, 2023, 17:35:21 (UTC-07:00) 24.4 KB
 eventsize0003_part_00.parquet parquet August 2, 2023, 17:35:21 (UTC-07:00) 24.0 KB
 eventsize0004_part_00.parquet parquet August 2, 2023, 17:35:21 (UTC-07:00) 25.3 KB
 eventsize0005_part_00.parquet parquet August 2, 2023, 17:35:21 (UTC-07:00) 24.8 KB
 eventsize0006_part_00.parquet parquet August 2, 2023, 17:35:21 (UTC-07:00) 25.0 KB
 eventsize0007_part_00.parquet parquet August 2, 2023, 17:35:21 (UTC-07:00) 25.6 KB
 eventsizemanifest                 -       August 2, 2023, 17:35:21 (UTC-07:00) 958.0 B
```

O conteúdo do arquivo `eventsizemanifest` é semelhante ao seguinte:

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket/eventsize0000_part_00.parquet", "meta":
 { "content_length": 25130 }},
    {"url":"s3://amzn-s3-demo-bucket/eventsize0001_part_00.parquet", "meta":
 { "content_length": 25428 }},
    {"url":"s3://amzn-s3-demo-bucket/eventsize0002_part_00.parquet", "meta":
 { "content_length": 25025 }},
    {"url":"s3://amzn-s3-demo-bucket/eventsize0003_part_00.parquet", "meta":
 { "content_length": 24554 }},
    {"url":"s3://amzn-s3-demo-bucket/eventsize0004_part_00.parquet", "meta":
 { "content_length": 25918 }},
    {"url":"s3://amzn-s3-demo-bucket/eventsize0005_part_00.parquet", "meta":
 { "content_length": 25362 }},
    {"url":"s3://amzn-s3-demo-bucket/eventsize0006_part_00.parquet", "meta":
 { "content_length": 25647 }},
    {"url":"s3://amzn-s3-demo-bucket/eventsize0007_part_00.parquet", "meta":
 { "content_length": 26256 }}
  ]
}
```

UPDATE

Tópicos

- Sintaxe
- Parâmetros
- Observações de uso
- Exemplos da instrução UPDATE

Atualiza valores em uma ou mais colunas de tabela quando uma condição é atendida.

> **Note**
>
> O tamanho máximo de uma única instrução SQL é 16 MB.

Sintaxe

```
[ WITH [RECURSIVE] common_table_expression [, common_table_expression , ...] ]
```

```
            UPDATE table_name [ [ AS ] alias ] SET column = { expression | DEFAULT }
 [,...]

[ FROM fromlist ]
[ WHERE condition ]
```

Parâmetros

Cláusula WITH

Cláusula opcional que especifica uma ou mais expressões de tabela-comum. Consulte Cláusula WITH.

table_name

Uma tabela temporária ou persistente. Somente o proprietário da tabela ou um usuário com o privilégio UPDATE na tabela pode atualizar linhas. Se usar a cláusula FROM ou selecionar das tabelas em uma expressão ou condição, você deve ter o privilégio SELECT nessas tabelas. Você não pode fornecer um alias à tabela aqui. No entanto, você pode especificar um alias na cláusula FROM.

> **Note**
>
> As tabelas externas do Amazon Redshift Spectrum são de somente leitura. Não é possível usar UPDATE uma tabela externa.

alias

Nome alternativo temporário para uma tabela de destino. Os aliases são opcionais. A palavra-chave AS é sempre opcional.

SET coluna =

Uma ou mais colunas que você deseja modificar. As colunas que não estão listadas mantêm seus valores atuais. Não inclui o nome da tabela na especificação de uma coluna de destino. Por exemplo, `UPDATE tab SET tab.col = 1` é inválido.

expression

Uma expressão que define o novo valor para a coluna especificada.

DEFAULT

Atualiza a coluna com o valor padrão atribuído à coluna na instrução CREATE TABLE.

FROM listadetabelas

Você pode atualizar uma tabela fazendo referência às informações em outras tabelas. Liste essas outras tabelas na cláusula FROM ou use uma subconsulta como parte da condição WHERE. As tabelas listadas na cláusula FROM podem ter aliases. Se você precisar incluir a tabela de destino da instrução UPDATE na lista, use um alias.

WHERE condição

Cláusula opcional que restringe atualizações em linhas que correspondem a uma condição. Quando a condição retorna como `true`, as colunas SET especificadas são atualizadas. A condição pode ser um predicado simples em uma coluna ou uma condição baseada no resultado de uma subconsulta.

Você pode nomear qualquer tabela na subconsulta, incluindo a tabela de destino para UPDATE.

Observações de uso

Depois de atualizar um grande número de linhas em uma tabela:

- Limpe a tabela para recuperar espaço e reclassificar as linhas.
- Analise a tabela para atualizar as estatísticas do planejador de consulta.

As junções esquerda, direita e externa completa não são compatíveis com a cláusula FROM de uma instrução UPDATE; elas retornam o seguinte erro:

```
ERROR: Target table must be part of an equijoin predicate
```

Se você precisar especificar uma junção externa, use uma subconsulta na cláusula WHERE da instrução UPDATE.

Se sua instrução UPDATE exigir uma junção automática à tabela de destino, você precisa especificar a condição de junção, além dos critérios da cláusula WHERE que qualificam as linhas à operação de atualização. Geralmente, quando ocorre uma junção dos próprios dados da tabela de destino ou com outra tabela, uma prática recomendada é usar uma subconsulta que separe claramente as condições de junção dos critérios que qualifiquem linhas para atualização.

Consultas UPDATE com várias correspondências por linha lançam um erro quando o parâmetro de configuração `error_on_nondeterministic_update` é definido como true. Para obter mais informações, consulte error_on_nondeterministic_update.

É possível atualizar uma coluna GENERATED BY DEFAULT AS IDENTITY. As colunas definidas como GENERATED BY DEFAULT AS IDENTITY podem ser atualizadas com os valores fornecidos. Para obter mais informações, consulte GENERATED BY DEFAULT AS IDENTITY.

Exemplos da instrução UPDATE

Para obter mais informações sobre as tabelas usadas nos exemplos a seguir, consulte Banco de dados de exemplo.

A tabela CATEGORY no banco de dados TICKIT contém as seguintes linhas:

```
+-------+----------+-----------+--------------------------------------------+
| catid | catgroup |  catname  |                  catdesc                   |
+-------+----------+-----------+--------------------------------------------+
| 5     | Sports   | MLS       | Major League Soccer                        |
| 11    | Concerts | Classical | All symphony, concerto, and choir concerts |
| 1     | Sports   | MLB       | Major League Baseball                      |
| 6     | Shows    | Musicals  | Musical theatre                            |
| 3     | Sports   | NFL       | National Football League                   |
| 8     | Shows    | Opera     | All opera and light opera                  |
| 2     | Sports   | NHL       | National Hockey League                     |
| 9     | Concerts | Pop       | All rock and pop music concerts            |
| 4     | Sports   | NBA       | National Basketball Association            |
| 7     | Shows    | Plays     | All non-musical theatre                    |
| 10    | Concerts | Jazz      | All jazz singers and bands                 |
+-------+----------+-----------+--------------------------------------------+
```

Atualizar uma tabela com base em um intervalo de valores

Atualize a coluna de CATGROUP com base em um intervalo de valores na coluna de CATID.

```
UPDATE category
SET catgroup='Theatre'
WHERE catid BETWEEN 6 AND 8;

SELECT * FROM category
WHERE catid BETWEEN 6 AND 8;
```

```
+-------+----------+----------+---------------------------+
| catid | catgroup | catname  |          catdesc          |
+-------+----------+----------+---------------------------+
| 6     | Theatre  | Musicals | Musical theatre           |
| 7     | Theatre  | Plays    | All non-musical theatre   |
| 8     | Theatre  | Opera    | All opera and light opera |
+-------+----------+----------+---------------------------+
```

Atualizar uma tabela com base em um valor atual

Atualize as colunas CATNAME e CATDESC com base em seu valor CATGROUP atual:

```
UPDATE category
SET catdesc=default, catname='Shows'
WHERE catgroup='Theatre';

SELECT * FROM category
WHERE catname='Shows';

+-------+----------+---------+---------+
| catid | catgroup | catname | catdesc |
+-------+----------+---------+---------+
| 6     | Theatre  | Shows   | NULL    |
| 7     | Theatre  | Shows   | NULL    |
| 8     | Theatre  | Shows   | NULL    |
+-------+----------+---------+---------+)
```

Nesse caso, a coluna CATDESC foi definida como nula, pois nenhum valor padrão foi definido quando a tabela foi criada.

Execute os seguintes comandos para configurar os dados da tabela CATEGORY de volta para os valores originais:

```
TRUNCATE category;

COPY category
FROM 's3://redshift-downloads/tickit/category_pipe.txt'
DELIMITER '|'
IGNOREHEADER 1
REGION 'us-east-1'
IAM_ROLE default;
```

Atualizar uma tabela com base no resultado de uma subconsulta da cláusula WHERE

Atualize a tabela CATEGORY com base no resultado de uma subconsulta na cláusula WHERE:

```
UPDATE category
SET catdesc='Broadway Musical'
WHERE category.catid IN
(SELECT category.catid FROM category
JOIN event ON category.catid = event.catid
JOIN venue ON venue.venueid = event.venueid
JOIN sales ON sales.eventid = event.eventid
WHERE venuecity='New York City' AND catname='Musicals');
```

Veja a tabela atualizada:

```
SELECT * FROM category ORDER BY catid;

+-------+----------+-----------+--------------------------------------------+
| catid | catgroup |  catname  |                  catdesc                   |
+-------+----------+-----------+--------------------------------------------+
| 2     | Sports   | NHL       | National Hockey League                     |
| 3     | Sports   | NFL       | National Football League                   |
| 4     | Sports   | NBA       | National Basketball Association            |
| 5     | Sports   | MLS       | Major League Soccer                        |
| 6     | Shows    | Musicals  | Broadway Musical                           |
| 7     | Shows    | Plays     | All non-musical theatre                    |
| 8     | Shows    | Opera     | All opera and light opera                  |
| 9     | Concerts | Pop       | All rock and pop music concerts            |
| 10    | Concerts | Jazz      | All jazz singers and bands                 |
| 11    | Concerts | Classical | All symphony, concerto, and choir concerts |
+-------+----------+-----------+--------------------------------------------+
```

Atualizar uma tabela com base no resultado de uma subconsulta da cláusula WITH

Para atualizar a tabela CATEGORY com base no resultado de uma subconsulta usando a cláusula WITH, use o exemplo a seguir.

```
WITH u1 as (SELECT catid FROM event ORDER BY catid DESC LIMIT 1)
UPDATE category SET catid='200' FROM u1 WHERE u1.catid=category.catid;

SELECT * FROM category ORDER BY catid DESC LIMIT 1;
```

```
+-------+----------+---------+---------------------------------+
| catid | catgroup | catname |             catdesc             |
+-------+----------+---------+---------------------------------+
| 200   | Concerts | Pop     | All rock and pop music concerts |
+-------+----------+---------+---------------------------------+
```

Atualizar uma tabela com base no resultado de uma condição de junção

Atualize as 11 linhas originais na tabela CATEGORY com base na correspondência das linhas CATID na tabela EVENT:

```
UPDATE category SET catid=100
FROM event
WHERE event.catid=category.catid;

SELECT * FROM category ORDER BY catid;

+-------+----------+-----------+--------------------------------------------+
| catid | catgroup |  catname  |                 catdesc                    |
+-------+----------+-----------+--------------------------------------------+
| 2     | Sports   | NHL       | National Hockey League                     |
| 3     | Sports   | NFL       | National Football League                   |
| 4     | Sports   | NBA       | National Basketball Association            |
| 5     | Sports   | MLS       | Major League Soccer                        |
| 10    | Concerts | Jazz      | All jazz singers and bands                 |
| 11    | Concerts | Classical | All symphony, concerto, and choir concerts |
| 100   | Concerts | Pop       | All rock and pop music concerts            |
| 100   | Shows    | Plays     | All non-musical theatre                    |
| 100   | Shows    | Opera     | All opera and light opera                  |
| 100   | Shows    | Musicals  | Broadway Musical                           |
+-------+----------+-----------+--------------------------------------------+
```

Observe que a tabela EVENT está listada na cláusula FROM e a condição de junção à tabela de destino é definida na cláusula WHERE. Somente quatro linhas qualificadas para a atualização. Essas quatro linhas são as linhas cujos os valores CATID eram originalmente 6, 7, 8 e 9; somente essas quatro categorias são representadas na tabela EVENT:

```
SELECT DISTINCT catid FROM event;

+-------+
| catid |
+-------+
```

```
| 6     |
| 7     |
| 8     |
| 9     |
+-------+
```

Atualize as 11 linhas originais na tabela CATEGORY, estendendo o exemplo anterior e adicionando outra condição à cláusula WHERE. Devido à restrição na coluna CATGROUP, somente uma linha se qualifica para atualização (embora quatro linhas se qualifiquem para junção).

```
UPDATE category SET catid=100
FROM event
WHERE event.catid=category.catid
AND catgroup='Concerts';

SELECT * FROM category WHERE catid=100;

+-------+----------+---------+---------------------------------+
| catid | catgroup | catname |             catdesc             |
+-------+----------+---------+---------------------------------+
| 100   | Concerts | Pop     | All rock and pop music concerts |
+-------+----------+---------+---------------------------------+
```

Uma forma alternativa de gravar este exemplo é:

```
UPDATE category SET catid=100
FROM event JOIN category cat ON event.catid=cat.catid
WHERE cat.catgroup='Concerts';
```

A vantagem desta abordagem é que os critérios de junção são separados claramente de quaisquer outros critérios que qualifiquem linhas para atualização. Observe o uso do alias CAT para a tabela CATEGORY na cláusula FROM.

Atualizações com junções externas na cláusula FROM

O exemplo anterior mostrou uma junção interna especificada na cláusula FROM de uma instrução UPDATE. O exemplo a seguir retorna um erro porque a cláusula FROM não é compatível com junções externas à tabela de destino:

```
UPDATE category SET catid=100
FROM event LEFT JOIN category cat ON event.catid=cat.catid
```

```
WHERE cat.catgroup='Concerts';
ERROR:  Target table must be part of an equijoin predicate
```

Se junções externas forem necessárias para a instrução UPDATE, você pode mover a sintaxe de junção externa para uma subconsulta:

```
UPDATE category SET catid=100
FROM
(SELECT event.catid FROM event LEFT JOIN category cat ON event.catid=cat.catid)
 eventcat
WHERE category.catid=eventcat.catid
AND catgroup='Concerts';
```

Atualizações com colunas de outra tabela na cláusula SET

Para atualizar a tabela listing no banco de dados de amostra TICKIT com valores da tabela sales, use o exemplo a seguir.

```
SELECT listid, numtickets FROM listing WHERE sellerid = 1 ORDER BY 1 ASC LIMIT 5;

+--------+------------+
| listid | numtickets |
+--------+------------+
| 100423 | 4          |
| 108334 | 24         |
| 117150 | 4          |
| 135915 | 20         |
| 205927 | 6          |
+--------+------------+

UPDATE listing
SET numtickets = sales.sellerid
FROM sales
WHERE sales.sellerid = 1 AND listing.sellerid = sales.sellerid;

SELECT listid, numtickets FROM listing WHERE sellerid = 1 ORDER BY 1 ASC LIMIT 5;

+--------+------------+
| listid | numtickets |
+--------+------------+
| 100423 | 1          |
| 108334 | 1          |
| 117150 | 1          |
```

```
| 135915 | 1          |
| 205927 | 1          |
+--------+------------+
```

USE

Altera o banco de dados no qual as consultas são executadas. SHOW USE aponta para o banco de dados usado mais recentemente com o comando USE. RESET USE redefine o banco de dados usado. Isso significa que, se o banco de dados não for especificado no SQL, os objetos serão pesquisados no banco de dados atual.

Sintaxe

```
USE database
```

Exemplos

Suponha que haja dois bancos de dados, `dev` e `pdb`, e `pdb2`. Considere que haja duas tabelas `t` nos esquemas públicos de cada um dos bancos de dados. Primeiro, insira os dados em tabelas de diferentes bancos de dados:

```
dev=# insert into dev.public.t values (1);
INSERT 0 1
dev=# insert into pdb.public.t values (2);
INSERT 0 1
```

Se você não definir um banco de dados explicitamente, o sistema usará seu banco de dados conectado. Verifique o contexto atual do banco de dados:

```
dev=# show use;
Use Database

(1 row)
dev=> show search_path;
search_path
$user, public
(1 row)
```

Se, ao consultar a tabela `t`, não for especificado um banco de dados, o sistema usará a tabela em seu banco de dados atual:

```
dev=# select * from t;
c
----
1
(1 row)
```

Use o comando `use` para alternar bancos de dados sem alterar sua conexão:

```
dev=# use pdb;
USE
dev=# show use;
 Use Database
--------------
 pdb
(1 row)
dev=# select * from t;
id
----
2
(1 row)
```

Também é possível especificar explicitamente o esquema.

```
dev=# select * from public.t;
id
----
2
(1 row)
```

Agora você pode criar tabelas em esquemas diferentes no banco de dados atual:

```
dev=# create table s1.t(id int);
CREATE TABLE
dev=# insert into pdb.s1.t values (3);
INSERT 0 1
```

O caminho de pesquisa determina quais objetos do esquema são acessados quando você não especifica um esquema:

```
dev=# set search_path to public, s1;
SET
```

```
dev=# select * from t;
 id
----
  2
(1 row)
```

Altere a ordem dos esquemas para acessar tabelas diferentes:

```
dev=# set search_path to s1, public;
SET
dev=# show search_path;
 search_path
-------------
 s1, public
(1 row)
dev=# select * from t;
 id
----
  3
(1 row)
```

Mude para outro banco de dados enquanto mantém sua conexão original:

```
dev=# show use;
 Use Database
--------------
 pdb
(1 row)
dev=# use pdb2;
USE
dev=# show use;
 Use Database
--------------
 pdb2
(1 row)
```

Ao alternar bancos de dados, o caminho de pesquisa é redefinido para o padrão:

```
dev=# show search_path;
  search_path
---------------
 $user, public
```

```
(1 row)
```

Crie uma tabela e insira dados no banco de dados atual:

```
dev=# create table pdb2.public.t(id int);
CREATE TABLE
dev=# insert into pdb2.public.t values (4);
INSERT 0 1
dev=# select * from t;
 id
----
  4
(1 row)
```

Nas transações, você pode gravar no banco de dados atual e ler de qualquer banco de dados usando a notação de três partes. Isso também inclui o banco de dados conectado:

```
dev=# show use;
 Use Database
--------------
 pdb2
(1 row)

dev=# BEGIN;
BEGIN
dev=# select * from t;
 id
----
  4
(1 row)

dev=# insert into t values (5);
INSERT 0 1
dev=# select * from t;
 id
----
  4
  5
(2 rows)

dev=# select * from pdb.public.t;
 id
----
```

```
  2
(1 row)

dev=# select * from dev.public.t;
 id
----
  1
(1 row)
```

Redefina para seu banco de dados conectado. Observe que isso não só reverte para o banco de dados pdb usado anteriormente, mas redefine para o banco de dados conectado. O caminho de pesquisa também muda para o padrão:

```
dev=# RESET USE;
RESET
dev=# select * from t;
c
----
1
(1 row)
dev=# show use;
 Use Database
--------------

(1 row)

dev=# show search_path;
  search_path
---------------
 $user, public
(1 row)
```

Você pode alterar os bancos de dados no início de uma transação, mas não depois de executar consultas:

```
dev=# BEGIN;
BEGIN
dev=# use pdb;
USE
dev=# use pdb2;
USE
dev=# use pdb;
```

```
USE
dev=# select * from t;
 id
----
  2
(1 row)
dev=# use pdb2;
ERROR:  USEd Database cannot be set or reset inside a transaction after another
 command.
dev=# rollback;
ROLLBACK
(1 row)
```

ARNs do Catálogo de Dados

Primeiro, crie tabelas em esquemas e catálogos diferentes para demonstrar consultas entre catálogos. Primeiro, crie tabelas no banco de dados conectado.

```
dev=# CREATE TABLE dev.public.t (col INT);
dev=# INSERT INTO dev.public.t VALUES (1);
dev=# CREATE SCHEMA write_schema;
dev=# CREATE TABLE dev.write_schema.t (state char (2));
dev=# INSERT INTO dev.write_schema.t VALUES ('WA');
```

Depois, crie tabelas semelhantes em um catálogo diferente. Isso demonstra como trabalhar com bancos de dados de consulta entre catálogos.

```
dev=# CREATE TABLE my_db@my_catalog.public.t (col INT);
dev=# INSERT INTO my_db@my_catalog.public.t VALUES (100);
dev=# CREATE SCHEMA my_db@my_catalog.write_schema;
dev=# CREATE TABLE my_db@my_catalog.write_schema.t (state char (2));
dev=# INSERT INTO my_db@my_catalog.write_schema.t VALUES ('CA');
```

Verifique o contexto atual do banco de dados. Se você não definir um banco de dados explicitamente, o sistema usará o banco de dados conectado.

```
dev=# SHOW USE;
 Use Database
--------------

(1 row)

dev=# SHOW search_path;
  search_path
---------------
 $user, public
(1 row)

dev=# SELECT * FROM t;
 col
-----
   1
(1 row)
```

Defina o banco de dados USEd para consultar tabelas em um catálogo diferente.

```
dev=# USE my_db@my_catalog;

dev=# SHOW USE;
            Use Database
-------------------------------------
 my_db@my_catalog
(1 row)

dev=# SHOW search_path;
  search_path
---------------
 $user, public
(1 row)
```

Ao consultar a tabela t, os resultados virão do banco de dados de consulta entre catálogos.

```
dev=# SELECT * FROM t;
 col
-----
 100
```

```
(1 row)

dev=# SELECT * FROM public.t;
 col
-----
 100
(1 row)

dev=# SELECT * FROM my_db@my_catalog.public.t;
 col
-----
 100
(1 row)
```

Altere o caminho de pesquisa para acessar tabelas em diferentes esquemas dentro do banco de dados USEd.

```
dev=# SET search_path to write_schema;

dev=# SHOW search_path;
 search_path
--------------
 write_schema
(1 row)

dev=# SELECT * FROM t;
 state
-------
 CA
(1 row)

dev=# SELECT * FROM write_schema.t;
 state
-------
 CA
(1 row)

dev=# SELECT * FROM my_db@my_catalog.write_schema.t;
 state
-------
 CA
```

```
(1 row)
```

Embora USE esteja definido como um banco de dados de consulta entre catálogos, ainda é possível consultar explicitamente o banco de dados original.

```
dev=# SELECT * FROM dev.write_schema.t;
 state
-------
 WA
(1 row)
```

Redefina o banco de dados USEd para que consulte novamente os objetos no banco de dados conectado.

```
dev=# RESET USE;

dev=# SHOW USE;
 Use Database
--------------

(1 row)
```

Observe que o search_path é redefinido quando USE é redefinido.

```
dev=# SHOW search_path;
  search_path
---------------
 $user, public
(1 row)
```

Após a redefinição, as consultas são direcionadas ao banco de dados conectado original.

```
dev=# SELECT * FROM t;
```

```
 col
-----
   1
(1 row)

dev=# SELECT * FROM public.t;
 col
-----
   1
(1 row)

dev=# SELECT * FROM dev.public.t;
 col
-----
   1
(1 row)
```

Você pode modificar o caminho de pesquisa no banco de dados original para acessar esquemas diferentes.

```
dev=# SET search_path to write_schema;

dev=# SHOW search_path;
 search_path
--------------
 write_schema
(1 row)

dev=# SELECT * FROM t;
 state
-------
 WA
(1 row)

dev=# SELECT * FROM write_schema.t;
 state
-------
 WA
(1 row)

dev=# SELECT * FROM dev.write_schema.t;
```

```
 state
-------
 WA
(1 row)
```

VACUUM

Reclassifica linhas e recupera espaço em qualquer tabela especificada ou em todas as tabelas no banco de dados atual.

> **Note**
>
> Somente usuários com as permissões de tabela necessárias podem efetivamente limpar uma tabela. Se VACUUM for executado sem as permissões necessárias de tabela, a operação será concluída com êxito, mas sem efeito. Para obter uma lista de permissões de tabela válidas para executar efetivamente VACUUM, consulte a seção Privilégios obrigatórios a seguir.

O Amazon Redshift classifica os dados automaticamente e executa VACUUM DELETE em segundo plano. Isso reduz a necessidade de executar o comando VACUUM. Para obter mais informações, consulte Vacuum de tabelas.

Por padrão, VACUUM ignora a fase de classificação para qualquer tabela em que mais de 95 por cento das linhas da tabela já estejam classificadas. Ignorar a fase de classificação pode melhorar significativamente a performance de VACUUM. Para alterar ou excluir o limite de classificação padrão para uma única tabela, inclua o nome da tabela e o parâmetro TO limite PERCENT ao executar o comando VACUUM.

Os usuários podem acessar tabelas enquanto elas estão sendo limpas. É possível executar consultas e operações de gravação enquanto uma tabela está sendo limpa, mas quando comandos de linguagem de manipulação de dados (DML) e uma limpeza são executados simultaneamente, ambos podem levar mais tempo. Se você executar instruções UPDATE e DELETE durante uma limpeza, a performance do sistema pode ser reduzida. VACUUM DELETE bloqueia temporariamente operações de atualização e exclusão.

O Amazon Redshift executa automaticamente uma limpeza DELETE ONLY em segundo plano. A operação de limpeza automática é pausada quando os usuários executam operações de linguagem de definição de dados (DDL), como ALTER TABLE.

> **Note**
>
> A sintaxe e o comportamento do comando VACUUM no Amazon Redshift são substancialmente diferentes da operação VACUUM do PostgreSQL. Por exemplo, a operação VACUUM padrão no Amazon Redshift é VACUUM FULL. Ela recupera o espaço em disco e reclassifica todas as linhas. Por outro lado, a operação padrão VACUUM no PostgreSQL recupera somente o espaço e o disponibiliza para reutilização.

Para obter mais informações, consulte Vacuum de tabelas.

Privilégios obrigatórios

A seguir estão os privilégios obrigatórios para VACUUM:

- Superusuário
- Usuários com privilégio VACUUM
- Proprietário da tabela
- Proprietário do banco de dados com o qual a tabela é compartilhada

Sintaxe

```
VACUUM [ FULL | SORT ONLY | DELETE ONLY | REINDEX | RECLUSTER ]
[ [ table_name ] [ TO threshold PERCENT ] [ BOOST ] ]
```

Parâmetros

FULL

Classifica a tabela especificada (ou todas as tabelas no banco de dados atual) e recupera o espaço em disco ocupado por linhas que foram marcadas para exclusão pelas operações UPDATE e DELETE anteriores. VACUUM FULL é o valor padrão.

Uma limpeza total não executa uma reindexação de tabelas intercaladas. Para reindexar tabelas intercaladas acompanhadas por uma limpeza completa, use a opção VACUUM REINDEX.

Por padrão, VACUUM FULL ignora a fase de classificação de qualquer tabela que já esteja pelo menos 95% classificada. Se VACUUM puder ignorar a fase de classificação, o comando

executará DELETE ONLY e recuperará o espaço na fase de exclusão para que pelo menos 95% das linhas restantes não sejam marcadas para exclusão.

Se o limite de classificação não for alcançado (por exemplo, se 90% das linhas forem classificadas) e VACUUM executar uma classificação completa, o comando também executará uma operação de exclusão completa, recuperando espaço de 100% das linhas excluídas.

Você pode alterar o limite de limpeza padrão apenas para uma única tabela. Para alterar o limite de limpeza padrão para uma única tabela, inclua o nome da tabela e o parâmetro TO limite PERCENT.

SORT ONLY

Classifica a tabela especificada (ou todas as tabelas no banco de dados atual) sem retomar espaço liberado pelas linhas excluídas. Essa opção é útil quando a recuperação de espaço em disco não é importante, mas sim a reclassificação de novas linhas. Uma limpeza do tipo SORT ONLY reduz o tempo decorrido para operações de limpeza quando a região não classificada não contém um grande número de linhas excluídas e não se estende por toda a região classificada. Aplicativos que não têm restrições de espaço em disco mas dependem de otimizações de consulta associadas à manutenção de linhas classificadas na tabela podem se beneficiar desse tipo de limpeza.

Por padrão, VACUUM SORT ONLY ignora qualquer tabela que esteja pelo menos 95% classificada. Para alterar o limite de classificação padrão para uma única tabela, inclua o nome da tabela e o parâmetro TO limite PERCENT ao executar o comando VACUUM.

DELETE ONLY

O Amazon Redshift executa automaticamente uma limpeza DELETE ONLY em segundo plano. Assim, você dificilmente precisará executar uma limpeza DELETE ONLY.

VACUUM DELETE solicita de volta o espaço em disco ocupado por linhas que foram marcadas para exclusão pelas operações UPDATE e DELETE anteriores, e compacta a tabela para liberar o espaço ocupado. Uma operação de limpeza DELETE ONLY não classifica dados da tabela.

Essa opção reduz o tempo decorrido para operações de vacuum quando é importante recuperar espaço em disco, mas reclassificar novas linhas não é. Essa opção também pode ser útil quando a performance da sua consulta já é ótimo, e reclassificar linhas para otimizar a performance da consulta não é uma necessidade.

Por padrão, VACUUM DELETE ONLY recupera o espaço, de maneira que pelo menos 95% das linhas restantes não sejam marcadas para exclusão. Para alterar o limite de exclusão padrão

para uma única tabela, inclua o nome da tabela e o parâmetro TO limite PERCENT ao executar o comando VACUUM.

Algumas operações, como `ALTER TABLE APPEND`, podem fazer com que as tabelas sejam fragmentadas. Ao usar a cláusula `DELETE ONLY`, a operação de limpeza solicitará de volta o espaço das tabelas fragmentadas. O mesmo valor limite de 95% se aplica à operação de desfragmentação.

REINDEX

Analisa a distribuição dos valores nas colunas de chave de classificação intercaladas e executa uma operação VACUUM completa. Se REINDEX é usado, um nome de tabela é necessário.

VACUUM REINDEX é bem mais demorado do que VACUUM FULL porque inclui uma etapa adicional para analisar as chaves de classificação intercaladas. As operações de classificação e mesclagem podem levar mais tempo para tabelas intercaladas porque a classificação intercalada pode precisar reorganizar mais linhas do que uma classificação composta.

Se uma operação VACUUM REINDEX encerrar antes da conclusão, a próxima operação VACUUM retomará a reindexação antes de executar a operação de limpeza completa.

VACUUM REINDEX não é compatível com TO threshold PERCENT.

RECLUSTER

Classifica as partes da tabela que não são classificadas. Partes da tabela que já estão classificadas por classificação automática de tabela são deixadas intactas. Esse comando não mescla os dados recém-classificados com a região classificada. Ele também não recupera todo o espaço marcado para exclusão. Quando esse comando for concluído, a tabela pode não aparecer totalmente classificada, conforme indicado pelo campo `unsorted` em SVV_TABLE_INFO.

Recomendamos que você use VACUUM RECLUSTER para tabelas grandes com ingestão frequente e consultas que acessam apenas os dados mais recentes.

VACUUM RECLUSTER não é compatível com o limite TO PERCENT. Se RECLUSTER for usado, um nome de tabela será necessário.

O VACUUM RECLUSTER não é compatível com tabelas com chaves de classificação intercaladas e tabelas com estilo de distribuição ALL.

table_name

Nome de uma tabela a ser limpa. Se você não especificar um nome para a tabela, a operação de limpeza se aplicará a todas as tabelas no banco de dados atual. Você pode especificar qualquer tabela permanente ou temporária criada por usuário. O comando não é significativo para outros objetos, como exibições e tabelas de sistema.

Se você incluir o parâmetro TO limite PERCENT, será necessário um nome para a tabela.

TO limite PERCENT

Cláusula que especifica o limite acima do qual VACUUM ignora a fase de classificação e o limite de destino para recuperar espaço na fase de exclusão. O limite de classificação é a porcentagem de linhas do total que já está classificada para a tabela especificada antes da limpeza. O delete threshold é a porcentagem mínima do total de linhas não marcadas para exclusão após a limpeza.

Como o VACUUM reclassifica as linhas apenas quando a porcentagem de linhas classificadas em uma tabela é menor que o limite de classificação, o Amazon Redshift geralmente pode reduzir significativamente os tempos de VACUUM. De modo semelhante, quando VACUUM não é restrito a recuperar o espaço de 100% das linhas marcadas para exclusão, muitas vezes é capaz de regravar blocos que contêm somente algumas linhas excluídas.

Por exemplo, se você especificar 75 para o limite, VACUUM ignorará a fase de classificação se 75% ou mais das linhas da tabela já estiverem classificadas. Para a fase de exclusão, VACUUMS define um objetivo de recuperação de espaço em disco de forma que 75% das linhas da tabela não sejam marcadas para exclusão após a limpeza. O valor do limite deve ser um número inteiro entre 0 e 100. O padrão é 95. Se você especificar um valor de 100, VACUUM sempre classificará a tabela a menos que ela já esteja totalmente classificada e recuperará o espaço de todas as linhas marcadas para exclusão. Se você especificar um valor de 0, VACUUM nunca classificará a tabela e nunca recuperará espaço.

Se você incluir o parâmetro TO limite PERCENT, também deverá especificar um nome para a tabela. Se um nome para a tabela for omitido, VACUUM retorna uma falha.

Não é possível usar o parâmetro TO threshold PERCENT com REINDEX.

BOOST

Executa o comando VACUUM com recursos adicionais, como memória e espaço em disco, conforme estiverem disponíveis. Com a opção BOOST, VACUUM opera em uma janela e bloqueia exclusões e atualizações simultâneas durante a operação VACUUM. A execução com a

opção BOOST compete com recursos do sistema, o que pode afetar a performance da consulta. Execute o VACUUM BOOST quando a carga no sistema for leve, como durante operações de manutenção.

Considere o seguinte ao usar a opção BOOST.

- Quando BOOST é especificado, o valor de table-name é necessário.
- BOOST não é compatível com REINDEX.
- BOOST é ignorado com DELETE ONLY.

Observações de uso

Para a maioria das aplicações do Amazon Redshift, uma limpeza completa é recomendada. Para obter mais informações, consulte Vacuum de tabelas.

Para executar uma operação de limpeza, observe o seguinte comportamento:

- Não é possível executar o comando VACUUM em um bloco de transação (BEGIN ... END). Para obter mais informações sobre transações, consulte Níveis de isolamento no Amazon Redshift.
- As tabelas podem crescer quando são limpas. Esse comportamento é esperado quando não há linhas excluídas a serem recuperadas ou quando a nova ordem de classificação da tabela resulta em uma taxa mais baixa de compactação de dados.
- Durante operações de limpeza, qualquer grau de degradação de performance de consulta é esperado. A performance normal é retomada assim que a operação de limpeza é concluída.
- As operações de gravação simultâneas continuam durante as operações de limpeza, mas não recomendamos realizar operações de gravação durante a limpeza. É mais eficiente concluir operações de gravação antes de executar a limpeza. Além disso, todos os dados gravados após o início de uma operação de limpeza não podem ser eliminados por essa operação. Nesse caso, uma segunda operação de limpeza é necessária.
- Uma operação de limpeza pode não começar se uma operação de carregamento ou inserção já estiver em andamento. As operações de limpeza exigem temporariamente acesso exclusivo às tabelas para iniciar. Esse acesso exclusivo é necessário por pouco tempo para que as operações de limpeza não obstruam cargas e inserções simultâneas por qualquer período significativo.
- As operações de limpeza são ignoradas quando não há trabalho a fazer para uma tabela específica. No entanto, há algumas sobrecargas associadas à descoberta de que a operação pode ser ignorada. Se você sabe que uma tabela é limpa ou não atende ao limite de limpeza, não execute uma operação de limpeza nela.

- Uma operação de limpeza DELETE ONLY em uma tabela pequena pode não reduzir o número de blocos usados para armazenar dados, especialmente quando a tabela tem um grande número colunas ou o cluster usa um grande número fatias por nó. Essas operações de limpeza adicionam um bloco por coluna por fatia para contabilizar inserções simultâneas na tabela, e essa sobrecarga pode exceder a redução da contagem de blocos do espaço em disco recuperado. Por exemplo, se uma tabela de 10 colunas em um cluster de 8 nós ocupar 1000 blocos antes de uma limpeza, a limpeza não reduzirá a contagem de blocos real a não ser que mais de 80 blocos de espaço em disco sejam recuperados devido às linhas excluídas. (Cada bloco de dados usa 1 MB.)

As operações de limpeza automáticas pausarão caso alguma destas condições não forem atendidas:

- Um usuário executa uma operação de linguagem de definição de dados (DDL), como ALTER TABLE, que precisa de um bloqueio exclusivo na tabela em que a limpeza automática está atuando.
- Um período de carga elevada no cluster.

Compatibilidade com VACUUM simultâneo

O Amazon Redshift comporta a execução simultânea de várias transações de vacuum em diferentes sessões em um cluster ou um grupo de trabalho. Isso significa que é possível emitir instâncias diferentes e múltiplas de todos os modos de vacuum de uma só vez, com cada transação de vacuum em uma tabela exclusiva. Duas operações de vacuum não podem trabalhar em uma única tabela ao mesmo tempo.

Diretrizes para execução simultânea de vacuum

- Ao executar transações de vacuum simultâneas em sessões diferentes, você deve monitorar os recursos do sistema e evitar executar muitas operações de vacuum simultaneamente.
- O nível de simultaneidade recomendado depende da quantidade de espaço a ser recuperado, do número e da largura das linhas a serem classificadas, do tamanho do data warehouse e do tamanho da workload executada com as operações VACUUM.
- Dependendo do modo da transação de vacuum, comece com duas operações de vacuum simultâneas e adicione mais, dependendo do tempo de execução e da carga do sistema. Assim como outras consultas pesadas emitidas por usuários, as operações de vacuum poderão começar a ser enfileiradas se você executar muitas simultaneamente quando o Amazon Redshift atingir os limites de recurso do sistema.

- Execute várias operações de BOOST Vacuum com cuidado. A execução de Vacuum com a opção BOOST compete com recursos do sistema, o que pode afetar a performance da consulta. Execute o VACUUM BOOST quando a carga no sistema for leve, como durante operações de manutenção.
- Se você não especificar um nome para a tabela, a operação de limpeza se aplicará a todas as tabelas no banco de dados atual. Essas operações de vacuum ainda são executadas sequencialmente.

Exemplos

Recupere o espaço e o banco de dados e reclassifique as linhas em todas as tabelas com base no limite padrão de 95% de vacuum.

```
vacuum;
```

Recupere espaço e reclassifique as linhas na tabela SALES com base no limite padrão de 95%.

```
vacuum sales;
```

Sempre recupere espaço e reclassifique as linhas na tabela SALES.

```
vacuum sales to 100 percent;
```

Reclassifique as linhas na tabela SALES somente se menos que 75% das linhas já estiverem organizadas.

```
 vacuum sort only sales to 75 percent;
```

Recupere espaço na tabela SALES de maneira que pelo menos 75% das linhas restantes não sejam marcadas para exclusão depois da limpeza.

```
vacuum delete only sales to 75 percent;
```

Reindexe e limpe a tabela LISTING.

```
vacuum reindex listing;
```

O comando a seguir retorna um erro.

```
vacuum reindex listing to 75 percent;
```

Reagrupe e limpe a tabela LISTING.

```
vacuum recluster listing;
```

Reagrupe e limpe a tabela LISTING com a opção BOOST.

```
vacuum recluster listing boost;
```

Referência de funções SQL

Tópicos

- Função de apenas nó líder
- Funções agregadas
- Funções de array
- Funções agregadas bit-wise
- Expressões condicionais
- Funções de formatação de tipo de dados
- Perfis de data e hora
- Funções de hash
- Funções HyperLogLog
- Funções JSON
- Funções de machine learning
- Funções matemáticas
- Funções de objetos
- Funções espaciais
- Funções de string
- Funções de informação de tipo SUPER
- Funções e operadores VARBYTE

- Funções de janela
- Funções de administração do sistema
- Funções de informação do sistema

O Amazon Redshift é compatível com diversas funções que são extensões do padrão SQL, bem como funções agregadas padrão, funções escalares e funções da janela.

> **Note**
>
> O Amazon Redshift é baseado no PostgreSQL. O Amazon Redshift e o PostgreSQL têm uma série de diferenças muito importantes que você deve conhecer ao projetar e desenvolver suas aplicações de data warehouse. Para mais informações sobre como o SQL do Amazon Redshift difere de PostgreSQL, consulte Amazon Redshift e PostgreSQL.

Função de apenas nó líder

Algumas consultas do Amazon Redshift são distribuídas e executadas em nós de computação; outras consultas são executadas exclusivamente no nó líder.

O nó de liderança distribui a SQL aos nós de computação quando uma consulta se refere a tabelas criadas pelo usuário ou tabelas de sistema (tabelas com um prefixo STL ou STV e exibições de sistema com um prefixo SVL ou SVV). Uma consulta que se refere apenas a tabelas do catálogo (tabelas com um prefixo PG, tal como PG_TABLE_DEF), ou que não se refere a qualquer tabela, é executada exclusivamente no nó de liderança.

Algumas funções do SQL do Amazon Redshift são aceitas apenas no nó líder e não são suportadas nos nós de computação. Uma consulta que usa uma função de nó líder deve ser executada exclusivamente no nó líder, não nos nós de computação, ou retornará um erro.

A documentação para cada função de apenas nó líder inclui uma observação indicando que a função retornará um erro se ela se referir a tabelas definidas por usuários ou tabelas de sistema do Amazon Redshift.

Para obter mais informações, consulte Funções SQL compatíveis no nó de liderança.

As seguintes funções SQL são funções somente do nó de liderança e não são compatíveis com os nós de computação:

Funções de informação do sistema

- CURRENT_SCHEMA
- CURRENT_SCHEMAS
- HAS_DATABASE_PRIVILEGE
- HAS_SCHEMA_PRIVILEGE
- HAS_TABLE_PRIVILEGE

Funções de string

- SUBSTR

Funções matemáticas

- FACTORIAL
- LOG

As seguintes funções somente do nó de liderança são obsoletas e não são mais compatível:

Funções de data

- AGE
- CURRENT_TIME
- CURRENT_TIMESTAMP
- LOCALTIME
- ISFINITE
- NOW

Funções de string

- GETBIT
- GET_BYTE
- SET_BIT
- SET_BYTE

- TO_ASCII

Funções agregadas

Tópicos

- Função ANY_VALUE
- Função APPROXIMATE PERCENTILE_DISC
- Função AVG
- Função COUNT
- Função LISTAGG
- Função MAX
- Função MEDIAN
- Função MIN
- Função PERCENTILE_CONT
- Funções STDDEV_SAMP e STDDEV_POP
- Função SUM
- Funções VAR_SAMP e VAR_POP

Funções agregadas computam um único valor de resultado a partir de um conjunto de valores de entrada.

Instruções SELECT usando funções agregadas podem incluir duas cláusulas opcionais: GROUP BY e HAVING. A sintaxe para essas cláusulas é seguinte (usando a função COUNT como um exemplo):

```
SELECT count (*) expression FROM table_reference
WHERE condition [GROUP BY expression ] [ HAVING condition]
```

A cláusula GROUP BY agrega e agrupa os resultados pelos valores exclusivos em uma coluna ou colunas especificada(s). A cláusula HAVING restringe os resultados obtidos para linhas onde determinada condição de agregação é verdadeira, tal como uma contagem (*) > 1. A cláusula HAVING é usada na mesma forma que WHERE para restringir as linhas com base no valor de uma coluna. Para obter um exemplo dessas cláusulas adicionais, consulte COUNT.

Funções agregadas não aceitam funções agregadas aninhadas ou funções da janela como argumentos.

Função ANY_VALUE

A função ANY_VALUE retorna qualquer valor dos valores de expressão de entrada não deterministicamente. Essa função retornará NULL se a expressão de entrada não resultar no retorno de nenhuma linha. A função também poderá retornar NULL se houver valores NULL na expressão de entrada. Se a entrada contiver valores NULL misturados com valores não NULL, NULL poderá ser retornado. Se todos os valores forem NULL, NULL será retornado. Se nenhuma linha corresponder à condição, NULL será retornado.

Sintaxe

```
ANY_VALUE( [ DISTINCT | ALL ] expression )
```

Argumentos

DISTINCT | ALL

Especifique DISTINCT ou ALL para retornar qualquer valor dos valores de expressão de entrada. O argumento DISTINCT não tem efeito e é ignorado.

expressão

A coluna ou expressão de destino na qual a função opera. A expressão é um destes tipos de dados:

- SMALLINT
- INTEGER
- BIGINT
- DECIMAL
- REAL
- DOUBLE PRECISION
- BOOLEAN
- CHAR
- VARCHAR
- DATE
- TIMESTAMP

- TIMESTAMPTZ
- TIME
- TIMETZ
- INTERVALO ENTRE UM ANO E UM MÊS
- INTERVALO ENTRE UM DIA E UM SEGUNDO
- VARBYTE
- SUPER
- HLLSKETCH
- GEOMETRY
- GEOGRAPHY

Retornos

Retorna o mesmo tipo de dados da expressão.

Observações de uso

Se uma instrução que especifica a função ANY_VALUE para uma coluna também incluir uma segunda referência de coluna, a segunda coluna deve aparecer em uma cláusula GROUP BY ou ser incluída em uma função agregada.

Exemplos

Os exemplos usam a tabela de eventos que é criada na Etapa 4: Carregar dados do Amazon S3 para o Amazon Redshift no Guia de conceitos básicos do Amazon Redshift. O exemplo a seguir retorna uma instância de qualquer dateid onde o eventname é Eagles.

```
select any_value(dateid) as dateid, eventname from event where eventname ='Eagles'
 group by eventname;
```

A seguir estão os resultados.

```
dateid | eventname
-------+---------------
 1878  | Eagles
```

O exemplo a seguir retorna uma instância de qualquer dateid em que o eventname é Eagles ou Cold War Kids.

```
select any_value(dateid) as dateid, eventname from event where eventname in('Eagles',
 'Cold War Kids') group by eventname;
```

A seguir estão os resultados.

```
dateid | eventname
-------+---------------
 1922  | Cold War Kids
 1878  | Eagles
```

Função APPROXIMATE PERCENTILE_DISC

APPROXIMATE PERCENTILE_DISC é uma função de distribuição inversa que assume um modelo de distribuição discreta. Ela pega um valor percentil e uma especificação de classificação e retorna um elemento do conjunto fornecido. A aproximação permite que a função seja executada muito mais rápido, com um baixo erro relativo de cerca de 0,5%.

Para dado valor percentil APPROXIMATE PERCENTILE_DISC usa um algoritmo de resumo quantílico para aproximação do percentil discreto da expressão na cláusula ORDER BY. APPROXIMATE PERCENTILE_DISC retorna o valor com o menor valor de distribuição cumulativa (em relação à mesma especificação de classificação) que é maior ou igual ao percentil.

Sintaxe

```
APPROXIMATE  PERCENTILE_DISC ( percentile )
WITHIN GROUP (ORDER BY expr)
```

Argumentos

percentil

Constante numérica entre 0 e 1. Nulls são ignorados no cálculo.

WITHIN GROUP (ORDER BY expr)

Cláusula que especifica valores numéricos ou de data/hora para classificação e computação do percentil.

Retornos

O mesmo tipo de dados que a expressão ORDER BY na cláusula WITHIN GROUP.

Observações de uso

Se a instrução APPROXIMATE PERCENTILE_DISC incluir uma cláusula GROUP BY, o conjunto de resultados é limitado. O limite varia com base no tipo de nó e no número de nós. Se o limite for excedido, a função falha e retorna o seguintes erro.

```
GROUP BY limit for approximate percentile_disc exceeded.
```

Se você precisar avaliar mais grupos do que o limite permite, considere usar Função PERCENTILE_CONT.

Exemplos

O seguinte exemplo retorna o número de vendas, vendas globais e o quinquagésimo valor percentil para as primeiras 10 datas.

```
select top 10 date.caldate,
count(totalprice), sum(totalprice),
approximate percentile_disc(0.5)
within group (order by totalprice)
from listing
join date on listing.dateid = date.dateid
group by date.caldate
order by 3 desc;

caldate    | count | sum        | percentile_disc
-----------+-------+------------+----------------
2008-01-07 |   658 | 2081400.00 |         2020.00
2008-01-02 |   614 | 2064840.00 |         2178.00
2008-07-22 |   593 | 1994256.00 |         2214.00
2008-01-26 |   595 | 1993188.00 |         2272.00
2008-02-24 |   655 | 1975345.00 |         2070.00
2008-02-04 |   616 | 1972491.00 |         1995.00
2008-02-14 |   628 | 1971759.00 |         2184.00
2008-09-01 |   600 | 1944976.00 |         2100.00
2008-07-29 |   597 | 1944488.00 |         2106.00
2008-07-23 |   592 | 1943265.00 |         1974.00
```

Função AVG

A função AVG retorna a média (meio aritmético) dos valores de expressão de entrada. A função AVG funciona com valores numéricos e ignora valores NULL.

Sintaxe

```
AVG ( [ DISTINCT | ALL ] expression )
```

Argumentos

expressão

A coluna ou expressão de destino na qual a função opera. A expressão é um destes tipos de dados:

- SMALLINT
- INTEGER
- BIGINT
- NUMERIC
- DECIMAL
- REAL
- DOUBLE PRECISION
- SUPER

DISTINCT | ALL

Com o argumento DISTINCT, a função elimina todos os valores duplicados da expressão especificada antes de calcular a média. Com o argumento ALL, a função retém todos os valores duplicados da expressão especificada para calcular a média. ALL é o padrão.

Tipos de dados

Os tipos de argumento compatíveis com a função AVG são SMALLINT, INTEGER, BIGINT, NUMERIC, DECIMAL, REAL, DOUBLE PRECISION e SUPER.

Os tipos de retorno compatíveis com a função AVG são:

- BIGINT para qualquer tipo argumento de inteiro

- DOUBLE PRECISION para um argumento de ponto flutuante
- Retorna o mesmo tipo de dados da expressão para qualquer outro tipo de argumento.

A precisão padrão para o resultado de uma função AVG com um argumento NUMERIC ou DECIMAL é 38. A escala do resultado é a mesma que a escala do argumento. Por exemplo, uma AVG de uma coluna DEC(5,2) retorna um tipo de dado DEC(38,2).

Exemplos

Encontre a quantidade média vendida por transação na tabela SALES:

```
select avg(qtysold)from sales;

avg
-----
2
(1 row)
```

Encontre o preço total médio listado para todas as ofertas:

```
select avg(numtickets*priceperticket) as avg_total_price from listing;

avg_total_price
-----------------
3034.41
(1 row)
```

Encontre o preço médio pago, agrupado por mês em ordem descendente:

```
select avg(pricepaid) as avg_price, month
from sales, date
where sales.dateid = date.dateid
group by month
order by avg_price desc;

avg_price | month
-----------+-------
659.34 | MAR
655.06 | APR
645.82 | JAN
643.10 | MAY
```

```
642.72 | JUN
642.37 | SEP
640.72 | OCT
640.57 | DEC
635.34 | JUL
635.24 | FEB
634.24 | NOV
632.78 | AUG
(12 rows)
```

Função COUNT

A função COUNT conta as linhas definidas pela expressão.

A função COUNT tem as variações a seguir.

- COUNT (*) conta todas as linhas na tabela de destino independente se elas contêm nulls ou não.
- COUNT (expressão) computa o número de linhas com valores não NULL em uma coluna ou expressão específica.
- COUNT (expressão DISTINCT) computa o número de valores distintos não NULL em uma coluna ou expressão.
- APPROXIMATE COUNT DISTINCT aproxima o número de valores distintos não NULL em uma coluna ou expressão.

Sintaxe

```
COUNT( * | expression )
```

```
COUNT ( [ DISTINCT | ALL ] expression )
```

```
APPROXIMATE COUNT ( DISTINCT expression )
```

Argumentos

expressão

A coluna ou expressão de destino na qual a função opera. A função COUNT é compatível com todos os tipos de dados de argumento.

DISTINCT | ALL

Com o argumento DISTINCT, a função elimina todos os valores duplicados da expressão especificada antes realizar a contagem. Com o argumento ALL, a função retém todos os valores duplicados da expressão para contagem. ALL é o padrão.

APPROXIMATE

Quando usada com APPROXIMATE, uma função COUNT DISTINCT usa um algoritmo de HyperLogLog para aproximar o número de valores distintos não NULL em uma coluna ou expressão. A consultas que usam a palavra-chave APPROXIMATE são executadas muito mais rápido, com um baixo erro relativo de cerca de 2%. A aproximação é justificada para consultas que retornam um grande número de valores distintos, com milhões ou mais por consulta ou grupo, se houver uma cláusula group by. Para conjuntos menores de valores distintos, na casa dos milhares, a aproximação pode ser mais lenta do que uma contagem precisa. APPROXIMATE pode ser usada somente com COUNT DISTINCT.

Tipo de retorno

A função COUNT retorna BIGINT.

Exemplos

Conte todos os usuários do estado da Flórida:

```
select count(*) from users where state='FL';

count
-------
510
```

Conte todos os nomes de eventos da tabela EVENT:

```
select count(eventname) from event;

count
-------
8798
```

Conte todos os nomes de eventos da tabela EVENT:

```
select count(all eventname) from event;

count
-------
8798
```

Conte todos os IDs exclusivos dos locais de evento da tabela EVENT:

```
select count(distinct venueid) as venues from event;

venues
--------
204
```

Conte o número de vezes que cada vendedor listou lotes de um ou mais ingressos para venda. Agrupe os resultados por ID de vendedor:

```
select count(*), sellerid from listing
where numtickets > 4
group by sellerid
order by 1 desc, 2;

count | sellerid
------+----------
12    |    6386
11    |    17304
11    |    20123
11    |    25428
...
```

Os seguintes exemplos comparam os valores de retorno e os tempos de execução para COUNT e APPROXIMATE COUNT.

```
select  count(distinct pricepaid) from sales;

count
-------
  4528


Time: 48.048 ms
```

```
select approximate count(distinct pricepaid) from sales;

count
-------
  4553


Time: 21.728 ms
```

Função LISTAGG

Para cada grupo em uma consulta, a função de agregação LISTAGG ordena as linhas desse grupo de acordo com a expressão ORDER BY e, depois, concatena os valores em uma única string.

Sintaxe

```
LISTAGG( [DISTINCT] aggregate_expression [, 'delimiter' ] )
[ WITHIN GROUP (ORDER BY order_list) ]
```

Argumentos

DISTINCT

Uma cláusula que elimina valores duplicados da expressão especificada antes de concatenar. Os espaços à direita são ignorados. Por exemplo, as strings `'a'` e `'a '` são tratadas como duplicatas. LISTAGG usa o primeiro valor encontrado. Para obter mais informações, consulte Significância de espaços em branco.

aggregate_expression

Qualquer expressão válida, como um nome de coluna, que forneça os valores para agregar. Valores NULL e strings vazias são ignoradas.

delimitador

A constante de string que separará os valores concatenados. O padrão é NULL.

WITHIN GROUP (ORDER BY order_list)

Uma cláusula que especifica a ordem de classificação dos valores agregados.

Retornos

VARCHAR(MAX). Se o resultado for maior que o tamanho máximo de VARCHAR, LISTAGG retornará o seguinte erro:

```
Invalid operation: Result size exceeds LISTAGG limit
```

Observações de uso

- Se uma instrução inclui várias funções LISTAGG que usam cláusulas WITHIN GROUP, todas as cláusulas WITHIN GROUP devem usar os mesmos valores ORDER BY.

  Por exemplo, a seguinte instrução retorna um erro.

  ```
  SELECT LISTAGG(sellerid)
  WITHIN GROUP (ORDER BY dateid) AS sellers,
  LISTAGG(dateid)
  WITHIN GROUP (ORDER BY sellerid) AS dates
  FROM sales;
  ```

  As instruções a seguir são executadas com êxito.

  ```
  SELECT LISTAGG(sellerid)
  WITHIN GROUP (ORDER BY dateid) AS sellers,
  LISTAGG(dateid)
  WITHIN GROUP (ORDER BY dateid) AS dates
  FROM sales;

  SELECT LISTAGG(sellerid)
  WITHIN GROUP (ORDER BY dateid) AS sellers,
  LISTAGG(dateid) AS dates
  FROM sales;
  ```

- Não é possível usar as funções agregadas LISTAGG, PERCENTILE_CONT e MEDIAN com outras funções agregadas distintas.

Exemplos

O seguinte exemplo agrega os IDs de vendedor, ordenados por ID de vendedor.

```
SELECT LISTAGG(sellerid, ', ')
```

```
WITHIN GROUP (ORDER BY sellerid)
FROM sales
WHERE eventid = 4337;

listagg

-----------------------------------------------------------------------------------------------
380, 380, 1178, 1178, 1178, 2731, 8117, 12905, 32043, 32043, 32043, 32432, 32432,
 38669, 38750, 41498, 45676, 46324, 47188, 47188, 48294
```

O exemplo a seguir usa DISTINCT para retornar uma lista de IDs de vendedor exclusivos.

```
SELECT LISTAGG(DISTINCT sellerid, ', ')
WITHIN GROUP (ORDER BY sellerid)
FROM sales
WHERE eventid = 4337;

listagg

-----------------------------------------------------------------------------------------------
380, 1178, 2731, 8117, 12905, 32043, 32432, 38669, 38750, 41498, 45676, 46324, 47188,
 48294
```

O seguinte exemplo agrega os IDs de vendedor por ordem de data.

```
SELECT LISTAGG(sellerid, ', ')
WITHIN GROUP (ORDER BY dateid)
FROM sales
WHERE eventid = 4337;

   listagg
-----------------------------------------------------------------------------------------------
 41498, 47188, 47188, 1178, 1178, 1178, 380, 45676, 46324, 48294, 32043, 32043, 32432,
 12905, 8117, 38750, 2731, 32432, 32043, 380, 38669
```

O exemplo a seguir retorna uma lista separada por pipes das datas de vendas do comprador com ID 660.

```
SELECT LISTAGG(
    (SELECT caldate FROM date WHERE date.dateid=sales.dateid), ' | '
)
WITHIN GROUP (ORDER BY sellerid DESC, salesid ASC)
```

```
FROM sales
WHERE buyerid = 660;

             listagg
-------------------------------------------------
2008-07-16 | 2008-07-09 | 2008-01-01 | 2008-10-26
```

O exemplo a seguir retorna uma lista separada por vírgulas dos IDs de venda dos compradores com ID 660, 661 e 662.

```
SELECT buyerid,
LISTAGG(salesid,', ')
WITHIN GROUP (ORDER BY salesid) AS sales_id
FROM sales
WHERE buyerid BETWEEN 660 AND 662
GROUP BY buyerid
ORDER BY buyerid;

buyerid |                sales_id
--------+-------------------------------------------------------
660     | 32872, 33095, 33514, 34548
661     | 19951, 20517, 21695, 21931
662     | 3318, 3823, 4215, 51980, 53202, 55908, 57832, 171603
```

Função MAX

A função MAX retorna o valor máximo em um conjunto de linhas. DISTINCT ou ALL podem ser usadas, mas não afetam os resultados.

Sintaxe

```
MAX ( [ DISTINCT | ALL ] expression )
```

Argumentos

expressão

A coluna ou expressão de destino na qual a função opera. A expressão é um destes tipos de dados:

- SMALLINT
- INTEGER

- BIGINT
- DECIMAL
- REAL
- DOUBLE PRECISION
- CHAR
- VARCHAR
- DATE
- TIMESTAMP
- TIMESTAMPTZ
- TIME
- TIMETZ
- VARBYTE
- SUPER

DISTINCT | ALL

Com o argumento DISTINCT, a função elimina todos os valores duplicados da expressão especificada antes de calcular o máximo. Com o argumento ALL, a função retém todos os valores duplicados da expressão para calcular o máximo. ALL é o padrão.

Tipos de dados

Retorna o mesmo tipo de dados da expressão. O equivalente booleano da função MIN é Função BOOL_AND e o equivalente booleano de MAX é Função BOOL_OR.

Exemplos

Encontre o preço mais alto pago de todas as vendas:

```
select max(pricepaid) from sales;

max
----------
12624.00
(1 row)
```

Encontre o preço mais alto pago por ingresso em todas as vendas:

```
select max(pricepaid/qtysold) as max_ticket_price
from sales;

max_ticket_price
-----------------
2500.00000000
(1 row)
```

Função MEDIAN

Calcula o valor médio para o intervalo de valores. Os valores NULL no intervalo são ignorados.

MEDIAN é uma função de distribuição inversa que assume um modelo de distribuição contínua.

MEDIAN é um caso especial de PERCENTILE_CONT.

Sintaxe

```
MEDIAN(median_expression)
```

Argumentos

median_expression

A coluna ou expressão de destino na qual a função opera.

Tipos de dados

O tipo de retorno é determinado pelo tipo de dados de median_expression. A tabela a seguir mostra o tipo de retorno para cada tipo de dados de median_expression.

Tipo de entrada	Tipo de retorno
INT2, INT4, INT8, NUMERIC, DECIMAL	DECIMAL
FLOAT, DOUBLE	DOUBLE
DATE	DATE
TIMESTAMP	TIMESTAMP

Tipo de entrada	Tipo de retorno
TIMESTAMPTZ	TIMESTAMPTZ

Observações de uso

Se o argumento de median_expression é um tipo de dados DECIMAL com a precisão máxima de 38 dígitos, é possível que MEDIAN retorne um resultado impreciso ou um erro. Se o valor de retorno da função MEDIAN excede 38 dígitos, o resultado é truncado, o que causa a perda de precisão. Se, durante a interpolação, um resultado intermediário excede a precisão máxima, um excedente numérico ocorre e função retorna um erro. Para evitar essas condições, recomendamos o uso de um tipo de dados com menor precisão ou a conversão do argumento median_expression para uma precisão mais baixa.

Se uma instrução inclui várias chamadas para funções agregadas baseadas em classificação (LISTAGG, PERCENTILE_CONT ou MEDIAN), todas devem usar os mesmos valores ORDER BY. Observe que MEDIAN aplica um order by implícito no valor da expressão.

Por exemplo, a seguinte instrução retorna um erro.

```
SELECT TOP 10 salesid, SUM(pricepaid),
PERCENTILE_CONT(0.6) WITHIN GROUP(ORDER BY salesid),
MEDIAN(pricepaid)
FROM sales
GROUP BY salesid, pricepaid;

An error occurred when executing the SQL command:
SELECT TOP 10 salesid, SUM(pricepaid),
PERCENTILE_CONT(0.6) WITHIN GROUP(ORDER BY salesid),
MEDIAN(pricepaid)
FROM sales
GROUP BY salesid, pricepaid;

ERROR: within group ORDER BY clauses for aggregate functions must be the same
```

A instrução a seguir é executada com êxito.

```
SELECT TOP 10 salesid, SUM(pricepaid),
PERCENTILE_CONT(0.6) WITHIN GROUP(ORDER BY salesid),
MEDIAN(salesid)
```

```
FROM sales
GROUP BY salesid, pricepaid;
```

Exemplos

Os exemplos a seguir usam o banco de dados de exemplo de TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

O seguinte exemplo mostra que MEDIAN produz os mesmos resultados que PERCENTILE_CONT(0,5).

```
SELECT TOP 10 DISTINCT sellerid, qtysold,
PERCENTILE_CONT(0.5) WITHIN GROUP(ORDER BY qtysold),
MEDIAN(qtysold)
FROM sales
GROUP BY sellerid, qtysold;

+----------+---------+-----------------+--------+
| sellerid | qtysold | percentile_cont | median |
+----------+---------+-----------------+--------+
|        2 |       2 |               2 |      2 |
|       26 |       1 |               1 |      1 |
|       33 |       1 |               1 |      1 |
|       38 |       1 |               1 |      1 |
|       43 |       1 |               1 |      1 |
|       48 |       2 |               2 |      2 |
|       48 |       3 |               3 |      3 |
|       77 |       4 |               4 |      4 |
|       85 |       4 |               4 |      4 |
|       95 |       2 |               2 |      2 |
+----------+---------+-----------------+--------+
```

O exemplo a seguir encontra a quantidade média vendida para cada sellerid.

```
SELECT sellerid,
MEDIAN(qtysold)
FROM sales
GROUP BY sellerid
ORDER BY sellerid
LIMIT 10;

+----------+--------+
| sellerid | median |
```

```
+----------+--------+
|        1 |    1.5 |
|        2 |      2 |
|        3 |      2 |
|        4 |      2 |
|        5 |      1 |
|        6 |      1 |
|        7 |    1.5 |
|        8 |      1 |
|        9 |      4 |
|       12 |      2 |
+----------+--------+
```

Para verificar os resultados da consulta anterior para o primeiro sellerid, use o exemplo a seguir.

```
SELECT qtysold
FROM sales
WHERE sellerid=1;

+---------+
| qtysold |
+---------+
|       2 |
|       1 |
+---------+
```

Função MIN

A função MIN retorna o valor mínimo em um conjunto de linhas. DISTINCT ou ALL podem ser usadas, mas não afetam os resultados.

Sintaxe

```
MIN ( [ DISTINCT | ALL ] expression )
```

Argumentos

expressão

A coluna ou expressão de destino na qual a função opera. A expressão é um destes tipos de dados:

- SMALLINT

- INTEGER
- BIGINT
- DECIMAL
- REAL
- DOUBLE PRECISION
- CHAR
- VARCHAR
- DATE
- TIMESTAMP
- TIMESTAMPTZ
- TIME
- TIMETZ
- VARBYTE
- SUPER

DISTINCT | ALL

Com o argumento DISTINCT, a função elimina todos os valores duplicados da expressão especificada antes de calcular o mínimo. Com o argumento ALL, a função retém todos os valores duplicados da expressão para calcular o mínimo. ALL é o padrão.

Tipos de dados

Retorna o mesmo tipo de dados da expressão. O equivalente booleano de função MIN é o Função BOOL_AND e o equivalente booleano de MAX é o Função BOOL_OR.

Exemplos

Encontre o preço mais baixo pago de todas as vendas:

```
select min(pricepaid) from sales;

min
-------
20.00
(1 row)
```

Encontre o preço mais baixo pago por ingresso em todas as vendas:

```
select min(pricepaid/qtysold)as min_ticket_price
from sales;

min_ticket_price
------------------
20.00000000
(1 row)
```

Função PERCENTILE_CONT

PERCENTILE_CONT é uma função de distribuição inversa que assume um modelo de distribuição contínua. Ela pega um valor percentil e uma especificação de classificação e retorna um valor intercalar que cairia dentro do valor percentil fornecido em relação à especificação de classificação.

PERCENTILE_CONT computa uma interpolação linear entre valores após ordená-los. Usando o valor percentil `(P)` e o número de linhas não nulas `(N)` no grupo de agregação, a função computa o número da linha após ordenar as linhas de acordo com a especificação de classificação. Esse número de linha `(RN)` é computado de acordo com a fórmula `RN = (1+ (P*(N-1))`. O resultado final da função agregada é computado por interpolação linear entre os valores das linhas nos números de linha `CRN = CEILING(RN)` e `FRN = FLOOR(RN)`.

O resultado final será o seguinte.

Se `(CRN = FRN = RN)`, o resultado é `(value of expression from row at RN)`

Caso contrário, o resultado é o seguinte:

`(CRN - RN) * (value of expression for row at FRN) + (RN - FRN) * (value of expression for row at CRN)`.

Sintaxe

```
PERCENTILE_CONT(percentile)
WITHIN GROUP(ORDER BY expr)
```

Argumentos

percentil

Constante numérica entre 0 e 1. Os valores NULL são ignorados no cálculo.

expr

Especifica valores numéricos ou de data/hora para classificação e computação do percentil.

Retornos

O tipo de retorno é determinado pelo tipo de dados da expressão ORDER BY na cláusula WITHIN GROUP. A tabela a seguir mostra o tipo de retorno para cada tipo de dados da expressão ORDER BY.

Tipo de entrada	Tipo de retorno
INT2, INT4, INT8, NUMERIC, DECIMAL	DECIMAL
FLOAT, DOUBLE	DOUBLE
DATE	DATE
TIMESTAMP	TIMESTAMP
TIMESTAMPTZ	TIMESTAMPTZ

Observações de uso

Se a expressão ORDER BY é um tipo de dados DECIMAL com a precisão máxima de 38 dígitos, é possível que PERCENTILE_CONT retorne um resultado impreciso ou um erro. Se o valor de retorno da função PERCENTILE_CONT excede 38 dígitos, o resultado é truncado, o que causa a perda de precisão. Se, durante a interpolação, um resultado intermediário excede a precisão máxima, um excedente numérico ocorre e função retorna um erro. Para evitar essas condições, recomendamos o uso de um tipo de dados com menor precisão ou a conversão da expressão ORDER BY para uma precisão mais baixa.

Se uma instrução inclui várias chamadas para funções agregadas baseadas em classificação (LISTAGG, PERCENTILE_CONT ou MEDIAN), todas devem usar os mesmos valores ORDER BY. Observe que MEDIAN aplica um order by implícito no valor da expressão.

Por exemplo, a seguinte instrução retorna um erro.

```
SELECT TOP 10 salesid, SUM(pricepaid),
```

```
PERCENTILE_CONT(0.6) WITHIN GROUP(ORDER BY salesid),
MEDIAN(pricepaid)
FROM sales
GROUP BY salesid, pricepaid;

An error occurred when executing the SQL command:
SELECT TOP 10 salesid, SUM(pricepaid),
PERCENTILE_CONT(0.6) WITHIN GROUP(ORDER BY salesid),
MEDIAN(pricepaid)
FROM sales
GROUP BY salesid, pricepaid;

ERROR: within group ORDER BY clauses for aggregate functions must be the same
```

A instrução a seguir é executada com êxito.

```
SELECT TOP 10 salesid, SUM(pricepaid),
PERCENTILE_CONT(0.6) WITHIN GROUP(ORDER BY salesid),
MEDIAN(salesid)
FROM sales
GROUP BY salesid, pricepaid;
```

Exemplos

Os exemplos a seguir usam o banco de dados de exemplo de TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

O seguinte exemplo mostra que PERCENTILE_CONT(0.5) produz os mesmos resultados que MEDIAN.

```
SELECT TOP 10 DISTINCT sellerid, qtysold,
PERCENTILE_CONT(0.5) WITHIN GROUP(ORDER BY qtysold),
MEDIAN(qtysold)
FROM sales
GROUP BY sellerid, qtysold;

+----------+---------+-----------------+--------+
| sellerid | qtysold | percentile_cont | median |
+----------+---------+-----------------+--------+
|        2 |       2 |               2 |      2 |
|       26 |       1 |               1 |      1 |
|       33 |       1 |               1 |      1 |
```

```
|       38 |       1 |                1 |      1 |
|       43 |       1 |                1 |      1 |
|       48 |       2 |                2 |      2 |
|       48 |       3 |                3 |      3 |
|       77 |       4 |                4 |      4 |
|       85 |       4 |                4 |      4 |
|       95 |       2 |                2 |      2 |
+----------+---------+------------------+--------+
```

O exemplo a seguir encontra PERCENTILE_CONT(0.5) e PERCENTILE_CONT(0.75) da quantidade vendida para cada sellerid na tabela SALES.

```
SELECT sellerid,
PERCENTILE_CONT(0.5) WITHIN GROUP(ORDER BY qtysold) as pct_50,
PERCENTILE_CONT(0.75) WITHIN GROUP(ORDER BY qtysold) as pct_75
FROM sales
GROUP BY sellerid
ORDER BY sellerid
LIMIT 10;

+----------+--------+---------+
| sellerid | pct_50 | pct_75 |
+----------+--------+---------+
|        1 |    1.5 |    1.75 |
|        2 |      2 |    2.25 |
|        3 |      2 |       3 |
|        4 |      2 |       2 |
|        5 |      1 |     1.5 |
|        6 |      1 |       1 |
|        7 |    1.5 |    1.75 |
|        8 |      1 |       1 |
|        9 |      4 |       4 |
|       12 |      2 |    3.25 |
+----------+--------+---------+
```

Para verificar os resultados da consulta anterior para o primeiro sellerid, use o exemplo a seguir.

```
SELECT qtysold
FROM sales
WHERE sellerid=1;

+---------+
| qtysold |
```

```
+---------+
|       2 |
|       1 |
+---------+
```

Funções STDDEV_SAMP e STDDEV_POP

As funções STDDEV_SAMP e STDDEV_POP retornam o desvio padrão da amostra e da população de um conjunto de valores numéricos (número inteiro, decimal ou ponto flutuante). O resultado da função STDDEV_SAMP é equivalente à raiz quadrada da variação de amostra do mesmo conjunto de valores.

STDDEV_SAMP e STDDEV são sinônimos para a mesma função.

Sintaxe

```
STDDEV_SAMP | STDDEV ( [ DISTINCT | ALL ] expression)
STDDEV_POP ( [ DISTINCT | ALL ] expression)
```

A expressão deve ter um tipo de dados de número inteiro, decimal ou ponto flutuante. Independente do tipo de dados da expressão, o tipo de retorno desta função é um número de precisão dupla.

> **Note**
>
> O desvio padrão é calculado utilizando a aritmética de ponto flutuante, que pode resultar em uma ligeira imprecisão.

Observações de uso

Quando o desvio padrão de amostra (STDDEV ou STDDEV_SAMP) é calculado para uma expressão que consiste em um único valor, o resultado da função é NULL ou 0.

Exemplos

A seguinte consulta retorna a média dos valores na coluna VENUESEATS da tabela VENUE, seguida pelo desvio padrão de amostra e desvio padrão de população do mesmo conjunto de valores. VENUESEATS é uma coluna INTEGER. A escala do resultado é reduzida a 2 dígitos.

```
select avg(venueseats),
```

```
cast(stddev_samp(venueseats) as dec(14,2)) stddevsamp,
cast(stddev_pop(venueseats) as dec(14,2)) stddevpop
from venue;

avg  | stddevsamp | stddevpop
-------+------------+-----------
17503 |   27847.76 |  27773.20
(1 row)
```

A seguinte consulta retorna o desvio padrão de amostra para a coluna COMMISSION na tabela SALES. COMMISSION é uma coluna DECIMAL. A escala do resultado é reduzida a 10 dígitos.

```
select cast(stddev(commission) as dec(18,10))
from sales;

stddev
----------------
130.3912659086
(1 row)
```

A seguinte consulta converte o desvio padrão de amostra para a coluna COMMISSION para um inteiro.

```
select cast(stddev(commission) as integer)
from sales;

stddev
--------
130
(1 row)
```

A seguinte consulta retorna o desvio padrão da amostra e a raiz quadrada da variação da amostra para a coluna COMMISSION. Os resultados desses cálculos são os mesmos.

```
select
cast(stddev_samp(commission) as dec(18,10)) stddevsamp,
cast(sqrt(var_samp(commission)) as dec(18,10)) sqrtvarsamp
from sales;

stddevsamp   |  sqrtvarsamp
---------------+----------------
```

```
130.3912659086 | 130.3912659086
(1 row)
```

Função SUM

A função SUM retorna a soma dos valores de entrada da coluna ou expressão. A função SUM funciona com valores numéricos e ignora valores NULL.

Sintaxe

```
SUM ( [ DISTINCT | ALL ] expression )
```

Argumentos

expressão

A coluna ou expressão de destino na qual a função opera. A expressão é um destes tipos de dados:

- SMALLINT
- INTEGER
- BIGINT
- NUMERIC
- DECIMAL
- REAL
- DOUBLE PRECISION
- SUPER

DISTINCT | ALL

Com o argumento DISTINCT, a função elimina todos os valores duplicados da expressão especificada antes de calcular a soma. Com o argumento ALL, a função retém todos os valores duplicados da expressão para calcular a soma. ALL é o padrão.

Tipos de dados

Os tipos de argumento compatíveis com a função SUM são SMALLINT, INTEGER, BIGINT, NUMERIC, DECIMAL, REAL, DOUBLE PRECISION e SUPER.

Os tipos de retorno compatíveis com a função SUM são

- BIGINT para argumentos BIGINT, SMALLINT e INTEGER
- NUMERIC para argumentos NUMERIC
- DOUBLE PRECISION para argumentos de ponto flutuante
- Retorna o mesmo tipo de dados da expressão para qualquer outro tipo de argumento.

A precisão padrão para o resultado de uma função SUM com um argumento NUMERIC ou DECIMAL é 38. A escala do resultado é a mesma que a escala do argumento. Por exemplo, uma SUM de uma coluna DEC(5,2) retorna um tipo de dado DEC(38,2).

Exemplos

Encontre a soma de todas as comissões pagas da tabela SALES:

```
select sum(commission) from sales;

sum
-------------
16614814.65
(1 row)
```

Encontre o número de assentos em todos os locais de evento no estado da Flórida:

```
select sum(venueseats) from venue
where venuestate = 'FL';

sum
--------
250411
(1 row)
```

Encontre o número de assentos vendidos em maio:

```
select sum(qtysold) from sales, date
where sales.dateid = date.dateid and date.month = 'MAY';

sum
-------
```

```
32291
(1 row)
```

Funções VAR_SAMP e VAR_POP

As funções VAR_SAMP e VAR_POP retornam a variação da amostra e da população de um conjunto de valores numéricos (número inteiro, decimal ou ponto flutuante). O resultado da função VAR_SAMP é equivalente à raiz quadrada do desvio padrão da amostra do mesmo conjunto de valores.

VAR_SAMP e VARIANCE são sinônimos para a mesma função.

Sintaxe

```
VAR_SAMP | VARIANCE ( [ DISTINCT | ALL ] expression)
VAR_POP ( [ DISTINCT | ALL ] expression)
```

A expressão deve ter um tipo de dados de número inteiro, decimal ou ponto flutuante. Independente do tipo de dados da expressão, o tipo de retorno desta função é um número de precisão dupla.

> **Note**
>
> Os resultados dessas funções podem variar entre os clusters de data warehouse dependendo da configuração do cluster em cada caso.

Observações de uso

Quando a variação da amostra (VARIANCE ou VAR_SAMP) é calculada para uma expressão que consiste em um único valor, o resultado da função é NULL ou 0.

Exemplos

A seguinte consulta retorna a variação arredondada da amostra e da população para a coluna NUMTICKETS da tabela LISTING.

```
select avg(numtickets),
round(var_samp(numtickets)) varsamp,
round(var_pop(numtickets)) varpop
from listing;
```

```
avg | varsamp | varpop
-----+---------+--------
10 |      54 |     54
(1 row)
```

A seguinte consulta executa os mesmos cálculos, mas converte os resultados para valores decimais.

```
select avg(numtickets),
cast(var_samp(numtickets) as dec(10,4)) varsamp,
cast(var_pop(numtickets) as dec(10,4)) varpop
from listing;

avg | varsamp | varpop
-----+---------+---------
10 | 53.6291 | 53.6288
(1 row)
```

Funções de array

A seguir, você pode encontrar uma descrição para as funções de array do SQL aceitas pelo Amazon Redshift para acessar e manipular arrays.

Tópicos

- Função ARRAY
- Função ARRAY_CONCAT
- Função ARRAY_CONTAINS
- Função ARRAY_DISTINCT
- Função ARRAY_EXCEPT
- Função ARRAY_FLATTEN
- Função ARRAY_INTERSECTION
- Função ARRAY_POSITION
- Função ARRAY_POSITIONS
- Função ARRAY_SORT
- Função ARRAY_UNION
- Função ARRAYS_OVERLAP

- Função GET_ARRAY_LENGTH
- Função SPLIT_TO_ARRAY
- Função SUBARRAY

Função ARRAY

Cria um array do tipo de dados SUPER.

Sintaxe

```
ARRAY( [ expr1 ] [, expr2 [, ... ]] )
```

Argumento

expr1, expr2

Expressões de qualquer tipo de dado do Amazon Redshift, exceto os tipos de data e hora, já que o Amazon Redshift não transmite os tipos de data e hora para o tipo de dados SUPER. Os argumentos não precisam ser do mesmo tipo de dado.

Tipo de retorno

A função ARRAY exibe o tipo de dados SUPER.

Exemplo

Os exemplos a seguir mostram um array de valores numéricos e um array de diferentes tipos de dados.

```
--an array of numeric values
select ARRAY(1,50,null,100);
      array
------------------
 [1,50,null,100]
(1 row)

--an array of different data types
select ARRAY(1,'abc',true,3.14);
        array
----------------------
```

```
 [1,"abc",true,3.14]
(1 row)
```

Consulte também

- Função ARRAY_CONCAT
- Função SPLIT_TO_ARRAY
- Função ARRAY_FLATTEN

Função ARRAY_CONCAT

Concatena duas matrizes para criar uma que contenha todos os elementos na primeira matriz seguidos de todos os elementos na segunda. Os dois argumentos devem ser matrizes SUPER válidas.

Sintaxe

```
ARRAY_CONCAT( array1, array2 )
```

Argumentos

array1

O valor que especifica o primeiro dos dois arrays a serem concatenados.

array2

O valor que especifica o segundo dos dois arrays a serem concatenados.

Tipo de retorno

A função ARRAY_CONCAT exibe um valor de dados SUPER.

Exemplo

Os exemplos a seguir mostram a concatenação de dois arrays do mesmo tipo e a concatenação de dois arrays de tipos diferentes.

```
-- concatenating two arrays
```

```
SELECT ARRAY_CONCAT(ARRAY(10001,10002),ARRAY(10003,10004));
              array_concat
-----------------------------------
 [10001,10002,10003,10004]
(1 row)

-- concatenating two arrays of different types
SELECT ARRAY_CONCAT(ARRAY(10001,10002),ARRAY('ab','cd'));
          array_concat
------------------------------
 [10001,10002,"ab","cd"]
(1 row)
```

Consulte também

- Função ARRAY_UNION
- Função ARRAY_FLATTEN
- Função SPLIT_TO_ARRAY
- Função ARRAY_DISTINCT

Função ARRAY_CONTAINS

Verifica se a matriz contém o valor fornecido e exibe TRUE se for encontrada.

Sintaxe

```
ARRAY_CONTAINS( array, value [, null_match] )
```

Argumentos

array

Uma expressão SUPER que especifica a matriz na qual pesquisar.

valor

Um valor que especifica o elemento a ser pesquisado.

null_match

Um valor booliano que especifica como os valores NULL são tratados:

- null_match = FALSE: a pesquisa por NULL exibe NULL. Se a matriz contiver valores NULL e nenhuma correspondência for encontrada para um valor de pesquisa não NULL, exibirá NULL.
- null_match = TRUE: os valores NULL são tratados como elementos válidos e pesquisáveis. Se a matriz contiver valores NULL e nenhuma correspondência for encontrada para um valor de pesquisa não NULL, exibirá FALSE.

O padrão é TRUE.

O tratamento padrão de NULL também pode ser especificado pela opção de configuração:

```
-- same as null_match = TRUE
SET default_array_search_null_handling to TRUE;
```

Tipo de retorno

A função ARRAY_CONTAINS exibe um tipo BOOLIANO.

Exemplo

Os exemplos a seguir mostram a função ARRAY_CONTAINS.

```
SELECT ARRAY_CONTAINS(ARRAY('red', 'green'), 'red');
array_contains
---------------
 t
(1 row)
```

Os exemplos a seguir mostram o comportamento da função com null_match definido como TRUE.

```
SET default_array_search_null_handling to TRUE;

-- NULL search is enabled
SELECT ARRAY_CONTAINS(ARRAY('red', NULL, 'green'), NULL);
array_contains
---------------
 t
(1 row)

-- The array can contain NULLs
SELECT ARRAY_CONTAINS(ARRAY('red', NULL, 'green'), 'blue', TRUE);
```

```
array_contains
---------------
 f
(1 row)
```

Os exemplos a seguir mostram o comportamento da função com null_match definido como FALSE. Observe que, ao especificar o comportamento null_match na função, a configuração padrão será substituída.

```
-- same as null_match = TRUE
SET default_array_search_null_handling to TRUE;

-- NULL search is disabled. The default behavior is overridden
SELECT ARRAY_CONTAINS(ARRAY('red', 'green'), NULL, FALSE);
array_contains
---------------

(1 row)

-- same as null_match = FALSE
SET default_array_search_null_handling to FALSE;

-- The array contains NULL and a match is found
SELECT ARRAY_CONTAINS(ARRAY('red', NULL, 'green'), 'green');
array_contains
---------------
 t
(1 row)

-- The array contains NULL but no match is found
SELECT ARRAY_CONTAINS(ARRAY('red', NULL, 'green'), 'blue');
array_contains
---------------

(1 row)
```

Consulte também

- Função ARRAY_POSITION
- Função ARRAY_POSITIONS
- Função ARRAYS_OVERLAP

Função ARRAY_DISTINCT

Cria uma matriz que contém somente elementos exclusivos da matriz de entrada, removendo todas as duplicatas. Não é garantido que a ordem dos elementos na matriz de saída corresponda à ordem de entrada. Os valores NULL são tratados como elementos válidos. Se existirem vários NULLs na matriz de entrada, somente um NULL aparecerá na saída.

Sintaxe

```
ARRAY_DISTINCT( array )
```

Argumentos

array

Uma expressão SUPER que especifica a matriz.

Tipo de retorno

A função ARRAY_DISTINCT exibe um tipo SUPER.

Exemplo

Os exemplos a seguir mostram a função ARRAY_DISTINCT.

```
SELECT ARRAY_DISTINCT(ARRAY(1, 1, 'a', 'a', NULL, NULL));
 array_distinct
----------------
 [1,"a",null]
(1 row)

SELECT ARRAY_DISTINCT(ARRAY_CONCAT(ARRAY(1,2,3,3),ARRAY(2,3,4,4)));
 array_distinct
----------------
 [1,2,3,4]
(1 row)
```

Consulte também

- Função ARRAY_UNION
- Função ARRAY_SORT

- Função ARRAY_EXCEPT
- Função ARRAY_INTERSECTION

Função ARRAY_EXCEPT

Exibe a diferença entre duas matrizes, mantendo elementos da primeira que não existem na segunda. A função é à prova de NULL, o que significa que os NULLs são tratados como objetos conhecidos.

Sintaxe

```
ARRAY_EXCEPT( array1, array2 [, distinct] )
```

Argumentos

array1

Uma expressão SUPER que especifica a primeira matriz.

array2

Uma expressão SUPER que especifica a segunda matriz.

distinct

Um valor booliano que especifica se somente elementos distintos devem ser exibidos:

- distinct = FALSE: a semântica de vários conjuntos se aplica. Cada ocorrência de um elemento na primeira matriz é comparada às ocorrências na segunda matriz. Se a primeira matriz tiver mais ocorrências de um elemento do que a segunda matriz, as ocorrências extras serão preservadas no resultado.
- distinct = TRUE: a semântica de conjunto se aplica. As duas matrizes são tratadas como conjuntos, ignorando elementos duplicados. Os elementos da primeira matriz serão removidos se existirem em qualquer lugar da segunda matriz, independentemente da contagem de ocorrências.

O padrão é FALSE.

Tipo de retorno

A função ARRAY_EXCEPT exibe um tipo SUPER.

Exemplo

Os exemplos a seguir mostram a função ARRAY_EXCEPT.

```
SELECT ARRAY_EXCEPT(ARRAY('a','b','c'), ARRAY('b','c','d'));
 array_except
--------------
 ["a"]
(1 row)
```

Semântica de vários conjuntos:

```
SELECT ARRAY_EXCEPT(ARRAY('b','b','b','b'), ARRAY('b','b'));
 array_except
--------------
 ["b","b"]
(1 row)
```

Semântica de conjunto:

```
SELECT ARRAY_EXCEPT(ARRAY('a','b','b'), ARRAY('b'), TRUE);
 array_except
--------------
 ["a"]
(1 row)
```

Os NULLs são tratados como objeto conhecido.

```
SELECT ARRAY_EXCEPT(ARRAY('a',NULL), ARRAY(NULL));
 array_except
--------------
 ["a"]
(1 row)
```

Consulte também

- Função ARRAY_INTERSECTION
- Função ARRAY_UNION
- Função ARRAY_DISTINCT

- Função ARRAYS_OVERLAP

Função ARRAY_FLATTEN

Mescla vários arrays em um único array do tipo SUPER. Os elementos da primeira matriz interna aparecem primeiro, seguidos dos elementos das matrizes internas subsequentes. Os NULLs são tratados como objetos conhecidos.

Sintaxe

```
ARRAY_FLATTEN( array )
```

Argumentos

array

Uma expressão SUPER em forma de matriz.

Tipo de retorno

A função ARRAY_FLATTEN exibe um tipo SUPER.

Exemplo

O exemplo a seguir mostra a função ARRAY_FLATTEN.

```
SELECT ARRAY_FLATTEN(ARRAY(ARRAY(1,2,3,4),ARRAY(5,6,7,8),ARRAY(9,10)));
     array_flatten
-----------------------
 [1,2,3,4,5,6,7,8,9,10]
(1 row)
```

Consulte também

- Função ARRAY_CONCAT
- Função SUBARRAY
- Função ARRAY_DISTINCT

Função ARRAY_INTERSECTION

Exibe uma nova matriz que contém somente os elementos que existem nas duas matrizes de entrada. A função é à prova de NULL, o que significa que os NULLs são tratados como objetos conhecidos. A ordem dos elementos no resultado não é garantida.

Sintaxe

```
ARRAY_INTERSECTION( array1, array2 [, distinct] )
```

Argumentos

array1

Uma expressão SUPER que especifica uma matriz.

array2

Uma expressão SUPER que especifica uma matriz.

distinct

Um valor booliano que especifica se somente elementos distintos devem ser exibidos:

- distinct = FALSE: a semântica de vários conjuntos se aplica. Os elementos duplicados são preservados, e a frequência de cada elemento no resultado é igual ao mínimo de suas frequências nas duas matrizes de entrada.
- distinct = TRUE: a semântica de conjunto se aplica. Somente elementos exclusivos comuns às duas matrizes são exibidos, sem duplicações.

O padrão é FALSE.

Tipo de retorno

A função ARRAY_INTERSECTION exibe um tipo SUPER.

Exemplo

Os exemplos a seguir mostram a função ARRAY_INTERSECTION.

```
SELECT ARRAY_INTERSECTION(ARRAY('a','b','c'), ARRAY('b','c','d'));
 array_intersection
--------------------
```

```
 ["b","c"]
(1 row)
```

Semântica de vários conjuntos:

```
SELECT ARRAY_INTERSECTION(ARRAY('a','b','b'), ARRAY('b','b','b'));
 array_intersection
--------------------
 ["b","b"]
(1 row)
```

Semântica de conjunto:

```
SELECT ARRAY_INTERSECTION(ARRAY('a','b','b'), ARRAY('b','b','b'), TRUE);
 array_intersection
--------------------
 ["b"]
(1 row)
```

Os NULLs são tratados como objeto conhecido.

```
SELECT ARRAY_INTERSECTION(ARRAY('a',NULL), ARRAY('b',NULL));
 array_intersection
--------------------
 [null]
(1 row)
```

Consulte também

- Função ARRAY_EXCEPT
- Função ARRAYS_OVERLAP
- Função ARRAY_UNION
- Função ARRAY_DISTINCT

Função ARRAY_POSITION

Exibe a posição (índice) da primeira ocorrência de um elemento especificado em uma matriz. O índice é de base 0, em que 0 indica o primeiro elemento, 1 indica o segundo elemento e assim por diante. Exibirá -1 se o elemento não for encontrado na matriz.

A função exibe somente a posição da primeira ocorrência. Para encontrar todas as ocorrências, considere a possibilidade de usar a função Função ARRAY_POSITIONS.

Sintaxe

```
ARRAY_POSITION( array, value [, null_match] )
```

Argumentos

array

Uma expressão SUPER que especifica a matriz na qual pesquisar.

valor

Um valor que especifica o elemento a ser pesquisado.

null_match

Um valor booliano que especifica como os valores NULL são tratados:

- null_match = FALSE: a pesquisa por NULL exibe NULL. Se a matriz contiver valores NULL e nenhuma correspondência for encontrada para um valor de pesquisa não NULL, exibirá NULL.
- null_match = TRUE: os valores NULL são tratados como elementos válidos e pesquisáveis. Se a matriz contiver valores NULL e nenhuma correspondência for encontrada para um valor de pesquisa não NULL, exibirá -1.

O padrão é TRUE.

O tratamento padrão de NULL também pode ser especificado pela opção de configuração:

```
-- same as null_match = TRUE
SET default_array_search_null_handling to TRUE;
```

Tipo de retorno

A função ARRAY_POSITION exibe um tipo INT.

Exemplo

Os exemplos a seguir mostram a função ARRAY_POSITION.

```
SELECT ARRAY_POSITION(ARRAY('red', 'green'), 'red');
 array_position
----------------
              0
(1 row)

SELECT ARRAY_POSITION(ARRAY(1, 2, 3), 4);
 array_position
----------------
             -1
(1 row)

-- only the position of the first occurrence is returned
SELECT ARRAY_POSITION(ARRAY('red', 'green', 'red'), 'red');
 array_position
----------------
              0
(1 row)
```

Os exemplos a seguir mostram o comportamento da função com null_match definido como TRUE.

```
SET default_array_search_null_handling to TRUE;

-- NULL search is enabled
SELECT ARRAY_POSITION(ARRAY('red', NULL, 'green'), NULL);
 array_position
----------------
              1
(1 row)

-- The array can contain NULLs
SELECT ARRAY_POSITION(ARRAY('red', NULL, 'green'), 'blue', TRUE);
 array_position
----------------
             -1
(1 row)
```

Os exemplos a seguir mostram o comportamento da função com null_match definido como FALSE. Observe que, ao especificar o comportamento null_match na função, a configuração padrão será substituída.

```
-- same as null_match = TRUE
```

```
SET default_array_search_null_handling to TRUE;

-- NULL search is disabled. The default behavior is overridden
SELECT ARRAY_POSITION(ARRAY('red', 'green'), NULL, FALSE);
 array_position
----------------

(1 row)

-- same as null_match = FALSE
SET default_array_search_null_handling to FALSE;

-- The array contains NULL and a match is found
SELECT ARRAY_POSITION(ARRAY('red', NULL, 'green'), 'green');
 array_position
----------------
              2
(1 row)

-- The array contains NULL but no match is found
SELECT ARRAY_POSITION(ARRAY('red', NULL, 'green'), 'blue');
 array_position
----------------

(1 row)
```

Consulte também

- Função ARRAY_POSITIONS
- Função ARRAY_CONTAINS
- Função SUBARRAY

Função ARRAY_POSITIONS

Exibe uma matriz de posições (índices) em que o elemento especificado aparece na matriz de entrada. Os índices são de base 0, em que 0 indica o primeiro elemento, 1 indica o segundo elemento e assim por diante. Exibirá uma matriz vazia se o elemento não for encontrado.

Sintaxe

```
ARRAY_POSITIONS( array, value [, null_match] )
```

Argumentos

array

Uma expressão SUPER que especifica a matriz na qual pesquisar.

valor

Um valor que especifica o elemento a ser pesquisado.

null_match

Um valor booliano que especifica como os valores NULL são tratados:

- null_match = FALSE: a pesquisa por NULL exibe NULL. Se a matriz contiver valores NULL e nenhuma correspondência for encontrada para um valor de pesquisa não NULL, exibirá NULL.
- null_match = TRUE: os valores NULL são tratados como elementos válidos e pesquisáveis. Se a matriz contiver valores NULL e nenhuma correspondência for encontrada para um valor de pesquisa não NULL, exibirá uma matriz vazia.

O padrão é TRUE.

O tratamento padrão de NULL também pode ser especificado pela opção de configuração:

```
-- same as null_match = TRUE
SET default_array_search_null_handling to TRUE;
```

Tipo de retorno

A função ARRAY_POSITIONS exibe um tipo SUPER.

Exemplo

Os exemplos a seguir mostram a função ARRAY_POSITIONS.

```
SELECT ARRAY_POSITIONS(ARRAY('red', 'green', 'red'), 'red');
 array_positions
-----------------
 [0,2]
(1 row)
```

```
SELECT ARRAY_POSITIONS(ARRAY(1, 2, 3), 4);
 array_positions
-----------------
 []
(1 row)
```

Os exemplos a seguir mostram o comportamento da função com null_match definido como TRUE.

```
SET default_array_search_null_handling to TRUE;

-- NULL search is enabled
SELECT ARRAY_POSITIONS(ARRAY('red', NULL, 'green', NULL), NULL);
 array_positions
-----------------
 [1,3]
(1 row)

-- The array can contain NULLs
SELECT ARRAY_POSITIONS(ARRAY('red', NULL, 'green'), 'blue', TRUE);
 array_positions
-----------------
 []
(1 row)
```

Os exemplos a seguir mostram o comportamento da função com null_match definido como FALSE. Observe que, ao especificar o comportamento null_match na função, a configuração padrão será substituída.

```
-- same as null_match = TRUE
SET default_array_search_null_handling to TRUE;

-- NULL search is disabled. The default behavior is overridden
SELECT ARRAY_POSITIONS(ARRAY('red', 'green'), NULL, FALSE);
 array_positions
-----------------

(1 row)

-- same as null_match = FALSE
SET default_array_search_null_handling to FALSE;

-- The array contains NULL and a match is found
```

```
SELECT ARRAY_POSITIONS(ARRAY('red', NULL, 'green'), 'green');
 array_positions
-----------------
 [2]
(1 row)

-- The array contains NULL but no match is found
SELECT ARRAY_POSITIONS(ARRAY('red', NULL, 'green'), 'blue');
 array_positions
-----------------

(1 row)
```

Consulte também

- Função ARRAY_POSITION
- Função ARRAY_CONTAINS
- Função SUBARRAY

Função ARRAY_SORT

Exibe uma versão classificada da matriz de entrada na ordem crescente ou decrescente. É possível especificar onde os valores NULL devem aparecer no resultado. A função é à prova de NULL, o que significa que os NULLs são tratados como objetos conhecidos.

Sintaxe

```
ARRAY_SORT( array [, sort_ascending [, nulls_first]] )
```

Argumentos

array

Uma expressão SUPER que especifica a matriz a ser classificada.

sort_ascending

Um valor booliano que especifica se a matriz deve ser classificada ordem crescente ou decrescente:

- Especifique TRUE para classificar os elementos em ordem crescente.

- Especifique FALSE para classificar os elementos em ordem decrescente.

O padrão é TRUE.

nulls_first

Um valor booliano que especifica o posicionamento de NULL.

- Especifique TRUE para colocar valores NULL no início da matriz classificada.
- Especifique FALSE para colocar valores NULL no final da matriz classificada.

Tipo de retorno

A função ARRAY_SORT exibe um tipo SUPER.

Observação

Ao classificar matrizes contendo tipos de dados mistos, os elementos são ordenados de acordo com a seguinte precedência de tipo:

- Valores boolianos
- Valores numéricos
- Valores de string
- Matrizes
- Objetos/dicionários

Dentro de cada categoria de tipo, os elementos são classificados de acordo com sua ordem natural (por exemplo, os números são classificados numericamente e as strings, em ordem alfabética).

Exemplo

Os exemplos a seguir mostram a função ARRAY_SORT.

```
-- Ascending order (default)
SELECT ARRAY_SORT(ARRAY('b', 'a', 0, NULL, 1, false));
        array_sort
--------------------------
 [false,0,1,"a","b",null]
(1 row)
```

```
-- Descending order
SELECT ARRAY_SORT(ARRAY('b', 'a', 0, NULL, 1, false), False);
        array_sort
--------------------------
 [null,"b","a",1,0,false]
(1 row)

-- Descending order with NULLs at the end of the sorted array
SELECT ARRAY_SORT(ARRAY('b', 'a', 0, NULL, 1, false), False, False);
        array_sort
--------------------------
 ["b","a",1,0,false,null]
(1 row)
```

Consulte também

- Função ARRAY_DISTINCT
- Função ARRAY_FLATTEN
- Função SUBARRAY

Função ARRAY_UNION

Associa duas matrizes e exibe uma única matriz que contém todos os valores exclusivos, removendo quaisquer duplicatas. A função é à prova de NULL, o que significa que os NULLs são tratados como objetos conhecidos. A ordem dos elementos no resultado não é garantida.

Sintaxe

```
ARRAY_UNION( array1, array2 )
```

Argumentos

array1

Uma expressão SUPER que especifica a primeira matriz.

array2

Uma expressão SUPER que especifica a segunda matriz.

Tipo de retorno

A função ARRAY_UNION exibe um tipo SUPER.

Exemplo

Os exemplos a seguir mostram a função ARRAY_UNION.

```
SELECT ARRAY_UNION(ARRAY('a','b','b'), ARRAY('b','c','c'));
  array_union
---------------
 ["a","b","c"]
(1 row)
```

A ordem dos elementos não é garantida.

```
SELECT ARRAY_UNION(ARRAY('b','a','b'), ARRAY(NULL,'b',NULL));
  array_union
----------------
 ["b","a",null]
(1 row)
```

Consulte também

- Função ARRAY_CONCAT
- Função ARRAY_DISTINCT
- Função ARRAY_INTERSECTION
- Função ARRAY_EXCEPT

Função ARRAYS_OVERLAP

Verifica se duas matrizes têm algum elemento em comum. Exibirá TRUE se as matrizes compartilharem pelo menos um elemento ou FALSE se não houver elementos em comum. A função é à prova de NULL, o que significa que os NULLs são tratados como objetos conhecidos.

Sintaxe

```
ARRAYS_OVERLAP( array1, array2 )
```

Argumentos

array1

Uma expressão SUPER que especifica uma matriz.

array2

Uma expressão SUPER que especifica uma matriz.

Tipo de retorno

A função ARRAYS_OVERLAP exibe um tipo booliano.

Exemplo

Os exemplos a seguir mostram a função ARRAY_OVERLAP.

```
SELECT ARRAYS_OVERLAP(ARRAY('blue', 'green'), ARRAY('red', 'green'));
 arrays_overlap
----------------
 t
(1 row)
```

Os exemplos a seguir mostram que os NULLs são tratados como elementos válidos.

```
SELECT ARRAYS_OVERLAP(ARRAY('red', NULL, 'blue'), ARRAY('green', NULL));
 arrays_overlap
----------------
 t
(1 row)

SELECT ARRAYS_OVERLAP(ARRAY('red', NULL, 'blue'), ARRAY('green'));
 arrays_overlap
----------------
 f
(1 row)

SELECT ARRAYS_OVERLAP(JSON_PARSE('[null]'), ARRAY(NULL));
 arrays_overlap
----------------
 t
```

```
(1 row)
```

Consulte também

- Função ARRAY_INTERSECTION
- Função ARRAY_CONTAINS
- Função ARRAY_EXCEPT

Função GET_ARRAY_LENGTH

Exibe o tamanho de uma matriz SUPER com base em um caminho de matriz ou objeto.

Sintaxe

```
GET_ARRAY_LENGTH( super_expr )
```

Argumentos

super_expr

Uma expressão SUPER válida de forma de array.

Tipo de retorno

A função GET_ARRAY_LENGTH exibe um INT.

Exemplo

O exemplo a seguir mostra a função GET_ARRAY_LENGTH.

```
SELECT GET_ARRAY_LENGTH(ARRAY(1,2,3,4,5,6,7,8,9,10));
 get_array_length
----------------------
            10
(1 row)
```

Função SPLIT_TO_ARRAY

Usa um delimitador como parâmetro opcional. Se nenhum delimitador estiver presente, o padrão será uma vírgula.

Sintaxe

```
SPLIT_TO_ARRAY( string, delimiter )
```

Argumentos

string

A string de entrada a ser dividida.

delimitador

Um valor opcional no qual a string de entrada será dividida. O padrão é uma vírgula.

Tipo de retorno

A função SPLIT_TO_ARRAY exibe um valor de dados SUPER.

Exemplo

O exemplo a seguir mostra a função SPLIT_TO_ARRAY.

```
SELECT SPLIT_TO_ARRAY('12|345|6789', '|');
     split_to_array
------------------------
 ["12","345","6789"]
(1 row)
```

Consulte também

- Função ARRAY
- Função ARRAY_CONCAT
- Função SUBARRAY
- Função ARRAY_FLATTEN

Função SUBARRAY

Extrai uma parte de uma matriz a partir de uma posição especificada. Exibe uma nova matriz que contém o número especificado de elementos da matriz de entrada.

Sintaxe

```
SUBARRAY( super_expr, start_position, length )
```

Argumentos

super_expr

Uma expressão SUPER válida em forma de array.

start_position

Um número inteiro que especifica a posição inicial para extração. O índice é de base 0, em que 0 indica o primeiro elemento. Se start_position estiver além do tamanho da matriz, uma matriz vazia será exibida.

length

Um número inteiro opcional que especifica o número de elementos a serem extraídos. Se omitido, todos os elementos da posição inicial até o fim da matriz serão exibidos.

Tipo de retorno

A função SUBARRAY exibe um valor de dados SUPER.

Exemplos

Veja a seguir um exemplo de uma função SUBARRAY.

```
 SELECT SUBARRAY(ARRAY('a', 'b', 'c', 'd', 'e', 'f'), 2, 3);
   subarray
---------------
 ["c","d","e"]
(1 row)
```

Consulte também

- Função ARRAY_POSITION
- Função ARRAY_POSITIONS
- Função ARRAY_FLATTEN
- Função ARRAY_CONCAT

Funções agregadas bit-wise

As funções agregadas bit-wise computam operações de bits para executar a agregação de colunas e colunas de inteiros que podem ser convertidas ou arredondadas para valores inteiros.

Tópicos

- Usar NULLs em agregações bit a bit
- Compatibilidade DISTINCT para agregações bit-wise
- Exemplos de visão geral para funções bit a bit
- Função BIT_AND
- Função BIT_OR
- Função BOOL_AND
- Função BOOL_OR

Usar NULLs em agregações bit a bit

Quando você aplica uma função bit a bit a uma coluna que pode ser nula, quaisquer valores NULL são eliminados antes que o resultado da função seja calculado. Se nenhuma linha se qualificar para agregação, a função bit-wise retorna NULL. O mesmo comportamento se aplica a funções agregadas regulares. Veja um exemplo a seguir.

```
select sum(venueseats), bit_and(venueseats) from venue
where venueseats is null;

sum  | bit_and
------+---------
null |    null
(1 row)
```

Compatibilidade DISTINCT para agregações bit-wise

Assim como as outras funções de agregação, as funções bit a bit suportam a palavra-chave DISTINCT.

No entanto, o uso de DISTINCT com essas funções não tem qualquer impacto nos resultados. A primeira instância de um valor é suficiente para satisfazer operações AND ou OR bit a bit. Não faz diferença se valores duplicados estiverem presentes na expressão sendo avaliada.

Como o processamento DISTINCT provavelmente incorrerá em alguma sobrecarga na execução da consulta, recomendamos que você não use DISTINCT com funções bit a bit.

Exemplos de visão geral para funções bit a bit

A seguir, você pode encontrar alguns exemplos de visão geral que demonstram como trabalhar com as funções bit a bit. Você também pode encontrar exemplos de código específicos com cada descrição de função.

Exemplos para as funções bit a bit são baseados no banco de dados de exemplo TICKIT. A tabela USERS no banco de dados de amostra TICKIT contém várias colunas booleanas que indicam se cada usuário reconhecidamente gosta de diferentes tipos de eventos, tais como esportes, teatro, ópera, etc. Veja a seguir um exemplo.

```
select userid, username, lastname, city, state,
likesports, liketheatre
from users limit 10;

userid | username | lastname  |     city     | state | likesports | liketheatre
-------+----------+-----------+--------------+-------+------------+-------------
1 | JSG99FHE | Taylor    | Kent         | WA    | t          | t
9 | MSD36KVR | Watkins   | Port Orford  | MD    | t          | f
```

Suponha que uma nova versão da tabela USERS é criada de uma maneira diferente. Nesta nova versão, uma única coluna inteira que define (na forma binária) oito tipos de eventos que cada usuário gosta ou não gosta. Neste experimento, cada posição de bit representa um tipo de evento. Um usuário que goste de todos os oito tipos tem os oito bits definidos como 1 (como na primeira linha da tabela a seguir). Um usuário que não gosta de nenhum desses eventos tem todos os oito bits definidos como 0 (consulte a segunda linha). Um usuário que gosta apenas de esportes e jazz é representado na terceira linha a seguir.

USER	ESPORTE	TEATRO	JAZZ	ÓPERA	ROCK	VEGAS	BROADW	CLÁSSICA
Usuário 1	1	1	1	1	1	1	1	1
Usuário 2	0	0	0	0	0	0	0	0

USER	ESPORTE	TEATRO	JAZZ	ÓPERA	ROCK	VEGAS	BROADW	CLÁSSICA
Usuário 3	1	0	1	0	0	0	0	0

Na tabela do banco de dados, esses valores binários podem ser armazenados em uma única coluna LIKES como inteiros, conforme mostrado a seguir.

Usuário	Valor binário	Valor armazenados (inteiro)
Usuário 1	11111111	255
Usuário 2	00000000	0
Usuário 3	10100000	160

Função BIT_AND

A função BIT_AND executa operações AND bit-wise em todos os valores em uma única coluna ou expressão de inteiros. Essa função agrega cada bit de cada valor binário que corresponde a cada valor inteiro na expressão.

A função BIT_AND retorna um resultado de `0` se nenhum dos bits estiver definido como 1 ao longo de todos os valores. Se um ou mais bits estiver definido como 1 ao longo de todos os valores, a função retorna um valor inteiro. Este inteiro é o número que corresponde ao valor binário para os bits.

Por exemplo, uma tabela contém quatro valores inteiros em uma coluna: 3, 7, 10 e 22. Esses números inteiros são representados na forma binária da seguinte forma:

Inteiro	Valor binário
3	11
7	111

Inteiro	Valor binário
10	1010
22	10110

Uma operação de BIT_AND neste conjunto de dados identifica que todos os bits estão definidos como 1 somente na penúltima posição. O resultado é um valor binário de `00000010`, que representa o valor inteiro 2. Portanto, a função BIT_AND retorna 2.

Sintaxe

```
BIT_AND ( [DISTINCT | ALL] expression )
```

Argumentos

expressão

A coluna ou expressão de destino na qual a função opera. Essa expressão deve ter um tipo de dados INT, INT2 ou INT8. A função retorna um tipo de dados INT, INT2 ou INT8 equivalente.

DISTINCT | ALL

Com o argumento DISTINCT, a função elimina todos os valores duplicados para a expressão especificada antes de calcular o resultado. Com o argumento ALL, a função retém todos os valores duplicados. ALL é o padrão. Para obter mais informações, consulte Compatibilidade DISTINCT para agregações bit-wise.

Exemplos

Considerando que informações comerciais significativas são armazenadas em colunas de inteiros, você pode usar funções bit-wise para extrair e agregar essas informações. A seguinte consulta aplica a função BIT_AND na coluna de LIKES em uma tabela chamada USERLIKES e agrupa os resultados pela coluna CITY.

```
select city, bit_and(likes) from userlikes group by city
order by city;
city          | bit_and
--------------+---------
```

```
Los Angeles   |       0
Sacramento    |       0
San Francisco |       0
San Jose      |      64
Santa Barbara |     192
(5 rows)
```

Esses resultados podem ser interpretados da seguinte forma:

- O valor inteiro 192 para Santa Bárbara é traduzido para o valor binário `11000000`. Em outras palavras, todos os usuários nesta cidade gostam de esportes e de teatro, mas nem todos os usuários gostam de qualquer outro tipo de evento.
- O inteiro 64 o traduz para `01000000`. Portanto, para os usuários de San José, o único tipo de evento de que todos gostam é o teatro.
- Os valores de `0` para as outras três cidades indicam que nenhum “gosto” é compartilhado por todos os usuários nessas cidades.

Função BIT_OR

A função BIT_OR executa operações OR bit-wise em todos os valores em uma única coluna ou expressão de inteiros. Essa função agrega cada bit de cada valor binário que corresponde a cada valor inteiro na expressão.

Por exemplo, suponha que sua tabela contenha quatro valores inteiros em uma coluna: 3, 7, 10 e 22. Esses inteiros são representados na forma binária da maneira a seguir.

Inteiro	Valor binário
3	11
7	111
10	1010
22	10110

Se você aplicar a função BIT_OR ao conjunto de valores inteiros, a operação procurará qualquer valor em que 1 for encontrado em cada posição. Nesse caso, 1 existe nas últimas cinco posições

para pelo menos um dos valores, produzindo um resultado binário de `00011111`; portanto, função retorna `31` (ou `16 + 8 + 4 + 2 + 1`).

Sintaxe

```
BIT_OR ( [DISTINCT | ALL] expression )
```

Argumentos

expressão

A coluna ou expressão de destino na qual a função opera. Essa expressão deve ter um tipo de dados INT, INT2 ou INT8. A função retorna um tipo de dados INT, INT2 ou INT8 equivalente.

DISTINCT | ALL

Com o argumento DISTINCT, a função elimina todos os valores duplicados para a expressão especificada antes de calcular o resultado. Com o argumento ALL, a função retém todos os valores duplicados. ALL é o padrão. Para obter mais informações, consulte Compatibilidade DISTINCT para agregações bit-wise.

Exemplo

A seguinte consulta aplica a função BIT_OR à coluna LIKES em uma tabela chamada USERLIKES e agrupa os resultados pela coluna CITY.

```
select city, bit_or(likes) from userlikes group by city
order by city;
city          | bit_or
--------------+--------
Los Angeles   |    127
Sacramento    |    255
San Francisco |    255
San Jose      |    255
Santa Barbara |    255
(5 rows)
```

Para quatro da cidades listadas, todos os tipos de evento são apreciados por pelo menos um usuário (`255=11111111`). Para Los Angeles, todos os tipos de evento, exceto esportes, são apreciados por pelo menos um usuário (`127=01111111`).

Função BOOL_AND

A função BOOL_AND opera em uma única coluna ou expressão de boolianos ou inteiros. Essa função aplica lógica semelhante às funções BIT_AND e BIT_OR. Para essa função, o tipo de retorno é um valor booliano (`true` ou `false`).

Se todos os valores em um conjunto forem verdadeiros, a função BOOL_AND retorna `true` (`t`). Se qualquer valor for falso, a função retorna `false` (`f`).

Sintaxe

```
BOOL_AND ( [DISTINCT | ALL] expression )
```

Argumentos

expressão

A coluna ou expressão de destino na qual a função opera. Essa expressão deve ter um tipo de dados BOOLEAN ou de inteiros. O tipo de retorno da função é BOOLEAN.

DISTINCT | ALL

Com o argumento DISTINCT, a função elimina todos os valores duplicados para a expressão especificada antes de calcular o resultado. Com o argumento ALL, a função retém todos os valores duplicados. ALL é o padrão. Para obter mais informações, consulte Compatibilidade DISTINCT para agregações bit-wise.

Exemplos

Você pode usar as funções booleanas com expressões booleanas ou expressões de inteiro. Por exemplo, o seguinte retorno de consulta resultada da tabela USERS padrão no banco de dados TICKIT, que tem várias colunas booleanas.

A função BOOL_AND retorna `false` para todas as cinco linhas. Nem todos os usuários em cada um dos estados gostam de esportes.

```
select state, bool_and(likesports) from users
group by state order by state limit 5;

state | bool_and
------+---------
```

```
AB    | f
AK    | f
AL    | f
AZ    | f
BC    | f
(5 rows)
```

Função BOOL_OR

A função BOOL_OR opera em uma única coluna ou expressão de boolianos ou inteiros. Essa função aplica lógica semelhante às funções BIT_AND e BIT_OR. Para essa função, o tipo de retorno é um valor booliano (`true`, `false` ou NULL).

Se um ou mais valores de um conjunto for `true`, a função BOOL_OR exibirá `true` (`t`). Se todos os valores em um conjunto forem `false`, a função exibirá `false` (`f`). NULL poderá ser retornado se o valor for desconhecido.

Sintaxe

```
BOOL_OR ( [DISTINCT | ALL] expression )
```

Argumentos

expressão

A coluna ou expressão de destino na qual a função opera. Essa expressão deve ter um tipo de dados BOOLEAN ou de inteiros. O tipo de retorno da função é BOOLEAN.

DISTINCT | ALL

Com o argumento DISTINCT, a função elimina todos os valores duplicados para a expressão especificada antes de calcular o resultado. Com o argumento ALL, a função retém todos os valores duplicados. ALL é o padrão. Consulte Compatibilidade DISTINCT para agregações bit-wise.

Exemplos

Você pode usar as funções booleanas com expressões booleanas ou expressões de inteiro. Por exemplo, o seguinte retorno de consulta resultada da tabela USERS padrão no banco de dados TICKIT, que tem várias colunas booleanas.

A função BOOL_OR retorna `true` para todas as cinco linhas. Pelo menos um usuário em cada um dos estados gosta de esportes.

```
select state, bool_or(likesports) from users
group by state order by state limit 5;

state | bool_or
------+--------
AB    | t
AK    | t
AL    | t
AZ    | t
BC    | t
(5 rows)
```

O exemplo a seguir retorna NULL.

```
SELECT BOOL_OR(NULL = '123')
               bool_or
------
NULL
```

Expressões condicionais

Tópicos

- Expressão condicional CASE
- Função DECODE
- Funções GREATEST e LEAST
- Funções NVL e COALESCE
- Função NVL2
- Função NULLIF

O Amazon Redshift é compatível com algumas expressões condicionais que são extensões do padrão SQL.

Expressão condicional CASE

A expressão CASE é uma expressão condicional, semelhante às instruções if/then/else encontradas em outras linguagens. CASE é usada para especificar um resultado onde há várias condições. Use CASE onde uma expressão SQL é válida, como em um comando SELECT.

Há dois tipos de expressões CASE: simples e pesquisada.

- Em expressões CASE simples, uma expressão é comparada a um valor. Quando uma correspondência é encontrada, a ação especificada na cláusula THEN é aplicada. Se nenhuma correspondência é encontrada, a ação especificada na cláusula ELSE é aplicada.
- Em expressões CASE pesquisadas, cada CASE é avaliado com base em uma expressão booleana e a instrução CASE retorna o primeiro CASE correspondente. Se nenhuma correspondência for encontrada entre as cláusulas WHEN, a ação na cláusula ELSE será retornada.

Sintaxe

Instrução CASE simples usada para correspondência de condições:

```
CASE expression
  WHEN value THEN result
  [WHEN...]
  [ELSE result]
END
```

Instrução CASE pesquisada usada para avaliação de cada condição:

```
CASE
  WHEN condition THEN result
  [WHEN ...]
  [ELSE result]
END
```

Argumentos

expressão

Um nome de coluna ou qualquer expressão válida.

valor

Valor ao qual a expressão é comparada, tal como uma constante numérica ou string de caracteres.

Resultado

O valor ou uma expressão de destino retornado quando uma expressão ou condição booleana é avaliada. Os tipos de dados de todas as expressões de resultados devem poder ser convertidos em um único tipo de saída.

Condição

Uma expressão booliana que avalia como verdadeiro ou falso. Se a condição for verdadeira, o valor da expressão CASE será o resultado que segue a condição e o restante da expressão CASE não será processado. Se a condição for falsa, todas as cláusulas WHEN subsequentes serão avaliadas. Se nenhum resultado da condição WHEN for verdadeiro, o valor da expressão CASE será o resultado da cláusula ELSE. Se a cláusula ELSE for omitida e não nenhuma condição for verdadeira, o resultado será nulo.

Exemplos

Os exemplos a seguir usam a tabela VENUE e a tabela SALES dos dados de amostra TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Use uma expressão CASE simples para substituir `New York City` por `Big Apple` em uma consulta da tabela VENUE. Substitua todos os outros nomes de cidade por `other`.

```
select venuecity,
  case venuecity
    when 'New York City'
    then 'Big Apple' else 'other'
  end
from venue
order by venueid desc;

venuecity       |   case
----------------+-----------
Los Angeles     | other
New York City   | Big Apple
San Francisco   | other
Baltimore       | other
```

```
...
```

Use uma expressão CASE pesquisada para atribuir números de grupo com base no valor PRICEPAID para vendas individuais de ingresso:

```
select pricepaid,
  case when pricepaid <10000 then 'group 1'
    when pricepaid >10000 then 'group 2'
    else 'group 3'
  end
from sales
order by 1 desc;

pricepaid |  case
----------+---------
12624     | group 2
10000     | group 3
10000     | group 3
9996      | group 1
9988      | group 1
...
```

Função DECODE

Uma expressão DECODE substitui um valor específico por outro valor específico ou por valor padrão, dependendo do resultado de uma condição de igualdade. Essa operação é equivalente à operação de uma expressão CASE simples ou de uma instrução IF-THEN-ELSE.

Sintaxe

```
DECODE ( expression, search, result [, search, result ]... [ ,default ] )
```

Este tipo de expressão é útil para substituir abreviações ou códigos armazenados em tabelas por valores de negócio significativos que são necessário para relatórios.

Parâmetros

expressão

A origem do valor que você deseja comparar, tal como uma coluna em uma tabela.

pesquisar

O valor de destino que é comparado à expressão de origem, tal como um valor numérico ou string de caracteres. A expressão da pesquisa deve avaliar para um único valor fixo. Você não pode especificar uma expressão que avalie para um intervalo de valores, tal como `age between 20 and 29`; você precisa especificar pares separados de pesquisa/resultados para cada valor que você deseja substituir.

O tipo de dados de todas as instâncias da expressão de pesquisa deve ser o mesmo ou compatível. Os parâmetros da expressão e pesquisa também devem ser compatíveis.

Resultado

O valor de substituição que a consulta retorna quando a expressão corresponde ao valor da pesquisa. Você deve incluir pelo menos um par de pesquisa/resultados na expressão de DECODE.

OS tipos de dados de todas as instâncias da expressão de resultado deve ser o mesmo ou compatível. Os parâmetros resultado e padrão também devem ser compatíveis.

default

Um valor padrão opcional que é usado para casos em que a condição da pesquisa falha. Se você não especificar uma valor padrão, a expressão DECODE retornará NULL.

Observações de uso

Se o valor da expressão e o valor da pesquisa forem NULL, o resultado de DECODE será o valor de resultado correspondente. Para uma ilustração deste uso da função, consulte a seção de exemplos.

Quando usada essa forma, DECODE é semelhante a Função NVL2, mas há algumas diferenças. Para uma descrição dessas diferenças, consulte as observações de uso de NVL2.

Exemplos

Quando o valor `2008-06-01` existe na coluna caldate de datetable, o exemplo a seguir a substitui por `June 1st, 2008`. O exemplo substitui todos outros valores de caldate por NULL.

```
select decode(caldate, '2008-06-01', 'June 1st, 2008')
from datetable where month='JUN' order by caldate;
```

```
case
----------------
June 1st, 2008

...
(30 rows)
```

O seguinte exemplo usa uma expressão DECODE para converter as cinco colunas CATNAME abreviadas na tabela CATEGORY para nomes completos e para converter outros valores na coluna para Unknown.

```
select catid, decode(catname,
'NHL', 'National Hockey League',
'MLB', 'Major League Baseball',
'MLS', 'Major League Soccer',
'NFL', 'National Football League',
'NBA', 'National Basketball Association',
'Unknown')
from category
order by catid;

catid  | case
-------+---------------------------------
1      | Major League Baseball
2      | National Hockey League
3      | National Football League
4      | National Basketball Association
5      | Major League Soccer
6      | Unknown
7      | Unknown
8      | Unknown
9      | Unknown
10     | Unknown
11     | Unknown
(11 rows)
```

Use uma expressão DECODE para localizar locais de evento no Colorado e em Nevada com NULL na coluna VENUESEATS; converta os NULLS para zeros. Se a coluna VENUESEATS não for NULA, retorne 1 como resultado.

```
select venuename, venuestate, decode(venueseats,null,0,1)
from venue
```

```
where venuestate in('NV','CO')
order by 2,3,1;

venuename               | venuestate     | case
-----------------------------+----------------+-----------
Coors Field                  | CO         |   1
Dick's Sporting Goods Park   | CO         |   1
Ellie Caulkins Opera House   | CO         |   1
INVESCO Field        | CO         |   1
Pepsi Center        | CO         |   1
Ballys Hotel        | NV         |   0
Bellagio Hotel               | NV         |   0
Caesars Palace               | NV         |   0
Harrahs Hotel                | NV         |   0
Hilton Hotel                 | NV         |   0
...
(20 rows)
```

Funções GREATEST e LEAST

Retorna o maior ou menor valor de uma lista com qualquer número de expressões.

Sintaxe

```
GREATEST (value [, ...])
LEAST (value [, ...])
```

Parâmetros

expression_list

Uma lista de expressões, tais como nomes de coluna, separadas por vírgula. As expressões devem ser conversíveis para um tipo de dados comum. Valores NULL na lista são ignorados. Se todas as expressões avaliarem para NULL, o resultado será NULL.

Retornos

Retorna o maior (para GREATEST) ou menor (para LEAST) valor da lista de expressões fornecida.

Exemplo

O seguinte exemplo retorna o valor mais alto alfabeticamente para `firstname` ou `lastname`.

```
select firstname, lastname, greatest(firstname,lastname) from users
where userid < 10
order by 3;

 firstname | lastname  | greatest
-----------+-----------+-----------
 Lars      | Ratliff   | Ratliff
 Reagan    | Hodge     | Reagan
 Colton    | Roy       | Roy
 Barry     | Roy       | Roy
 Tamekah   | Juarez    | Tamekah
 Rafael    | Taylor    | Taylor
 Victor    | Hernandez | Victor
 Vladimir  | Humphrey  | Vladimir
 Mufutau   | Watkins   | Watkins
(9 rows)
```

Funções NVL e COALESCE

Retorna o valor da primeira expressão não nula em uma série de expressões. Quando um valor não nulo é encontrado, as demais expressões na lista não são avaliadas.

NVL é idêntica a COALESCE. São funções sinônimas. Este tópico explica a sintaxe e apresenta exemplos de ambas.

Sintaxe

```
NVL( expression, expression, ... )
```

A sintaxe de COALESCE é a mesma:

```
COALESCE( expression, expression, ... )
```

Se todas as expressões forem nulas, o resultado será nulo.

Essas funções são úteis para retornar um valor secundário quando um valor primário está ausente ou é nulo. Por exemplo, uma consulta pode retornar o primeiro dos três números de telefone disponíveis: celular, residencial ou profissional. A ordem das expressões na função determina a ordem de avaliação.

Argumentos

expressão

Uma expressão, tal como um nome de coluna, a ser avaliada quanto ao status nulo.

Tipo de retorno

O Amazon Redshift determina o tipo de dado do valor retornado com base nas expressões de entrada. Se os tipos de dados das expressões de entrada não tiverem um tipo comum, um erro será retornado.

Exemplos

Se a lista contiver expressões do tipo inteiro, a função retornará um inteiro.

```
SELECT COALESCE(NULL, 12, NULL);

coalesce
--------------
12
```

Esse exemplo, que é igual ao exemplo anterior, exceto pelo fato de usar NVL, retorna o mesmo resultado.

```
SELECT NVL(NULL, 12, NULL);

coalesce
--------------
12
```

O exemplo a seguir retorna um tipo string.

```
SELECT COALESCE(NULL, 'Amazon Redshift', NULL);

coalesce
--------------
Amazon Redshift
```

O exemplo a seguir resulta em um erro porque os tipos de dados variam na lista de expressões. Nesse caso, há uma string e um número na lista.

```
SELECT COALESCE(NULL, 'Amazon Redshift', 12);
ERROR: invalid input syntax for integer: "Amazon Redshift"
```

Para esse exemplo, crie uma tabela com as colunas START_DATE e END_DATE, insira linhas que incluam valores nulos e aplique uma expressão NVL nas duas colunas.

```
create table datetable (start_date date, end_date date);
insert into datetable values ('2008-06-01','2008-12-31');
insert into datetable values (null,'2008-12-31');
insert into datetable values ('2008-12-31',null);
```

```
select nvl(start_date, end_date)
from datetable
order by 1;

coalesce
------------
2008-06-01
2008-12-31
2008-12-31
```

O nome de coluna padrão para uma expressão NVL é COALESCE. A consulta a seguir retorna os mesmos resultados:

```
select coalesce(start_date, end_date)
from datetable
order by 1;
```

Para os exemplos de consulta a seguir, crie uma tabela com exemplos de informações de reservas em hotéis e insira várias linhas. Alguns registros contêm valores nulos.

```
create table booking_info (booking_id int, booking_code character(8), check_in date,
 check_out date, funds_collected numeric(12,2));
```

Insira os seguintes dados de exemplo. Alguns registros não têm uma data de `check_out` nem um valor de `funds_collected`.

```
insert into booking_info values (1, 'OCEAN_WV', '2023-02-01','2023-02-03',100.00);
insert into booking_info values (2, 'OCEAN_WV', '2023-04-22','2023-04-26',120.00);
insert into booking_info values (3, 'DSRT_SUN', '2023-03-13','2023-03-16',125.00);
```

```
insert into booking_info values (4, 'DSRT_SUN', '2023-06-01','2023-06-03',140.00);
insert into booking_info values (5, 'DSRT_SUN', '2023-07-10',null,null);
insert into booking_info values (6, 'OCEAN_WV', '2023-08-15',null,null);
```

A consulta a seguir retorna uma lista de datas. Se a data de `check_out` não estiver disponível, ela listará a data de `check_in`.

```
select coalesce(check_out, check_in)
from booking_info
order by booking_id;
```

Os resultados são mostrados a seguir. Observe que os dois últimos registros mostram a data de `check_in`.

```
coalesce
------------
2023-02-03
2023-04-26
2023-03-16
2023-06-03
2023-07-10
2023-08-15
```

Se você espera que uma consulta retorne valores nulos para determinadas funções ou colunas, você pode usar uma expressão NVL para substituir os nulos por outro valor qualquer. Por exemplo, as funções agregadas, tal como SUM, retornam valores nulos em vez de zeros quando não há linhas para avaliar. É possível usar uma expressão NVL para substituir esses valores nulos por `700.0`. Em vez de `485`, o resultado da soma de `funds_collected` é `1885` porque duas linhas que têm nulo são substituídas por `700`.

```
select sum(nvl(funds_collected, 700.0)) as sumresult from booking_info;

sumresult
------
 1885
```

Função NVL2

Retorna um de dois valores, dependendo se uma expressão especificada avalia para NULL ou NOT NULL.

Sintaxe

```
NVL2 ( expression, not_null_return_value, null_return_value )
```

Argumentos

expressão

Uma expressão, tal como um nome de coluna, a ser avaliada quanto ao status nulo.

not_null_return_value

O valor retornado se a expressão avaliar para NOT NULL. O valor not_null_return_value deve ter o mesmo tipo de dados que a expressão ou ser implicitamente conversível para esse tipo de dados.

null_return_value

O valor retornado se a expressão avaliar para NULL. O valor null_return_value deve ter o mesmo tipo de dados que a expressão ou ser implicitamente conversível para esse tipo de dados.

Tipo de retorno

O tipo de retorno de NVL2 é determinado da seguinte forma:

- Se not_null_return_value ou null_return_value for nulo, o tipo de dados da expressão não nula será retornado.

Se not_null_return_value e null_return_value não forem nulos:

- Se not_null_return_value e null_return_value tiverem o mesmo tipo de dados, esse tipo de dados será retornado.
- Se not_null_return_value e null_return_value tiverem diferentes tipos de dados numéricos, o menor tipo de dados numérico compatível será retornado.
- Se not_null_return_value e null_return_value tiverem diferentes tipos de dados datetime, um tipo de dados de timestamp será retornado.
- Se not_null_return_value e null_return_value tiverem diferentes tipos de dados de caracteres, o tipo de dados de not_null_return_value será retornado.
- Se not_null_return_value e null_return_value tiverem tipos de dados numéricos e não numéricos variados, o tipo de dados de not_null_return_value será retornado.

> **Important**
>
> Nos últimos dois casos onde o tipo de dados de not_null_return_value é retornado, o null_return_value é convertido implicitamente para esse tipo de dados. Se os tipos de dados forem incompatíveis, a função falhará.

Observações de uso

Função DECODE pode ser usada de forma similar a NVL2 quando os parâmetros da expressão e da pesquisa forem nulos. A diferença é que para DECODE, o retorno terá o valor e o tipo de dados do parâmetro do resultado. Em contraste, para NVL2, o retorno terá o valor do parâmetro not_null_return_value ou null_return_value, o que for selecionado pela função, mas terá o tipo de dados de not_null_return_value.

Por exemplo, supondo que column1 seja NULL, as consultas seguintes retornarão o mesmo valor. Contudo, o tipo de dados do valor de retorno para DECODE será INTEGER e o tipo de dados do valor de retorno para NVL2 será VARCHAR.

```
select decode(column1, null, 1234, '2345');
select nvl2(column1, '2345', 1234);
```

Exemplo

O seguinte exemplo altera alguns dados de amostra e, então, avalia dois campos para fornecer as informações de contato apropriadas para usuários:

```
update users set email = null where firstname = 'Aphrodite' and lastname = 'Acevedo';

select (firstname + ' ' + lastname) as name,
nvl2(email, email, phone) AS contact_info
from users
where state = 'WA'
and lastname  like 'A%'
order by lastname, firstname;

name        contact_info
--------------------+--------------------------------------------
Aphrodite Acevedo (906) 632-4407
Caldwell Acevedo  Nunc.sollicitudin@Duisac.ca
```

```
Quinn Adams    vel@adipiscingligulaAenean.com
Kamal Aguilar  quis@vulputaterisusa.com
Samson Alexander  hendrerit.neque@indolorFusce.ca
Hall Alford    ac.mattis@vitaediamProin.edu
Lane Allen     et.netus@risusDonec.org
Xander Allison    ac.facilisis.facilisis@Infaucibus.com
Amaya Alvarado    dui.nec.tempus@eudui.edu
Vera Alvarez   at.arcu.Vestibulum@pellentesque.edu
Yetta Anthony  enim.sit@risus.org
Violet Arnold  ad.litora@at.com
August Ashley  consectetuer.euismod@Phasellus.com
Karyn Austin   ipsum.primis.in@Maurisblanditenim.org
Lucas Ayers    at@elitpretiumet.com
```

Função NULLIF

Sintaxe

A expressão NULLIF compara dois argumentos e retorna nulo se os argumentos forem iguais. Se eles não forem iguais, o primeiro argumento é retornado. Essa expressão é o inverso da expressão NVL ou COALESCE.

```
NULLIF ( expression1, expression2 )
```

Argumentos

expression1, expression2

As colunas ou expressões de destino que são comparadas. O tipo de retorno é igual ao tipo da primeira expressão. O nome padrão da coluna do resultado de NULLIF é o nome da coluna da primeira expressão.

Exemplos

No exemplo a seguir, a consulta retorna a string `first` porque os argumentos não são iguais.

```
SELECT NULLIF('first', 'second');

case
-------
first
```

No exemplo a seguir, a consulta retorna NULL porque os argumentos literais da string são iguais.

```
SELECT NULLIF('first', 'first');

case
-------
NULL
```

No exemplo a seguir, a consulta retorna 1 porque os argumentos inteiros não são iguais.

```
SELECT NULLIF(1, 2);

case
-------
1
```

No exemplo a seguir, a consulta retorna NULL porque os argumentos inteiros são iguais.

```
SELECT NULLIF(1, 1);

case
-------
NULL
```

No exemplo a seguir, a consulta retorna nulo quando há correspondência dos valores LISTID e SALESID:

```
select nullif(listid,salesid), salesid
from sales where salesid<10 order by 1, 2 desc;

listid  | salesid
--------+---------
     4  |       2
     5  |       4
     5  |       3
     6  |       5
     10 |       9
     10 |       8
     10 |       7
     10 |       6
        |       1
```

```
(9 rows)
```

É possível usar NULLIF para garantir que strings vazias sejam sempre retornadas como nulas. No exemplo abaixo, o expressão NULLIF retorna um valor nulo ou uma string que contém pelo menos um caractere.

```
insert into category
values(0,'','Special','Special');

select nullif(catgroup,'') from category
where catdesc='Special';

catgroup
----------
null
(1 row)
```

NULLIF ignora espaços em branco à direita. Se uma string não estiver vazia, mas contiver espaços em branco, NULLIF continuará retornando nulo:

```
create table nulliftest(c1 char(2), c2 char(2));

insert into nulliftest values ('a','a ');

insert into nulliftest values ('b','b');


select nullif(c1,c2) from nulliftest;
c1
------
null
null
(2 rows)
```

Funções de formatação de tipo de dados

Tópicos

- Função CAST
- Função CONVERT
- TEXT_TO_INT_ALT

- TEXT_TO_NUMERIC_ALT
- TO_CHAR
- Função TO_DATE
- TO_NUMBER
- Função TRY_CAST
- Strings de formato datetime
- Strings de formato numérico
- Caracteres de formatação de estilo Teradata para dados numéricos

As funções de formatação do tipo de dados fornecem uma forma fácil de converter valores de um tipo de dados para outro. Para cada uma dessas funções, o primeiro argumento é sempre o valor a ser formatado e o segundo argumento contém o modelo para o novo formato. O Amazon Redshift suporta várias funções de formatação de tipo de dados.

Função CAST

A função CAST converte um tipo de dados em outro compatível. Por exemplo, é possível converter uma string em uma data ou um tipo numérico em uma string. CAST executa uma conversão em tempo de execução, o que significa que a conversão não altera o tipo de dados de um valor em uma tabela de origem. Isso é alterado somente no contexto da consulta.

A função CAST é muito semelhante à the section called “CONVERT”, pois ambas convertem um tipo de dado em outro, mas têm nomes diferentes.

Determinados tipos de dados exigem uma conversão explícita para outros tipos de dados usando a função CAST ou CONVERT. Outros tipos de dados podem ser convertidos implicitamente, como parte de outro comando, sem usar CAST ou CONVERT. Consulte Compatibilidade e conversão dos tipos.

Sintaxe

Use uma dessas duas formas equivalentes de sintaxe para transmitir expressões de um tipo de dados para outro.

```
CAST ( expression AS type )
expression :: type
```

Argumentos

expressão

Uma expressão que avalia para um ou mais valores, tal como um nome de coluna ou um literal. A conversão de valores nulos retorna nulos. A expressão não pode conter strings em branco ou vazias.

type

Um dos compatíveis Tipos de dados.

Tipo de retorno

CAST retorna o tipo de dados especificado pelo argumento type.

> **Note**
>
> O Amazon Redshift retornará um erro se você tentar realizar uma conversão problemática, como uma conversão DECIMAL que perde precisão, como a seguinte:
>
> ```
> select 123.456::decimal(2,1);
> ```
>
> ou uma conversão INTEGER que causa um transbordamento:
>
> ```
> select 12345678::smallint;
> ```

Exemplos

Alguns dos exemplos usam o banco de dados TICKIT como exemplo. Para ter mais informações sobre como configurar os dados de exemplo, consulte Carregar dados de amostra.

As seguintes duas consultas são equivalentes. Ambas convertem um valor decimal em um número inteiro:

```
select cast(pricepaid as integer)
from sales where salesid=100;
```

```
pricepaid
-----------
162
(1 row)
```

```
select pricepaid::integer
from sales where salesid=100;

pricepaid
-----------
162
(1 row)
```

O seguinte produz um resultado semelhante. Ele não exige dados de exemplo para ser executado:

```
select cast(162.00 as integer) as pricepaid;

pricepaid
-----------
162
(1 row)
```

Neste exemplo, os valores em uma coluna de carimbo de data/hora são transmitidos como datas, o que resulta na remoção do horário de cada resultado:

```
select cast(saletime as date), salesid
from sales order by salesid limit 10;

 saletime  | salesid
-----------+---------
2008-02-18 |       1
2008-06-06 |       2
2008-06-06 |       3
2008-06-09 |       4
2008-08-31 |       5
2008-07-16 |       6
2008-06-26 |       7
2008-07-10 |       8
2008-07-22 |       9
2008-08-06 |      10
(10 rows)
```

Se você não usasse CAST conforme ilustrado no exemplo anterior, os resultados incluiriam o horário: 2008-02-18 02:36:48.

A consulta a seguir converte dados de caracteres variáveis em uma data. Ele não exige dados de exemplo para ser executado.

```
select cast('2008-02-18 02:36:48' as date) as mysaletime;

mysaletime
--------------------
2008-02-18
(1 row)
```

Neste exemplo, os valores em uma coluna de data são convertidos como timestamps:

```
select cast(caldate as timestamp), dateid
from date order by dateid limit 10;

      caldate       | dateid
--------------------+--------
2008-01-01 00:00:00 |   1827
2008-01-02 00:00:00 |   1828
2008-01-03 00:00:00 |   1829
2008-01-04 00:00:00 |   1830
2008-01-05 00:00:00 |   1831
2008-01-06 00:00:00 |   1832
2008-01-07 00:00:00 |   1833
2008-01-08 00:00:00 |   1834
2008-01-09 00:00:00 |   1835
2008-01-10 00:00:00 |   1836
(10 rows)
```

Em um caso como no exemplo anterior, é possível obter controle adicional sobre a formatação de saída usando TO_CHAR.

Neste exemplo, um número inteiro é convertido como uma string de caracteres:

```
select cast(2008 as char(4));

bpchar
--------
2008
```

Neste exemplo, um valor DECIMAL(6,3) é convertido como um valor DECIMAL(4,1):

```
select cast(109.652 as decimal(4,1));

numeric
---------
109.7
```

Este exemplo mostra uma expressão mais complexa. A coluna PRICEPAID (uma coluna DECIMAL(8,2)) na tabela SALES é convertida em uma coluna DECIMAL(38,2) e os valores são multiplicados por 100.000.000.000.000.000.000.

```
select salesid, pricepaid::decimal(38,2)*100000000000000000000
as value from sales where salesid<10 order by salesid;


 salesid |           value
---------+----------------------------
       1 | 72800000000000000000000.00
       2 |  76000000000000000000000.00
       3 | 35000000000000000000000.00
       4 | 17500000000000000000000.00
       5 | 15400000000000000000000.00
       6 | 39400000000000000000000.00
       7 | 78800000000000000000000.00
       8 | 19700000000000000000000.00
       9 | 59100000000000000000000.00
(9 rows)
```

> **Note**
>
> Não é possível executar uma operação CAST ou CONVERT no tipo de dados `GEOMETRY` para alterá-lo para outro tipo de dados. No entanto, você pode fornecer uma representação hexadecimal de uma string literal em formato binário bem-conhecido estendido (EWKB - extended well-known binary) como entrada para funções que aceitam um argumento `GEOMETRY`. Por exemplo, a função `ST_AsText` seguinte espera um tipo de dados `GEOMETRY`.
>
> ```
> SELECT ST_AsText('010100000000000000000001C400000000000002040');
> ```

```
st_astext
------------
 POINT(7 8)
```

Também é possível especificar explicitamente o tipo de dados GEOMETRY.

```
SELECT ST_AsText('0101000000000000000000144000000000000001840'::geometry);
```

```
st_astext
------------
 POINT(5 6)
```

Função CONVERT

Semelhante à função CAST, a função CONVERT converte um tipo de dado em outro compatível. Por exemplo, é possível converter uma string em uma data ou um tipo numérico em uma string. CONVERT executa uma conversão em runtime, o que significa que a conversão não altera o tipo de dado de um valor em uma tabela de origem. Isso é alterado somente no contexto da consulta.

Determinados tipos de dados exigem uma conversão explícita para outros tipos de dados usando a função CONVERT. Outros tipos de dados podem ser convertidos implicitamente, como parte de outro comando, sem usar CAST ou CONVERT. Consulte Compatibilidade e conversão dos tipos.

Sintaxe

```
CONVERT ( type, expression )
```

Argumentos

type

Um dos compatíveis Tipos de dados.

expressão

Uma expressão que avalia para um ou mais valores, tal como um nome de coluna ou um literal. A conversão de valores nulos retorna nulos. A expressão não pode conter strings em branco ou vazias.

Tipo de retorno

CONVERT retorna o tipo de dados especificado pelo argumento type.

> **Note**
>
> O Amazon Redshift retornará um erro se você tentar realizar uma conversão problemática, como uma conversão DECIMAL que perde precisão, como a seguinte:
>
> ```
> SELECT CONVERT(decimal(2,1), 123.456);
> ```
>
> ou uma conversão INTEGER que causa um transbordamento:
>
> ```
> SELECT CONVERT(smallint, 12345678);
> ```

Exemplos

Alguns dos exemplos usam o banco de dados TICKIT como exemplo. Para ter mais informações sobre como configurar os dados de exemplo, consulte Carregar dados de amostra.

A consulta a seguir usa a função CONVERT para converter uma coluna de números decimais em inteiros.

```
SELECT CONVERT(integer, pricepaid)
FROM sales WHERE salesid=100;
```

Este exemplo converte um número inteiro em uma string.

```
SELECT CONVERT(char(4), 2008);
```

Neste exemplo, a data e hora atuais são convertidas em um tipo de dados de caractere variável:

```
SELECT CONVERT(VARCHAR(30), GETDATE());

getdate
---------
2023-02-02 04:31:16
```

Este exemplo converte a coluna saletime para remover as datas de cada linha e manter apenas a hora.

```
SELECT CONVERT(time, saletime), salesid
FROM sales order by salesid limit 10;
```

Para obter informações sobre como converter um carimbo de data/hora de um fuso horário para outro, consulte Função CONVERT_TIMEZONE. Para funções adicionais de data e hora, consulte Perfis de data e hora.

O exemplo a seguir converte dados de caracteres variáveis em objetos de data e hora.

```
SELECT CONVERT(datetime, '2008-02-18 02:36:48') as mysaletime;
```

> **Note**
>
> Não é possível executar uma operação CAST ou CONVERT no tipo de dados `GEOMETRY` para alterá-lo para outro tipo de dados. No entanto, você pode fornecer uma representação hexadecimal de uma string literal em formato binário bem-conhecido estendido (EWKB - extended well-known binary) como entrada para funções que aceitam um argumento `GEOMETRY`. Por exemplo, a função `ST_AsText` seguinte espera um tipo de dados `GEOMETRY`.
>
> ```
> SELECT ST_AsText('010100000000000000000001C40000000000002040');
> ```
>
> ```
> st_astext
> ------------
>  POINT(7 8)
> ```
>
> Também é possível especificar explicitamente o tipo de dados `GEOMETRY`.
>
> ```
> SELECT ST_AsText('010100000000000000000014400000000000001840'::geometry);
> ```
>
> ```
> st_astext
> ------------
>  POINT(5 6)
> ```

TEXT_TO_INT_ALT

TEXT_TO_INT_ALT converte uma string de caracteres em um inteiro usando formatação no estilo Teradata. Os dígitos de fração no resultado são truncados.

Sintaxe

```
TEXT_TO_INT_ALT (expression [ , 'format'])
```

Argumentos

expressão

Uma expressão que resulta em um ou mais valores CHAR ou VARCHAR, como um nome de coluna ou sequência literal. A conversão de valores nulos retorna nulos. A função converte strings de caracteres em branco ou vazio para 0.

formato

Um literal de string que define o formato da expressão de entrada. Para obter mais informações sobre os caracteres de formatação que podem ser especificados, consulte Caracteres de formatação de estilo Teradata para dados numéricos.

Tipo de retorno

TEXT_TO_INT_ALT retorna um valor INTEGER.

A parte fracionária do resultado da conversão é truncada.

O Amazon Redshift retorna um erro se a conversão para a frase format que você especificar não for bem-sucedida.

Exemplos

O exemplo a seguir converte a string da expressão de entrada “123-” para o inteiro -123.

```
select text_to_int_alt('123-');
```

```
text_to_int_alt
----------
      -123
```

O exemplo a seguir converte a string da expressão de entrada “2147483647+” para o inteiro 2147483647.

```
select text_to_int_alt('2147483647+');
```

```
text_to_int_alt
----------
2147483647
```

O exemplo a seguir converte a string da expressão de entrada exponencial “-123E-2” para o inteiro -1.

```
select text_to_int_alt('-123E-2');
```

```
text_to_int_alt
----------
        -1
```

O exemplo a seguir converte a string da expressão de entrada “2147483647+” para o inteiro 2147483647.

```
select text_to_int_alt('2147483647+');
```

```
text_to_int_alt
----------
2147483647
```

O exemplo a seguir converte a string da expressão de entrada “123{“ com o a frase de format “999S” para o inteiro 1230. O caractere S indica um Decimal com zona assinada. Para obter mais informações, consulte Caracteres de formatação de estilo Teradata para dados numéricos.

```
select text_to_int_alt('123{', '999S');
```

```
text_to_int_alt
----------
      1230
```

O exemplo a seguir converte a string da expressão de entrada “USD123“ com o a frase de format “C9(I)” para o inteiro 123. Consulte Caracteres de formatação de estilo Teradata para dados numéricos.

```
select text_to_int_alt('USD123', 'C9(I)');
```

```
text_to_int_alt
----------
       123
```

O exemplo a seguir especifica uma coluna de tabela como a expressão de entrada.

```
select text_to_int_alt(a), text_to_int_alt(b) from t_text2int order by 1;
```

```
 text_to_int_alt | text_to_int_alt
-----------------+-----------------
            -123 |            -123
            -123 |            -123
             123 |             123
             123 |             123
```

A seguir está a definição da tabela e a instrução de inserção para este exemplo.

```
create table t_text2int (a varchar(200), b char(200));
```

```
insert into t_text2int VALUES('123', '123'),('123.123', '123.123'), ('-123', '-123'),
 ('123-', '123-');
```

TEXT_TO_NUMERIC_ALT

TEXT_TO_NUMERIC_ALT executa uma operação de conversão no estilo Teradata para converter uma string de caracteres em um formato de dados numérico.

Sintaxe

```
TEXT_TO_NUMERIC_ALT (expression [, 'format'] [, precision, scale])
```

Argumentos

expressão

Uma expressão que avalia um ou mais valores CHAR ou VARCHAR, como um nome de coluna ou literal. A conversão de valores nulos retorna nulos. As cadeias em branco ou vazias são convertidas em 0.

formato

Um literal de string que define o formato da expressão de entrada. Para obter mais informações, consulte Caracteres de formatação de estilo Teradata para dados numéricos.

precisão

O número de dígitos no resultado numérico. O padrão é 38.

scale

O número de dígitos à direita do separador decimal no resultado numérico. O padrão é 0.

Tipo de retorno

TEXT_TO_NUMERIC_ALT retorna um número DECIMAL.

O Amazon Redshift retorna um erro se a conversão para a frase de format que você especificar não for bem-sucedida.

O Amazon Redshift lança a string da expressão de entrada para o tipo numérico com a precisão mais alta que você especificar para esse tipo na opção precisão. Se o comprimento do valor numérico exceder o valor especificado para precisão, o Amazon Redshift arredonda o valor numérico de acordo com as seguintes regras:

- Se o comprimento do resultado de conversão exceder o comprimento especificado na frase de format, o Amazon Redshift retorna um erro.
- Se o resultado for convertido em um valor numérico, o resultado será arredondado para o valor mais próximo. Se a porção fracionada estiver exatamente a meio caminho entre o resultado de elenco superior e inferior, o resultado será arredondado para o valor par mais próximo.

Exemplos

O exemplo a seguir converte a string da expressão de entrada “1.5” para o valor numérico “2”. Como a declaração não especifica a escala, o padrão da escala é 0 e o resultado da conversão não inclui um resultado de fração. Como .5 está no meio do caminho entre 1 e 2, o resultado da conversão é arredondado para o valor par de 2.

```
select text_to_numeric_alt('1.5');
```

```
 text_to_numeric_alt
---------------------
                   2
```

O exemplo a seguir converte a string da expressão de entrada '2.51' para o valor numérico 3. Como a declaração não especifica o valor da escala, o padrão da escala é 0 e o resultado da conversão não inclui um resultado de fração. Como 0,51 está mais próximo de 3 do que 2, o resultado da conversão é arredondado para o valor de 3.

```
select text_to_numeric_alt('2.51');
```

```
 text_to_numeric_alt
---------------------
                   3
```

O exemplo a seguir converte a string da expressão de entrada 123.52501 com uma precisão de 10 e uma escala de 2 para o valor numérico 123,53.

```
select text_to_numeric_alt('123.52501', 10, 2);
```

```
 text_to_numeric_alt
---------------------
              123.53
```

O exemplo a seguir converte a string da expressão de entrada “123{“ com a frase de formato “999S” no numérico 1230. O caractere S indica um Decimal com zona assinada. Para obter mais informações, consulte Caracteres de formatação de estilo Teradata para dados numéricos.

```
select text_to_numeric_alt('123{', '999S');
```

```
text_to_int_alt
----------
      1230
```

O exemplo a seguir converte a string da expressão de entrada “USD123“ com a frase de formato “C9(I)” no numérico 124. Consulte Caracteres de formatação de estilo Teradata para dados numéricos.

```
select text_to_numeric_alt('USD123.9', 'C9(I)');
```

```
text_to_numeric_alt
----------
       124
```

O exemplo a seguir especifica uma coluna de tabela como a expressão de entrada.

```
select text_to_numeric_alt(a), text_to_numeric_alt(b) from t_text2numeric order by 1;
```

```
          text_to_numeric_alt           |          text_to_numeric_alt
----------------------------------------+----------------------------------------
 -99999999999999999999999999999999999999 | -99999999999999999999999999999999999999
                                  -12300 |                                  -12300
                                     123 |                                     123
                                     123 |                                     123
  99999999999999999999999999999999999999 |  99999999999999999999999999999999999999
```

A seguir está a definição da tabela e a instrução de inserção para este exemplo.

```
create table  t_text2numeric (a varchar(200), b char(200));
```

```
insert into  t_text2numeric values
('123', '123'),
('+123.456', '+123.456'),
('-' || repeat('9', 38), '-' || repeat('9', 38)),
(repeat('9', 38) || '+', repeat('9', 38) || '+'),
('-123E2', '-123E2');
```

TO_CHAR

TO_CHAR Converte uma expressão de timestamp ou numérica para o formato de dados de string de caracteres.

Sintaxe

```
TO_CHAR (timestamp_expression | numeric_expression , 'format')
```

Argumentos

timestamp_expression

Uma expressão que resulta em um valor do tipo TIMESTAMP ou TIMESTAMPTZ ou um valor que pode ser implicitamente forçado para um timestamp.

numeric_expression

Uma expressão que resulta em um valor de tipo de dados numérico ou em um valor que pode implicitamente ser convertido para tipo numérico. Para obter mais informações, consulte Tipos numéricos. TO_CHAR insere um espaço à esquerda da string numérica.

> **Note**
>
> TO_CHAR não é compatível com valores do tipo DECIMAL de 128 bits.

formato

O formato para o novo valor. Para obter os formatos válidos, consulte Strings de formato datetime e Strings de formato numérico.

Tipo de retorno

VARCHAR

Exemplos

O exemplo a seguir converte um carimbo de data/hora em um valor com a data e a hora em um formato com o nome do mês preenchido com nove caracteres, o nome do dia da semana e o número do dia do mês.

```
select to_char(timestamp '2009-12-31 23:15:59', 'MONTH-DY-DD-YYYY HH12:MIPM');

to_char
-------------------------
DECEMBER -THU-31-2009 11:15PM
```

O exemplo a seguir converte um carimbo de data/hora em um valor com o número do dia do ano.

```
select to_char(timestamp '2009-12-31 23:15:59', 'DDD');

to_char
-------------------------
365
```

O exemplo a seguir converte um carimbo de data/hora em um número do dia da semana da norma ISO.

```
select to_char(timestamp '2022-05-16 23:15:59', 'ID');

to_char
-------------------------
1
```

O exemplo a seguir extrai o nome do mês de uma data.

```
select to_char(date '2009-12-31', 'MONTH');

to_char
-------------------------
DECEMBER
```

O seguinte exemplo converte cada valor de STARTTIME na tabela EVENT em uma string que consiste em horas, minutos e segundos.

```
select to_char(starttime, 'HH12:MI:SS')
from event where eventid between 1 and 5
order by eventid;

to_char
----------
02:30:00
```

```
08:00:00
02:30:00
02:30:00
07:00:00
```

O exemplo a seguir converte um valor de timestamp inteiro em um formato diferente.

```
select starttime, to_char(starttime, 'MON-DD-YYYY HH12:MIPM')
from event where eventid=1;

      starttime      |       to_char
---------------------+---------------------
 2008-01-25 14:30:00 | JAN-25-2008 02:30PM
```

O exemplo a seguir converte um literal de timestamp em uma string de caracteres.

```
select to_char(timestamp '2009-12-31 23:15:59','HH24:MI:SS');

to_char
----------
23:15:59
```

O exemplo a seguir converte um número decimal em uma string de caracteres.

```
select to_char(125.8, '999.99');

to_char
---------
125.80
```

O exemplo a seguir converte um número decimal em uma string de caracteres.

```
select to_char(125.8, '999D99');

to_char
---------
125.80
```

O exemplo a seguir converte um número em uma string de caracteres com um zero inicial.

```
select to_char(125.8, '0999D99');
```

```
to_char
---------
0125.80
```

O exemplo a seguir converte um número em uma string de caracteres com o sinal menos no final.

```
select to_char(-125.8, '999D99S');

to_char
---------
125.80-
```

O exemplo a seguir converte um número em uma string de caracteres com o sinal positivo ou negativo na posição especificada.

```
select to_char(125.8, '999D99SG');

to_char
---------
125.80+
```

O exemplo a seguir converte um número em uma string de caracteres com o sinal positivo na posição especificada.

```
select to_char(125.8, 'PL999D99');

to_char
---------
+ 125.80
```

O exemplo a seguir converte um número em uma string de caracteres com o símbolo de moeda.

```
select to_char(-125.88, '$S999D99');

to_char
---------
$-125.88
```

O exemplo a seguir converte um número em uma string de caracteres com o símbolo de moeda na posição especificada.

```
select to_char(-125.88, 'S999D99L');

to_char
---------
-125.88$
```

O exemplo a seguir converte um número em uma string de caracteres usando um separador de milhar (vírgula).

```
select to_char(1125.8, '9,999.99');

to_char
---------
1,125.80
```

O exemplo a seguir converte um número em uma string de caracteres usando colchetes angulares para números negativos.

```
select to_char(-125.88, '$999D99PR');

to_char
---------
$<125.88>
```

O exemplo a seguir converte um número em uma string de numerais romanos.

```
select to_char(125, 'RN');

to_char
---------
   CXXV
```

O exemplo a seguir converte uma data em um código de século.

```
select to_char(date '2020-12-31', 'CC');

to_char
---------
21
```

O exemplo a seguir exibe o dia da semana.

```
SELECT to_char(current_timestamp, 'FMDay, FMDD HH12:MI:SS');

to_char
-----------------------
Wednesday, 31 09:34:26
```

O exemplo a seguir exibe o sufixo de número ordinal de um número.

```
SELECT to_char(482, '999th');

to_char
-----------------------
 482nd
```

O exemplo a seguir subtrai a comissão do preço pago na tabela de vendas. A diferença é, então, arredondada para cima e convertida em um numeral romano, exibido na coluna to_char:

```
select salesid, pricepaid, commission, (pricepaid - commission)
as difference, to_char(pricepaid - commission, 'rn') from sales
group by sales.pricepaid, sales.commission, salesid
order by salesid limit 10;

 salesid | pricepaid | commission | difference |     to_char
---------+-----------+------------+------------+-----------------
       1 |    728.00 |     109.20 |     618.80 |           dcxix
       2 |     76.00 |      11.40 |      64.60 |             lxv
       3 |    350.00 |      52.50 |     297.50 |        ccxcviii
       4 |    175.00 |      26.25 |     148.75 |           cxlix
       5 |    154.00 |      23.10 |     130.90 |           cxxxi
       6 |    394.00 |      59.10 |     334.90 |         cccxxxv
       7 |    788.00 |     118.20 |     669.80 |           dclxx
       8 |    197.00 |      29.55 |     167.45 |          clxvii
       9 |    591.00 |      88.65 |     502.35 |             dii
      10 |     65.00 |       9.75 |      55.25 |              lv
```

O seguinte exemplo adiciona o cifrão aos valores da diferença exibidos na coluna to_char:

```
select salesid, pricepaid, commission, (pricepaid - commission)
as difference, to_char(pricepaid - commission, 'l99999D99') from sales
group by sales.pricepaid, sales.commission, salesid
order by salesid limit 10;
```

```
salesid | pricepaid | commission | difference |  to_char
--------+-----------+------------+------------+------------
      1 |    728.00 |     109.20 |     618.80 | $   618.80
      2 |     76.00 |      11.40 |      64.60 | $    64.60
      3 |    350.00 |      52.50 |     297.50 | $   297.50
      4 |    175.00 |      26.25 |     148.75 | $   148.75
      5 |    154.00 |      23.10 |     130.90 | $   130.90
      6 |    394.00 |      59.10 |     334.90 | $   334.90
      7 |    788.00 |     118.20 |     669.80 | $   669.80
      8 |    197.00 |      29.55 |     167.45 | $   167.45
      9 |    591.00 |      88.65 |     502.35 | $   502.35
     10 |     65.00 |       9.75 |      55.25 | $    55.25
```

O seguinte exemplo lista o século em que cada venda foi efetuada.

```
select salesid, saletime, to_char(saletime, 'cc') from sales
order by salesid limit 10;

 salesid |      saletime       | to_char
---------+---------------------+---------
       1 | 2008-02-18 02:36:48 | 21
       2 | 2008-06-06 05:00:16 | 21
       3 | 2008-06-06 08:26:17 | 21
       4 | 2008-06-09 08:38:52 | 21
       5 | 2008-08-31 09:17:02 | 21
       6 | 2008-07-16 11:59:24 | 21
       7 | 2008-06-26 12:56:06 | 21
       8 | 2008-07-10 02:12:36 | 21
       9 | 2008-07-22 02:23:17 | 21
      10 | 2008-08-06 02:51:55 | 21
```

O seguinte exemplo converte cada valor de STARTTIME na tabela EVENT em uma string que consiste em horas, minutos, segundos e fuso horário:

```
select to_char(starttime, 'HH12:MI:SS TZ')
from event where eventid between 1 and 5
order by eventid;

to_char
----------
02:30:00 UTC
08:00:00 UTC
02:30:00 UTC
```

```
02:30:00 UTC
07:00:00 UTC
```

O seguinte exemplo exibe a formatação para segundos, milissegundos e microssegundos.

```
select sysdate,
to_char(sysdate, 'HH24:MI:SS') as seconds,
to_char(sysdate, 'HH24:MI:SS.MS') as milliseconds,
to_char(sysdate, 'HH24:MI:SS:US') as microseconds;

timestamp           | seconds  | milliseconds | microseconds
--------------------+----------+--------------+----------------
2015-04-10 18:45:09 | 18:45:09 | 18:45:09.325 | 18:45:09:325143
```

Função TO_DATE

TO_DATE converte uma data representada em uma string de caracteres para um tipo de dados DATE.

> **Note**
>
> TO_DATE não aceita strings de formato com Q (número do trimestre).

Sintaxe

```
TO_DATE(string, format)
```

```
TO_DATE(string, format, is_strict)
```

Argumentos

string

Uma string a ser convertida.

formato

Um literal de string que define o formato da string de entrada, em termos de suas partes de data. Para obter uma lista dos formatos válidos de dia, mês e ano, consulte Strings de formato datetime.

is_strict

Um valor booleano opcional que especifica se um erro é retornado se um valor de data de entrada estiver fora do intervalo. Quando is_strict é definido como TRUE, um erro será retornado se houver um valor fora do intervalo. Quando is_strict é definido como FALSE, que é o padrão, então os valores de estouro são aceitos.

Tipo de retorno

TO_DATE retorna uma DATE, dependendo do valor do format.

Se ocorrer falha na conversão no formato, um erro será gerado.

Exemplos

A instrução SQL a seguir converte a data `02 Oct 2001` em um tipo de dados de data.

```
select to_date('02 Oct 2001', 'DD Mon YYYY');

to_date
------------
2001-10-02
(1 row)
```

A instrução SQL a seguir converte a string `20010631` em uma data.

```
select to_date('20010631', 'YYYYMMDD', FALSE);
```

O resultado é 1 de julho de 2001, porque há apenas 30 dias em junho.

```
to_date
------------
2001-07-01
```

A seguinte instrução SQL converte a string `20010631` em uma data:

```
to_date('20010631', 'YYYYMMDD', TRUE);
```

O resultado é um erro porque há apenas 30 dias em junho.

```
ERROR:  date/time field date value out of range: 2001-6-31
```

TO_NUMBER

TO_NUMBER converte uma string em um valor numérico (decimal).

> **Note**
>
> Recomendamos que você use FM em sua string de formato para suprimir espaços em branco e zeros de preenchimento. Para obter uma lista dos formatos válidos, consulte Strings de formato numérico.

Sintaxe

```
to_number(string, format)
```

Argumentos

string

String a ser convertida. O formato deve ser um valor literal.

formato

O segundo argumento é uma string de formato que indica como a string de caracteres deve ser analisada para criar o valor numérico. Por exemplo, o formato `'FM99D999'` especifica que a string a ser convertida consiste em cinco dígitos com o ponto decimal na terceira posição. Por exemplo, `to_number('12.345','FM99D999')` retorna `12.345` como um valor numérico. Para obter uma lista dos formatos válidos, consulte Strings de formato numérico.

Tipo de retorno

TO_NUMBER retorna um número DECIMAL.

Se ocorrer falha na conversão no formato, um erro será gerado.

Exemplos

O exemplo a seguir converte a string `12,454.8-` em um número:

```
select to_number('12,454.8-', 'FM99G999D9S');

to_number
-----------
-12454.8
```

O exemplo a seguir converte a string `$ 12,454.88` em um número:

```
select to_number('$ 12,454.88', 'FML99G999D99');

to_number
-----------
12454.88
```

O exemplo a seguir converte a string `$ 2,012,454.88` em um número:

```
select to_number('$ 2,012,454.88', 'FML9,999,999.99');

to_number
-----------
2012454.88
```

Função TRY_CAST

Em comparação com a função CAST, TRY_CAST primeiro tenta converter a expressão no tipo especificado. Se a conversão falhar devido a erros, a operação exibirá null. Se uma conversão não for explicitamente permitida, a operação exibirá um erro. É possível encontrar a lista de conversões possíveis nas notas de uso abaixo. Por exemplo, tentar converter um booliano em um carimbo de data/hora não é permitido.

Sintaxe

```
TRY_CAST(expression AS type)
```

Argumentos

expressão

Uma expressão que avalia para um ou mais valores, tal como um nome de coluna ou um literal. A conversão de valores nulos retorna nulos. A expressão não pode conter strings em branco ou vazias.

type

Um dos tipos de dados aceitos. Para ver uma lista completa de tipos de recurso, consulte Tipos de dados. Para ver a lista de pares de tipos de dados de origem e de destino aceitos, consulte as observações de uso abaixo.

Tipo de retorno

TRY_CAST exibe um valor do tipo de dados especificado pelo argumento type. Se a conversão falhar, a operação exibirá null.

Observações de uso

Veja a lista de pares de tipos de dados de origem e de destino aceitos no Amazon Redshift para TRY_CAST.

BOOL

SMALLINT, INT, BIGINT, SUPER

SMALLINT

BOOL, INT, BIGINT, DECIMAL, REAL, FLOAT, BPCHAR, TEXT, VARCHAR, SUPER

INT

BOOL, SMALLINT, BIGINT, DECIMAL, REAL, FLOAT, BPCHAR, TEXT, VARCHAR, SUPER

BIGINT

BOOL, SMALLINT, INT, DECIMAL, REAL, FLOAT, BPCHAR, TEXT, VARCHAR, SUPER

DECIMAL

SMALLINT, INT, BIGINT, REAL, FLOAT, BPCHAR, TEXT, VARCHAR, SUPER

REAL

SMALLINT, INT, BIGINT, DECIMAL, FLOAT, BPCHAR, TEXT, VARCHAR, SUPER

FLOAT

SMALLINT, INT, BIGINT, DECIMAL, REAL, BPCHAR, TEXT, VARCHAR, SUPER

BPCHAR

SMALLINT, INT, BIGINT, DECIMAL, REAL, FLOAT, TEXT, VARCHAR, TIMESTAMP, TIMESTAMPTZ, DATE, TIME, TIMETZ, SUPER

TEXT

SMALLINT, INT, BIGINT, DECIMAL, REAL, FLOAT, BPCHAR, VARCHAR, TIMESTAMP, TIMESTAMPTZ, DATE, TIME, TIMETZ, SUPER

VARCHAR

SMALLINT, INT, BIGINT, DECIMAL, REAL, FLOAT, BPCHAR, TEXT, TIMESTAMP, TIMESTAMPTZ, DATE, TIME, TIMETZ, SUPER

TIMESTAMP

BPCHAR, TEXT, VARCHAR, TIMESTAMPTZ, DATE, TIME

TIMESTAMPTZ

BPCHAR, TEXT, VARCHAR, TIMESTAMP, DATE, TIME, TIMETZ

DATE

BPCHAR, TEXT, VARCHAR, TIMESTAMP, TIMESTAMPTZ

TIME

BPCHAR, TEXT, VARCHAR

TIMETZ

BPCHAR, TEXT, VARCHAR

SUPER

SUPER pode ser convertido em qualquer outro tipo de dados, com exceção de DATE, TIME, TIMETZ, TIMESTAMP e TIMESTAMPTZ.

Exemplos

O exemplo a seguir converte STRING em um INTEGER.

```
SELECT TRY_CAST('123' AS INT);

int
----
123
```

O exemplo a seguir exibe null. A conversão de STRING em INTEGER é permitida para que TRY_CAST não exiba um erro, mas “foo” não é um número inteiro, então a função exibe null.

```
SELECT TRY_CAST('foo' AS INT)
```

O exemplo a seguir exibe um erro, pois a conversão de BOOLEAN em TIMESTAMP não é permitida.

```
SELECT TRY_CAST(true as timestamp);
```

Como TRY_CAST exibirá nulo em vez de exibir imediatamente um erro se a conversão falhar, é possível usar TRY_CAST para filtrar dados inválidos. Pense no exemplo a seguir, em que uma linha inválida é filtrada devido a uma falha de conversão na coluna de idade para Akua Mansa.

```
CREATE TABLE IF NOT EXISTS student_data(
name VARCHAR(100) NOT NULL,
age VARCHAR(3) NOT NULL,
enrollment_date DATE NOT NULL);

INSERT INTO student_data (name, age, enrollment_date)
VALUES
('Alejandro Rosalez', '10', '01/01/2000'),
('Akua Mansa', 'Ten', '01/01/2000');

SELECT * FROM student_data WHERE TRY_CAST(age AS INT) IS NOT NULL;

--Akua is not returned.
 name              | age | enrollment_date
-------------------+-----+-----------------
 Alejandro Rosalez | 10  | 01/01/2000
```

Strings de formato datetime

Você pode encontrar uma referência para strings de formato datetime a seguir.

Os seguintes formatos de strings aplicam-se a funções como TO_CHAR. Essas strings podem conter separadores datetime (como '-', '/' ou ':') e os "dateparts" e "timeparts" a seguir.

Datepart ou timepart	Significado
AC ou A.C., DC ou D.C., a.c. ou ac, dc ou d.c.	Indicadores de era maiúsculos e minúsculos
CC	Número de século com dois dígitos

Datepart ou timepart	Significado
YYYY, YYY, YY, Y	Número de ano com 4 dígitos, 3 dígitos, 2 dígitos, 1 dígito
Y,YYY	Número de ano de 4 dígitos com vírgula
IYYY, IYY, IY, I	Número de ano da Organização Internacional de normalização (ISO) de 4 dígitos, 3 dígitos, 2 dígitos, 1 dígito
Q	Número do trimestre (1 a 4)
MÊS, Mês, mês	Nome do mês (maiúsculas, maiúsculas e minúsculas, minúsculas, cercado por espaços com até 9 caracteres)
MÊS, Mês, mês	Nome do mês abreviado (letras maiúscula s, letras maiúsculas e minúsculas, letras minúsculas, com preenchimento de até três caracteres)
MM	Número do mês (01-12)
RM, rm	Número do mês em algarismos romanos (I–XII, sendo I janeiro, maiúscula ou minúscula)
W	Semana do mês (1–5; a primeira semana começa no primeiro dia do mês).
WW	Número da semana do ano (1–53; a primeira semana começa no primeiro dia do ano).
IW	Número ISO da semana do ano (a primeira quinta-feira do novo ano é na semana 1.)
DIA, Dia, dia	Nome do dia (maiúsculas, maiúsculas e minúsculas, minúsculas, cercado por espaços com até 9 caracteres)

Datepart ou timepart	Significado
DY, Dy, dy	Nome do dia abreviado (maiúsculas, maiúsculas e minúsculas, minúsculas, cercado por espaços com até 3 caracteres)
DDD	Dia do ano (001-366)
IDDD	Dia do ano numerado por semanas da ISO 8601 (001-371; o dia 1 do ano é a segunda-feira da primeira semana da ISO)
DD	Dia do mês como um número (01–31)
D	Dia da semana (1–7; domingo é 1) **Note** A datepart D comporta-se de forma diferente da datepart de dia da semana (DOW) usada para funções de datetime DATE_PART e EXTRACT. DOW baseia-se no números inteiros 0–6, onde domingo é 0. Para obter mais informações, consulte Partes da data para funções de data ou de timestamp.
ID	Dia da semana da ISO 8601, segunda-feira (1) a domingo (7)
J	Dia juliano (dias desde 1º de janeiro de 4.712 AC)
HH24	Hora (relógio de 24 horas, 00–23)
HH ou HH12	Hora (relógio de 12 horas, 01–12)
MI	Minutos (00–59)

Datepart ou timepart	Significado
SS	Segundos (00–59)
MS	Milissegundos (,000)
US	Microssegundos (,000000)
AM ou PM, A.M. ou P.M., a.m. ou p.m., am ou pm	Indicadores meridianos maiúsculos e minúsculos (para relógio de 12 horas)
TZ, tz	Abreviação do fuso horário em maiúsculas e minúsculas; válida somente para TIMESTAMP TZ
OF	Deslocamento do UTC; válido somente para TIMESTAMPTZ

> **Note**
>
> É necessário colocar os separadores de data e hora (como '-', '/' ou ':') entre aspas simples, mas é necessário colocar os "dateparts" e "os timeparts" listados na tabela anterior entre aspas duplas.

Exemplos

Para obter exemplos de formatação de datas como strings, consulte TO_CHAR.

Strings de formato numérico

Você pode encontrar uma referência para strings de formato númerico a seguir.

A string de formato a seguir se aplica a funções como TO_NUMBER e TO_CHAR.

- Para obter exemplos de strings de formatação como números, consulte TO_NUMBER.
- Para obter exemplos de números de formatação como strings, consulte TO_CHAR.

Formato	Descrição
9	Valor numérico com o número especificado de dígitos.
0	Valor numérico com zeros iniciais.
. (ponto final), D	Ponto decimal.
, (vírgula)	Separador de milhares.
CC	Código de século. Por exemplo, o século 21 começou em 2001-01-01 (compatível somente com TO_CHAR).
FM	Modo de preenchimento. Excluir espaços em branco e zeros.
PR	Valor negativo entre colchetes angulares.
S	Sinal ancorado a um número.
L	Símbolo de cifrão na posição especificada.
G	Separador de grupo.
MI	Sinal de menos na posição especificada para números menores que 0.
PL	Sinal de mais na posição especificada para números maiores que 0.
SG	Sinal de mais ou menos na posição especific ada.
RN	Numeral romano entre 1 e 3.999 (compatível somente com TO_CHAR).

Formato	Descrição
TH ou th	Sufixo de número ordinal. Não converte números ou valores fracionários menores que zero.

Caracteres de formatação de estilo Teradata para dados numéricos

A seguir, você pode encontrar como as funções TEXT_TO_INT_ALT e TEXT_TO_NUMERIC_ALT interpretam os caracteres na string da expressão de entrada. Você também pode encontrar uma lista de caracteres que podem ser especificados na frase de format. Além disso, você pode encontrar uma descrição das diferenças entre a formatação do estilo Teradata e o Amazon Redshift para a opção de format.

Formato	Descrição
G	Não aceito como separador de grupo na string da expressão de entrada. Não é possível especificar esse caractere na frase de format.
D	Símbolo Radix. É possível especificar esse caractere na frase de format. Esse caractere é equivalente ao “.” (ponto final). O símbolo Radix não pode aparecer em uma frase de format que contenha espaços ou um destes caracteres: • . (ponto final) • S (“s” maiúsculo) • V (“v” maiúsculo)
/ , : %	Caracteres de inserção / (barra), vírgula (,), : (dois-pontos) e % (sinal de porcentagem). Não é possível especificar esses caracteres na frase de format.

Formato	Descrição
	O Amazon Redshift ignora esses caracteres na string da expressão de entrada.
.	Ponto final como um caractere de radix, que é um separador decimal. Este caractere não pode aparecer em uma frase de format que contenha qualquer um destes caracteres: • D (“d” maiúsculo) • S (“s” maiúsculo) • V (“v” maiúsculo)
B	Não é possível incluir o caractere de espaço em branco (B) na frase de format. Na string da expressão de entrada, espaços à esquerda e à direita são ignorados e espaços entre dígitos não são permitidos.
+ -	Não é possível incluir o sinal de mais (+) ou de menos (-) na frase de format. No entanto, o sinal de mais (+) e de menos (-) são analisados implicitamente como parte do valor numérico se eles aparecem na string da expressão de entrada.
V	Indicador de posição do separador decimal. Este caractere não pode aparecer em uma frase de format que contenha qualquer um destes caracteres: • D (“d” maiúsculo) • . (ponto final)

Formato	Descrição
Z	Dígito decimal suprimido de zero. O Amazon Redshift corta zeros iniciais. O caractere Z não pode seguir um caractere 9. O caractere Z deve estar à esquerda do caractere de radix se a parte de fração contiver o caractere 9.
9	Dígito decimal.
CHAR(n)	Para esse formato, é possível especificar o seguinte: • CHAR consiste em Z ou 9 caracteres. O Amazon Redshift não oferece suporte a + (mais) ou - (menos) no valor CHAR. • n é uma constante inteira, I, ou F. Para I, este é o número de caracteres necessári os para exibir a parte inteira de dados numéricos ou inteiros. Para F, esse é o número de caracteres necessários para exibir a parte fracionada dos dados numéricos.
-	Caractere de hífen (-). Não é possível especificar esse caractere na frase de format. O Amazon Redshift ignora esse caractere na string da expressão de entrada.

Formato	Descrição
S	Decimal com zona assinada. O caractere S deve seguir o último dígito decimal na frase de format. O último caractere da string da expressão de entrada e a conversão numérica correspondente estão listadas em Caracteres de formatação de dados para formato de dados numéricos decimais da zona assinada, estilo Teradata. Este caractere não pode aparecer em uma frase de format que contenha qualquer um destes caracteres: • + (sinal de adição) • . (ponto final) • D (“d” maiúsculo) • Z (“z” maiúsculo) • F (“f” maiúsculo) • E (“e” maiúsculo)
E	Notação exponencial. A string da expressão de entrada pode incluir o caractere expoente. Você não pode especificar E como um caractere expoente na frase de format.
FN9	Não aceito no Amazon Redshift.
FNE	Não é suportado no Amazon Redshift.

Formato	Descrição
$, USD, US Dollars	Sinal de dólar ($), símbolo de moeda ISO (USD) e o nome da moeda US Dollars. O símbolo de moeda ISO USD e o nome da moeda US Dollars diferenciam maiúsculas e minúsculas. O Amazon Redshift oferece suporte apenas à moeda USD. A string da expressão de entrada pode incluir espaços entre o símbolo de moeda USD e o valor numérico, por exemplo “$ 123E2” ou “123E2 $”.
L	Símbolo de moeda. Este caractere de símbolo de moeda só pode aparecer uma vez na frase de format. Não é possível especificar caractere s de símbolo de moeda repetidos.
C	Símbolo de moeda ISO. Este caractere de símbolo de moeda só pode aparecer uma vez na frase de format. Não é possível especificar caracteres de símbolo de moeda repetidos.
N	Nome completo da moeda. Este caractere de símbolo de moeda só pode aparecer uma vez na frase de format. Não é possível especificar caracteres de símbolo de moeda repetidos.
U	Símbolo de moeda dupla. Não é possível especificar esse caractere na frase de format.
U	Símbolo de moeda ISO dupla. Não é possível especificar esse caractere na frase de format.
A	Nome completo da moeda dupla. Não é possível especificar esse caractere na frase de format.

Caracteres de formatação de dados para formato de dados numéricos decimais da zona assinada, estilo Teradata

Você pode usar os seguintes caracteres na frase de format das funções TEXT_TO_INT_ALT e TEXT_TO_NUMERIC_ALT para um valor decimal com zona assinada.

Último caractere da string de entrada	Conversão numérica
{ ou 0	n … 0
A ou 1	n … 1
B ou 2	n … 2
C ou 3	n … 3
D ou 4	n … 4
E ou 5	n … 5
F ou 6	n … 6
G ou 7	n … 7
H ou 8	n … 8
I ou 9	n … 9
}	-n … 0
J	-n … 1
K	-n … 2
L	-n … 3
M	-n … 4
N	-n … 5
U	-n … 6

Último caractere da string de entrada	Conversão numérica
P	-n … 7
Q	-n … 8
R	-n … 9

Perfis de data e hora

Nesta seção, você pode encontrar informações sobre as funções escalares de data e hora compatíveis com o Amazon Redshift.

Tópicos

- Resumo das funções de data e hora
- Funções de data e hora em transações
- Funções somente nó de liderança defasadas
- Operador + (Concatenação)
- Função ADD_MONTHS
- Função AT TIME ZONE
- Função CONVERT_TIMEZONE
- Função CURRENT_DATE
- Função DATE_CMP
- Função DATE_CMP_TIMESTAMP
- Função DATE_CMP_TIMESTAMPTZ
- Função DATEADD
- Função DATEDIFF
- Função DATE_PART
- Função DATE_PART_YEAR
- Função DATE_TRUNC
- Função EXTRACT
- Função GETDATE
- Função INTERVAL_CMP

- Função LAST_DAY
- Função MONTHS_BETWEEN
- Função NEXT_DAY
- Função SYSDATE
- Função TIMEOFDAY
- Função TIMESTAMP_CMP
- Função TIMESTAMP_CMP_DATE
- Função TIMESTAMP_CMP_TIMESTAMPTZ
- Função TIMESTAMPTZ_CMP
- Função TIMESTAMPTZ_CMP_DATE
- Função TIMESTAMPTZ_CMP_TIMESTAMP
- Função TIMEZONE
- Função TO_TIMESTAMP
- Função TRUNC
- Partes da data para funções de data ou de timestamp

Resumo das funções de data e hora

Função	Sintaxe	Retornos
Operador + (Concatenação) Concatena uma data para uma hora em ambos os lados do símbolo + e retorna um TIMESTAMP ou TIMESTAMPTZ.	date + time	`TIMESTAMP` ou `TIMESTAMPTZ`
ADD_MONTHS Adiciona o número especificado de meses a uma data ou timestamp.	ADD_MONTHS ({date|timestamp}, integer)	`TIMESTAMP`
AT TIME ZONE Especifica qual fuso horário a usar com uma expressão TIMESTAMP ou TIMESTAMPTZ.	AT TIME ZONE 'timezone'	`TIMESTAMP` ou

Função	Sintaxe	Retornos
		`TIMESTAMPZ`
CONVERT_TIMEZONE Converte um timestamp de um fuso horário para outro.	CONVERT_TIMEZONE (['timezone',] 'timezone', timestamp)	`TIMESTAMP`
CURRENT_DATE Retorna uma data no fuso horário da sessão atual (UTC por padrão) para o início da transação atual.	CURRENT_DATE	`DATE`
DATE_CMP Compara duas datas e exibe `0` se as datas forem idênticas, 1 se date1 for maior e `-1` se date2 for maior.	DATE_CMP (date1, date2)	`INTEGER`
DATE_CMP_TIMESTAMP Compara uma data a uma hora e exibe `0` se os valores forem idênticos, 1 se date for maior e `-1` se timestamp for maior.	DATE_CMP_TIMESTAMP (date, timestamp)	`INTEGER`
DATE_CMP_TIMESTAMPTZ Compara uma data e um carimbo de data/hora com fuso horário e exibe `0` se os valores forem idênticos, 1 se date for maior e `-1` se timestamptz for maior.	DATE_CMP_TIMESTAMPTZ (date, timestamptz)	`INTEGER`
DATE_PART_YEAR Extrai o ano de uma data.	DATE_PART_YEAR (date)	`INTEGER`

Função	Sintaxe	Retornos
DATEADD Incrementa uma data ou hora com um intervalo especificado.	DATEADD (datepart, interval, {date|time|timetz|timestamp})	TIMESTAMP ou TIME ou TIMETZ
DATEDIFF Retorna a diferença entre as duas datas ou horas para determinada parte da data, tal como um dia ou mês.	DATEDIFF (datepart, {date|time|timetz|timestamp}, {date|time|timetz|timestamp})	BIGINT
DATE_PART Extrai um valor da parte de data de uma data ou hora.	DATE_PART (datepart, {date|timestamp})	DOUBLE
DATE_TRUNC Trunca um timestamp com base em uma parte da data.	DATE_TRUNC ('datepart', timestamp)	TIMESTAMP
EXTRACT Extrai uma parte da data ou hora de um timestamp, timestamptz, time ou timetz.	EXTRACT (datepart FROM source)	INTEGER or DOUBLE
GETDATE Retorna a atual data e hora no fuso horário da sessão atual (UTC por padrão). Os parênteses são necessários.	GETDATE()	TIMESTAMP
INTERVAL_CMP Compara dois intervalos e exibe 0 se os intervalos forem iguais, 1 se interval1 for maior e -1 se interval2 for maior.	INTERVAL_CMP (interval1, interval2)	INTEGER

Função	Sintaxe	Retornos
LAST_DAY Retorna a data do último dia do mês que contém data.	LAST_DAY(date)	DATE
MONTHS_BETWEEN Retorna o número de meses entre duas datas.	MONTHS_BETWEEN (date, date)	FLOAT8
NEXT_DAY Retorna a data da primeira instância do dia que é mais recente que a data.	NEXT_DAY (date, day)	DATE
SYSDATE Retorna a data e hora em UTC para o início da transação atual.	SYSDATE	TIMESTAMP
TIMEOFDAY Retorna o atual dia da semana, data e hora no fuso horário da sessão atual (UTC por padrão) com um valor de string.	TIMEOFDAY()	VARCHAR
TIMESTAMP_CMP Compara dois carimbos de data/hora e exibe 0 se os carimbos forem iguais, 1 se timestamp1 for maior e -1 se timestamp2 for maior.	TIMESTAMP_CMP (timestamp1, timestamp2)	INTEGER
TIMESTAMP_CMP_DATE Compara um carimbo de data/hora a uma data e exibe 0 se os valores forem idênticos, 1 se timestamp for maior e -1 se date for maior.	TIMESTAMP_CMP_DATE (timestamp, date)	INTEGER

Função	Sintaxe	Retornos
TIMESTAMP_CMP_TIMESTAMPTZ Compara um carimbo de data/hora a um carimbo de data/hora com fuso horário e exibe `0` se os valores forem iguais, 1 se timestamp for maior e `-1` se timestamptz for maior.	TIMESTAMP_CMP_TIMESTAMPTZ (timestamp, timestamptz)	`INTEGER`
TIMESTAMPTZ_CMP Compara dois carimbos de data/hora com valores de fuso horário e exibe `0` se os valores forem iguais, 1 se timestamptz1 for maior e `-1` se timestamptz2 for maior.	TIMESTAMPTZ_CMP (timestamptz1, timestamptz2)	`INTEGER`
TIMESTAMPTZ_CMP_DATE Compara o valor de um carimbo de data/hora com fuso horário a uma data e exibe `0` se os valores forem iguais, 1 se timestamptz for maior e `-1` se date for maior.	TIMESTAMPTZ_CMP_DATE (timestamptz, date)	`INTEGER`
TIMESTAMPTZ_CMP_TIMESTAMP Compara um carimbo de data/hora com fuso horário a um carimbo de data/hora e exibe `0` se os valores forem iguais, 1 se timestamptz for maior e `-1` se timestamp for maior.	TIMESTAMPTZ_CMP_TIMESTAMP (timestamptz, timestamp)	`INTEGER`
TIMEZONE Retorna um timestamp para o valor de fuso horário e timestamp especificados.	TIMEZONE ('timezone' { timestamp | timestamptz)	`TIMESTAMP` ou `TIMESTAMPTZ`

Função	Sintaxe	Retornos
TO_TIMESTAMP Retorna um timestamp com fuso horário para o formato de timestamp e fuso horário especific ados.	TO_TIMESTAMP ('timestamp', 'format')	`TIMESTAMP TZ`
TRUNC Trunca um timestamp e retorna uma data.	TRUNC(timestamp)	`DATE`

> **Note**
>
> Segundos intercalados não são considerados em cálculos tempo decorrido.

Funções de data e hora em transações

Quando você executa as seguintes funções em um bloco de transação (BEGIN… END), a função retorna a data ou hora de início da transação atual, não o início da instrução atual.

- SYSDATE
- TIMESTAMP
- CURRENT_DATE

As seguintes funções sempre retornam a data ou hora de início da atual instrução, mesmo quando estiverem em um bloco de transação.

- GETDATE
- TIMEOFDAY

Funções somente nó de liderança defasadas

As seguintes funções de data estão defasadas, pois executam somente no nó de liderança. Para obter mais informações, consulte Função de apenas nó líder.

- AGE. Use Função DATEDIFF em vez disso.
- CURRENT_TIME. Use Função GETDATE ou SYSDATE em vez disso.
- CURRENT_TIMESTAMP. Use Função GETDATE ou SYSDATE em vez disso.
- LOCALTIME. Use Função GETDATE ou SYSDATE em vez disso.
- LOCALTIMESTAMP. Use Função GETDATE ou SYSDATE em vez disso.
- ISFINITE
- NOW. Use Função GETDATE ou SYSDATE em vez disso. Se você usar a função NOW em uma visão materializada, ela será definida como o carimbo de data/hora da criação da visão materializada, em vez do carimbo de data/hora atual.

Operador + (Concatenação)

Concatena um DATE para um TIME ou TIMETZ em ambos os lados do símbolo + e retorna um TIMESTAMP ou TIMESTAMPTZ.

Sintaxe

```
date + {time | timetz}
```

A ordem dos argumentos pode ser invertida. Por exemplo, hora + data.

Argumentos

date

Uma coluna de tipo de dados `DATE` ou uma expressão que é avaliada implicitamente como um tipo `DATE`.

hora

Uma coluna de tipo de dados `TIME` ou uma expressão que é avaliada implicitamente como um tipo `TIME`.

timetz

Uma coluna de tipo de dados `TIMETZ` ou uma expressão que é avaliada implicitamente como um tipo `TIMETZ`.

Tipo de retorno

TIMESTAMP se a entrada for date + time.

TIMESTAMPTZ se a entrada for date + timetz.

Exemplos

Exemplo de configuração

Para configurar as tabelas TIME_TEST e TIMETZ_TEST utilizadas nos exemplos, use o comando a seguir.

```
create table time_test(time_val time);

insert into time_test values
('20:00:00'),
('00:00:00.5550'),
('00:58:00');

create table timetz_test(timetz_val timetz);

insert into timetz_test values
('04:00:00+00'),
('00:00:00.5550+00'),
('05:58:00+00');
```

Exemplos com uma coluna time

O TIME_TEST da tabela a seguir tem uma coluna TIME_VAL (tipo TIME) com três valores inseridos.

```
select time_val from time_test;

time_val
--------------------
20:00:00
00:00:00.5550
00:58:00
```

O exemplo a seguir concatena um literal de data e uma coluna TIME_VAL.

```
select date '2000-01-02' + time_val as ts from time_test;
```

```
ts
---------------------
2000-01-02 20:00:00
2000-01-02 00:00:00.5550
2000-01-02 00:58:00
```

O exemplo a seguir concatena um literal de data e um literal de tempo.

```
select date '2000-01-01' + time '20:00:00' as ts;

         ts
---------------------
 2000-01-01 20:00:00
```

O exemplo a seguir concatena um literal de hora e um literal de data.

```
select time '20:00:00' + date '2000-01-01' as ts;

         ts
---------------------
 2000-01-01 20:00:00
```

Exemplos com uma coluna TIMETZ

O TIMETZ_TEST da tabela de exemplo a seguir tem uma coluna TIMETZ_VAL (tipo TIMETZ) com três valores inseridos.

```
select timetz_val from timetz_test;

timetz_val
------------------
04:00:00+00
00:00:00.5550+00
05:58:00+00
```

O exemplo a seguir concatena um literal de data e uma coluna TIMETZ_VAL.

```
select date '2000-01-01' + timetz_val as ts from timetz_test;
ts
```

```
---------------------
2000-01-01 04:00:00+00
2000-01-01 00:00:00.5550+00
2000-01-01 05:58:00+00
```

O exemplo a seguir concatena uma coluna TIMETZ_VAL e um literal de data.

```
select timetz_val + date '2000-01-01' as ts from timetz_test;
ts
---------------------
2000-01-01 04:00:00+00
2000-01-01 00:00:00.5550+00
2000-01-01 05:58:00+00
```

O exemplo a seguir concatena um literal DATE e um literal TIMETZ. O exemplo retorna um TIMESTAMPTZ que está no fuso horário UTC por padrão. O UTC está oito horas à frente do PST, então o resultado está oito horas antes do horário da entrada.

```
select date '2000-01-01' + timetz '20:00:00 PST' as ts;

           ts
------------------------
 2000-01-02 04:00:00+00
```

Função ADD_MONTHS

ADD_MONTHS adiciona o número especificado de meses a um valor ou expressão de data ou timestamp. A função DATEADD oferece funcionalidade semelhante.

Sintaxe

```
ADD_MONTHS( {date | timestamp}, integer)
```

Argumentos

date | timestamp

Uma coluna de tipo de dados DATE ou TIMESTAMP ou uma expressão que é avaliada implicitamente como um tipo DATE ou TIMESTAMP. Se a data for o último dia do mês ou se o mês

resultante for mais curto, a função retorna o último dia do mês nos resultados. Para outras datas, o resultado contém o mesmo número de dia que a expressão de data.

integer

Um valor de de tipo de dados `INTEGER`. Use um número negativo para subtrair meses de datas.

Tipo de retorno

TIMESTAMP

Exemplos

A seguinte consulta usa a função de ADD_MONTHS dentro de uma função TRUNC. A função TRUNC remove o horário do dia dos resultados de ADD_MONTHS. A função ADD_MONTHS adiciona 12 meses a cada valor da coluna CALDATE. Os valores na coluna CALDATE são datas.

```
select distinct trunc(add_months(caldate, 12)) as calplus12,
trunc(caldate) as cal
from date
order by 1 asc;

 calplus12  |    cal
------------+------------
 2009-01-01 | 2008-01-01
 2009-01-02 | 2008-01-02
 2009-01-03 | 2008-01-03
...
(365 rows)
```

O exemplo a seguir usa a função ADD_MONTHS para adicionar um mês a um carimbo de data/hora.

```
select add_months('2008-01-01 05:07:30', 1);

add_months
---------------------
2008-02-01 05:07:30
```

Os seguintes exemplos demonstram o comportamento quando a função ADD_MONTHS opera em datas com meses que têm diferentes número de dias. Este exemplo mostra como a função lida com a adição de um mês a 31 de março e a adição de um mês a 30 de abril. Abril tem 30 dias, então

adicionar 1 mês a 31 de março resulta em 30 de abril. Maio tem 31 dias, então adicionar 1 mês a 30 de abril resulta em 31 de maio.

```
select add_months('2008-03-31',1);

add_months
---------------------
2008-04-30 00:00:00

select add_months('2008-04-30',1);

add_months
---------------------
2008-05-31 00:00:00
```

Função AT TIME ZONE

AT TIME ZONE especifica qual fuso horário a usar com uma expressão TIMESTAMP ou TIMESTAMPTZ.

Sintaxe

```
AT TIME ZONE 'timezone'
```

Argumentos

timezone

O TIMEZONE do valor de retorno. O fuso horário pode ser especificado como um nome de fuso horário (como **'Africa/Kampala'** ou **'Singapore'**) ou como uma abreviação de fuso horário (como **'UTC'** ou **'PDT'**).

Para visualizar uma lista de nomes de fusos horários compatíveis, execute o comando a seguir.

```
select pg_timezone_names();
```

Para visualizar uma lista de abreviações de fusos horários compatíveis, execute o comando a seguir.

```
select pg_timezone_abbrevs();
```

Para ter mais informações e exemplos, consulte Observações quanto ao uso de fuso horário.

Tipo de retorno

TIMESTAMPTZ quando usado com uma expressão TIMESTAMP. TIMESTAMP quando usado com uma expressão TIMESTAMPTZ.

Exemplos

O exemplo a seguir converte um valor de carimbo de data/hora sem fuso horário e o interpreta como o fuso MST (UTC+7 em POSIX). O exemplo retorna um valor do tipo de dados TIMESTAMPTZ para o fuso horário UTC. Se configurar o fuso horário padrão como um fuso horário diferente de UTC, você poderá ver um resultado diferente.

```
SELECT TIMESTAMP '2001-02-16 20:38:40' AT TIME ZONE 'MST';

timezone
------------------------
2001-02-17 03:38:40+00
```

O exemplo a seguir pega um time stamp de entrada com um valor de fuso horário em que o fuso horário especificado é EST (UTC+5 em POSIX) e o converte para MST (UTC+7 em POSIX). O exemplo retorna um valor do tipo de dados TIMESTAMP.

```
SELECT TIMESTAMPTZ '2001-02-16 20:38:40-05' AT TIME ZONE 'MST';

timezone
------------------------
2001-02-16 18:38:40
```

Função CONVERT_TIMEZONE

CONVERT_TIMEZONE converte um timestamp de um fuso horário para outro. A função se ajusta automaticamente para o horário de verão.

Sintaxe

```
CONVERT_TIMEZONE( ['source_timezone',] 'target_timezone', 'timestamp')
```

Argumentos

source_timezone

(Opcional) O fuso horário do timestamp atual. O padrão é UTC. Para obter mais informações, consulte Observações quanto ao uso de fuso horário.

target_timezone

O fuso horário do novo timestamp. Para obter mais informações, consulte Observações quanto ao uso de fuso horário.

timestamp

Uma coluna de timestamp ou uma expressão que converta implicitamente para um timestamp.

Tipo de retorno

TIMESTAMP

Observações quanto ao uso de fuso horário

source_timezone ou target_timezone pode ser especificado como o nome do fuso horário (tal como “África/Kampala” ou “Singapura”) ou como uma abreviatura de fuso horário (tal como “UTC” ou “PDT”). Você não precisa converter nomes de fuso horário em nomes ou abreviaturas em abreviaturas. Por exemplo, você pode escolher um carimbo de data e hora do nome de fuso horário de origem “Singapura” e convertê-lo em um carimbo de data e hora na abreviatura de fuso horário “PDT”.

> **Note**
>
> Os resultados da utilização de um nome de fuso horário ou de uma abreviatura de fuso horário podem ser diferentes devido ao horário sazonal local, como o horário de verão.

Usar um nome de fuso horário

Para visualizar uma lista completa e atualizada de nomes de fuso horário, execute o comando a seguir.

```
select pg_timezone_names();
```

Cada linha contém uma string separada por vírgula com o nome do fuso horário, abreviatura, deslocamento do UTC e indicador se o fuso horário segue o horário de verão (`t` ou `f`). Por exemplo, o trecho a seguir mostra duas linhas resultantes. A primeira linha é o fuso horário `Antarctica/South Pole`, abreviatura `NZDT`, com deslocamento `13:00:00` do UTC e `f` para indicar que não segue o horário de verão. A segunda linha é o fuso horário `Europe/Paris`, abreviatura `CET`, com deslocamento `01:00:00` do UTC e `f` para indicar que segue o horário de verão.

```
pg_timezone_names
------------------
(Antarctica/South_Pole,NZDT,13:00:00,t)
(Europe/Paris,CET,01:00:00,f)
```

Execute a instrução SQL para obter a lista inteira e encontrar um nome de fuso horário. Aproximadamente 600 linhas são retornadas. Embora alguns dos nomes de fuso horário retornados sejam iniciais em maiúsculas ou acrônimos (por exemplo, GB, PRC, ROK), a função CONVERT_TIMEZONE os trata como nomes de fuso horário, e não abreviações.

Se você especificar um fuso horário usando um nome de fuso horário, CONVERT_TIMEZONE fará o ajuste automático para o horário de verão (DST) ou qualquer outro protocolo sazonal local, tal como horário de verão, horário padrão ou horário de inverno, que esteja em vigor durante a data e hora especificadas pelo “carimbo de data/hora”. Por exemplo, “Europa/Londres” representa UTC no inverno e adiciona uma hora no verão. Observe que o Amazon Redshift usa o banco de dados de fuso horário IANA como fonte autorizada de especificação de fuso horário.

Usar uma abreviação de fuso horário

Para visualizar uma lista completa e atualizada de abreviaturas de fuso horário, execute o comando a seguir.

```
select pg_timezone_abbrevs();
```

Os resultados contêm uma string separada por vírgulas com a abreviatura do fuso horário, o deslocamento do UTC e o indicador que informa se o fuso horário segue o horário de verão (`t` ou `f`). Por exemplo, o trecho a seguir mostra duas linhas resultantes. A primeira linha contém a abreviatura de Pacific Daylight Time, `PDT`, com um deslocamento de `-07:00:00` do UTC e `t` para indicar que segue o horário de verão. A segunda linha contém a abreviatura de Pacific Standard Time, `PST`, com um deslocamento de `-08:00:00` do UTC e `f` para indicar que não segue o horário de verão.

```
pg_timezone_abbrevs
--------------------
(PDT,-07:00:00,t)
(PST,-08:00:00,f)
```

Execute a instrução SQL para obter a lista completa e encontrar uma abreviatura com base em seu deslocamento e indicador de horário de verão. Aproximadamente 200 linhas são retornadas.

As abreviações de fuso horário representam um deslocamento fixo do UTC. Se você especificar um fuso horário usando uma abreviatura de fuso horário, CONVERT_TIMEZONE usará o deslocamento fixo do UTC e não fará o ajuste para nenhum protocolo sazonal local.

Usar o formato de estilo POSIX

Uma especificação de fuso horário no estilo POSIX tem o formato STDoffset ou STDoffsetDST, em que STD é uma abreviatura de fuso horário, offset é o desvio numérico em horas a oeste de UTC e DST é uma abreviatura opcional de fuso horário de verão. Supõe-se que o horário de verão seja uma hora a mais que o deslocamento fornecido.

Os formatos de fuso horário estilo POSIX usam deslocamentos positivos à Oeste de Greenwich, em contraste com a convenção ISO-8601, usando deslocamentos positivos à Leste de Greenwich.

A seguir, exemplos de fusos horários no estilo POSIX:

- PST8
- PST8PDT
- EST5
- EST5EDT

> **Note**
>
> O Amazon Redshift não valida especificações de fuso horário no estilo POSIX, portanto é possível definir o fuso horário com um valor inválido. Por exemplo, o seguinte comando não retorna um erro, mesmo que ele defina o fuso horário com um valor inválido.
>
> ```
> set timezone to ‘xxx36’;
> ```

Exemplos

Muitos dos exemplos usam a amostra de conjunto de dados TICKIT. Para obter mais informações, consulte Banco de dados de amostra.

O exemplo a seguir converte o valor de carimbo de data/hora do fuso horário UTC padrão em PST.

```
select convert_timezone('PST', '2008-08-21 07:23:54');

 convert_timezone
-----------------------
2008-08-20 23:23:54
```

O exemplo a seguir converte o valor do timestamp na coluna LISTTIME do fuso horário UTC padrão para PST. Embora o timestamp esteja no período de horário de verão, ele é convertido para o horário padrão, pois o fuso horário de destino é especificado como uma abreviação (PST).

```
select listtime, convert_timezone('PST', listtime) from listing
where listid = 16;

     listtime       |   convert_timezone
--------------------+-------------------
2008-08-24 09:36:12     2008-08-24 01:36:12
```

O seguinte exemplo converte uma coluna de fuso horário LISTTIME do fuso horário UTC padrão para o fuso horário EUA/Pacífico. A fuso horário de destino usa um nome de fuso horário e o timestamp está no horário de verão, portanto a função retorna o horário de verão.

```
select listtime, convert_timezone('US/Pacific', listtime) from listing
where listid = 16;

     listtime       |   convert_timezone
--------------------+---------------------
2008-08-24 09:36:12 | 2008-08-24 02:36:12
```

O exemplo a seguir converte uma string de timestamp de EST para PST:

```
select convert_timezone('EST', 'PST', '20080305 12:25:29');

 convert_timezone
-------------------
```

```
2008-03-05 09:25:29
```

O exemplo a seguir converte um timestamp para o horário padrão do Leste dos EUA, pois o fuso horário de destino usa um nome de fuso horário (America/New_York) e o timestamp está dentro do período de horário padrão.

```
select convert_timezone('America/New_York', '2013-02-01 08:00:00');

 convert_timezone
---------------------
2013-02-01 03:00:00
(1 row)
```

O seguinte exemplo converte o timestamp para o horário de verão do Leste dos EUA, pois o fuso horário de destino usa um nome de fuso horário (America/New_York) e o timestamp está dentro do período do horário de verão.

```
select convert_timezone('America/New_York', '2013-06-01 08:00:00');

 convert_timezone
---------------------
2013-06-01 04:00:00
(1 row)
```

O seguinte exemplo demonstra o uso de deslocamentos.

```
SELECT CONVERT_TIMEZONE('GMT','NEWZONE +2','2014-05-17 12:00:00') as newzone_plus_2,
CONVERT_TIMEZONE('GMT','NEWZONE-2:15','2014-05-17 12:00:00') as newzone_minus_2_15,
CONVERT_TIMEZONE('GMT','America/Los_Angeles+2','2014-05-17 12:00:00') as la_plus_2,
CONVERT_TIMEZONE('GMT','GMT+2','2014-05-17 12:00:00') as gmt_plus_2;

   newzone_plus_2    | newzone_minus_2_15  |      la_plus_2      |     gmt_plus_2
---------------------+---------------------+---------------------+---------------------
2014-05-17 10:00:00 | 2014-05-17 14:15:00 | 2014-05-17 10:00:00 | 2014-05-17 10:00:00
(1 row)
```

Função CURRENT_DATE

CURRENT_DATE retorna uma data no fuso horário da sessão atual (UTC por padrão) no formato padrão: AAAA-MM-DD.

> **Note**
>
> CURRENT_DATE retorna a data de início para a transação atual, não para o início da instrução atual. Considere o cenário em que você inicia uma transação contendo várias declarações em 10/01/08 23:59, e a declaração contendo CURRENT_DATE é executada em 10/02/08 00:00. CURRENT_DATE retorna `10/01/08`, não `10/02/08`.

Sintaxe

```
CURRENT_DATE
```

Tipo de retorno

DATE

Exemplos

O exemplo a seguir retorna a data atual (na Região da AWS onde a função é executada).

```
select current_date;

   date
------------
2008-10-01
```

O exemplo a seguir cria uma tabela, insere uma linha em que o padrão da coluna `todays_date` é CURRENT_DATE e, depois, seleciona todas as linhas na tabela.

```
CREATE TABLE insert_dates(
    label varchar(128) NOT NULL,
    todays_date DATE DEFAULT CURRENT_DATE);

INSERT INTO insert_dates(label)
VALUES('Date row inserted');

SELECT * FROM insert_dates;


 label            | todays_date
```

```
------------------+-------------
Date row inserted | 2023-05-10
```

Função DATE_CMP

DATE_CMP compara duas datas. A função exibe `0` se as datas forem idênticas, `1` se date1 for maior e `-1` se date2 for maior.

Sintaxe

```
DATE_CMP(date1, date2)
```

Argumentos

date1

Uma coluna de tipo de dados `DATE` ou uma expressão que é avaliada como um tipo `DATE`.

date2

Uma coluna de tipo de dados `DATE` ou uma expressão que é avaliada como um tipo `DATE`.

Tipo de retorno

INTEGER

Exemplos

A seguinte consulta compara os valores de DATE na coluna CALDATE à data 4 de janeiro de 2008 e retorna se o valor em CALDATE é antes (`-1`), igual a (`0`) ou depois (1) de 4 de janeiro de 2008:

```
select caldate, '2008-01-04',
date_cmp(caldate,'2008-01-04')
from date
order by dateid
limit 10;

 caldate   |  ?column?  | date_cmp
-----------+------------+----------
2008-01-01 | 2008-01-04 |       -1
2008-01-02 | 2008-01-04 |       -1
2008-01-03 | 2008-01-04 |       -1
2008-01-04 | 2008-01-04 |        0
```

```
2008-01-05 | 2008-01-04 |        1
2008-01-06 | 2008-01-04 |        1
2008-01-07 | 2008-01-04 |        1
2008-01-08 | 2008-01-04 |        1
2008-01-09 | 2008-01-04 |        1
2008-01-10 | 2008-01-04 |        1
(10 rows)
```

Função DATE_CMP_TIMESTAMP

DATE_CMP_TIMESTAMP compara uma data a um carimbo de data/hora e exibe `0` se os valores forem idênticos, 1 se date for maior cronologicamente e `-1` se timestamp for maior.

Sintaxe

```
DATE_CMP_TIMESTAMP(date, timestamp)
```

Argumentos

date

Uma coluna de tipo de dados `DATE` ou uma expressão que é avaliada como um tipo `DATE`.

timestamp

Uma coluna de tipo de dados `TIMESTAMP` ou uma expressão que é avaliada como um tipo `TIMESTAMP`.

Tipo de retorno

INTEGER

Exemplos

O seguinte exemplo compara a data `2008-06-18` à LISTTIME. Os valores da coluna LISTTIME são carimbos de data/hora. Ofertas feitas antes desta data retornam 1; ofertas feitas após esta data retornam `-1`.

```
select listid, '2008-06-18', listtime,
date_cmp_timestamp('2008-06-18', listtime)
from listing
order by 1, 2, 3, 4
```

```
limit 10;

 listid |  ?column?  |      listtime       | date_cmp_timestamp
--------+------------+---------------------+--------------------
      1 | 2008-06-18 | 2008-01-24 06:43:29 |                  1
      2 | 2008-06-18 | 2008-03-05 12:25:29 |                  1
      3 | 2008-06-18 | 2008-11-01 07:35:33 |                 -1
      4 | 2008-06-18 | 2008-05-24 01:18:37 |                  1
      5 | 2008-06-18 | 2008-05-17 02:29:11 |                  1
      6 | 2008-06-18 | 2008-08-15 02:08:13 |                 -1
      7 | 2008-06-18 | 2008-11-15 09:38:15 |                 -1
      8 | 2008-06-18 | 2008-11-09 05:07:30 |                 -1
      9 | 2008-06-18 | 2008-09-09 08:03:36 |                 -1
     10 | 2008-06-18 | 2008-06-17 09:44:54 |                  1
(10 rows)
```

Função DATE_CMP_TIMESTAMPTZ

DATE_CMP_TIMESTAMPTZ compara uma data a um carimbo de data/hora com fuso horário e exibe `0` se os valores forem idênticos, 1 se date for maior cronologicamente e `-1` se timestamptz for maior.

Sintaxe

```
DATE_CMP_TIMESTAMPTZ(date, timestamptz)
```

Argumentos

date

Uma coluna de tipo de dados `DATE` ou uma expressão que é avaliada implicitamente como um tipo `DATE`.

timestamptz

Uma coluna de tipo de dados `TIMESTAMPTZ` ou uma expressão que é avaliada implicitamente como um tipo `TIMESTAMPTZ`.

Tipo de retorno

INTEGER

Exemplos

O seguinte exemplo compara a data `2008-06-18` à LISTTIME. Ofertas feitas antes desta data retornam 1; ofertas feitas após esta data retornam `-1`.

```
select listid, '2008-06-18', CAST(listtime AS timestamptz),
date_cmp_timestamptz('2008-06-18', CAST(listtime AS timestamptz))
from listing
order by 1, 2, 3, 4
limit 10;

 listid |  ?column?  |      timestamptz       | date_cmp_timestamptz
--------+------------+------------------------+----------------------
      1 | 2008-06-18 | 2008-01-24 06:43:29+00 |                    1
      2 | 2008-06-18 | 2008-03-05 12:25:29+00 |                    1
      3 | 2008-06-18 | 2008-11-01 07:35:33+00 |                   -1
      4 | 2008-06-18 | 2008-05-24 01:18:37+00 |                    1
      5 | 2008-06-18 | 2008-05-17 02:29:11+00 |                    1
      6 | 2008-06-18 | 2008-08-15 02:08:13+00 |                   -1
      7 | 2008-06-18 | 2008-11-15 09:38:15+00 |                   -1
      8 | 2008-06-18 | 2008-11-09 05:07:30+00 |                   -1
      9 | 2008-06-18 | 2008-09-09 08:03:36+00 |                   -1
     10 | 2008-06-18 | 2008-06-17 09:44:54+00 |                    1
(10 rows)
```

Função DATEADD

Incrementa um valor DATE, TIME, TIMETZ ou TIMESTAMP com um intervalo especificado.

Sintaxe

```
DATEADD( datepart, interval, {date|time|timetz|timestamp} )
```

Argumentos

datepart

A parte da data (ano, mês, dia ou hora, por exemplo) sobre a qual a função atua. Para obter mais informações, consulte Partes da data para funções de data ou de timestamp.

intervalo

Um número inteiro que especificou o intervalo (número de dias, por exemplo) a adicionar à expressão de destino. Um número inteiro negativo subtrai o intervalo.

date|time|timetz|timestamp

Uma coluna DATE, TIME, TIMETZ ou TIMESTAMP ou uma expressão que converta implicitamente para DATE, TIME, TIMETZ ou TIMESTAMP A expressão DATE, TIME, TIMETZ ou TIMESTAMP deve conter a parte de data especificada.

Tipo de retorno

TIMESTAMP, TIME ou TIMETZ, dependendo do tipo de dados de entrada.

Exemplos com uma coluna DATE

O exemplo a seguir adiciona 30 dias a cada data em novembro que existe na tabela DATE.

```
select dateadd(day,30,caldate) as novplus30
from date
where month='NOV'
order by dateid;

novplus30
---------------------
2008-12-01 00:00:00
2008-12-02 00:00:00
2008-12-03 00:00:00
...
(30 rows)
```

O exemplo a seguir adiciona 18 meses a um valor de data literal.

```
select dateadd(month,18,'2008-02-28');

date_add
---------------------
2009-08-28 00:00:00
(1 row)
```

O nome padrão da coluna para uma função DATEADD é DATE_ADD. O timestamp padrão para um valor de data é `00:00:00`.

O exemplo a seguir adiciona 30 minutos a um valor de data que não especifica um timestamp.

```
select dateadd(m,30,'2008-02-28');

date_add
---------------------
2008-02-28 00:30:00
(1 row)
```

Você pode nomear as partes da data por completo ou abreviá-las. Neste caso, m significa minutos, não meses.

Exemplos com uma coluna TIME

O TIME_TEST da tabela a seguir tem uma coluna TIME_VAL (tipo TIME) com três valores inseridos.

```
select time_val from time_test;

time_val
---------------------
20:00:00
00:00:00.5550
00:58:00
```

O exemplo a seguir adiciona 5 minutos a cada TIME_VAL na tabela TIME_TEST.

```
select dateadd(minute,5,time_val) as minplus5 from time_test;

minplus5
---------------
20:05:00
00:05:00.5550
01:03:00
```

O exemplo a seguir adiciona 8 horas a um valor de tempo literal.

```
select dateadd(hour, 8, time '13:24:55');

date_add
```

```
---------------
21:24:55
```

O exemplo a seguir mostra quando um tempo passa por 24:00:00 ou abaixo de 00:00:00.

```
select dateadd(hour, 12, time '13:24:55');

date_add
---------------
01:24:55
```

Exemplos com uma coluna TIMETZ

Os valores de saída nestes exemplos estão em UTC, que é o fuso horário padrão.

O TIMETZ_TEST da tabela de exemplo a seguir tem uma coluna TIMETZ_VAL (tipo TIMETZ) com três valores inseridos.

```
select timetz_val from timetz_test;

timetz_val
------------------
04:00:00+00
00:00:00.5550+00
05:58:00+00
```

O exemplo a seguir adiciona 5 minutos a cada TIMETZ_VAL na tabela TIMETZ_TEST.

```
select dateadd(minute,5,timetz_val) as minplus5_tz from timetz_test;

minplus5_tz
---------------
04:05:00+00
00:05:00.5550+00
06:03:00+00
```

O exemplo a seguir adiciona 2 horas a um valor timetz literal.

```
select dateadd(hour, 2, timetz '13:24:55 PST');

date_add
---------------
```

```
23:24:55+00
```

Exemplos com uma coluna TIMESTAMP

Os valores de saída nestes exemplos estão em UTC, que é o fuso horário padrão.

O exemplo de tabela TIMESTAMP_TEST a seguir tem uma coluna TIMESTAMP_VAL (tipo TIMESTAMP) com três valores inseridos.

```
SELECT timestamp_val FROM timestamp_test;

timestamp_val
------------------
1988-05-15 10:23:31
2021-03-18 17:20:41
2023-06-02 18:11:12
```

O exemplo a seguir adiciona 20 anos somente aos valores de TIMESTAMP_VAL em TIMESTAMP_TEST anteriores ao ano 2000.

```
SELECT dateadd(year,20,timestamp_val)
FROM timestamp_test
WHERE timestamp_val < to_timestamp('2000-01-01 00:00:00', 'YYYY-MM-DD HH:MI:SS');

date_add
---------------
2008-05-15 10:23:31
```

O exemplo a seguir adiciona 5 segundos a um valor literal de carimbo de data/hora gravado sem um indicador de segundos.

```
SELECT dateadd(second, 5, timestamp '2001-06-06');

date_add
---------------
2001-06-06 00:00:05
```

Observações de uso

As funções DATEADD(month, …) e ADD_MONTHS lidam com datas que caiam no final do mês de forma diferente.

- ADD_MONTHS: Se a data que você estiver adicionando for o último dia do mês, o resultado será sempre o último dia do mês do resultado, independentemente do tamanho do mês. Por exemplo, 30 de abril + 1 mês é o dia 31 de maio.

```
select add_months('2008-04-30',1);

add_months
---------------------
2008-05-31 00:00:00
(1 row)
```

- DATEADD: Se houver menos dias na data que você está adicionando do que no mês de resultado, o resultado será o dia correspondente do mês de resultado, não o último dia desse mês. Por exemplo, 30 de abril + 1 mês é o dia 30 de maio.

```
select dateadd(month,1,'2008-04-30');

date_add
---------------------
2008-05-30 00:00:00
(1 row)
```

A função DATEADD lida com a data de ano bissexto 02-29 de forma diferente ao usar dateadd(month, 12,…) ou dateadd(year, 1, …).

```
select dateadd(month,12,'2016-02-29');

date_add
---------------------
2017-02-28 00:00:00

select dateadd(year, 1, '2016-02-29');

date_add
---------------------
2017-03-01 00:00:00
```

Função DATEDIFF

DATEDIFF retorna a diferença entre as partes de data de duas expressões de data ou hora.

Sintaxe

```
DATEDIFF( datepart, {date|time|timetz|timestamp}, {date|time|timetz|timestamp} )
```

Argumentos

datepart

A parte específica do valor de data ou hora (ano, mês ou dia, hora, minuto, segundo, milissegundo ou microsegundo) sobre a qual a função atua. Para obter mais informações, consulte Partes da data para funções de data ou de timestamp.

Especificamente, DATEDIFF determina o número de limites da parte da data que são cruzados entre duas expressões. Por exemplo, suponha que você esteja calculando a diferença em anos entre duas datas, `12-31-2008` e `01-01-2009`. Neste caso, a função retorna 1 ano, apesar do fato de que essas datas são apenas um dia de diferença. Se você estiver encontrando a diferença em horas entre dois timestamps, `01-01-2009 8:30:00` e `01-01-2009 10:00:00`, o resultado é 2 horas. Se você estiver encontrando a diferença em horas entre dois timestamps, `8:30:00` e `10:00:00`, o resultado é 2 horas.

date|time|timetz|timestamp

Uma coluna ou expressões DATE, TIME, TIMETZ ou TIMESTAMP que implicitamente convertem em DATE, TIME, TIMETZ ou TIMESTAMP. As expressões devem conter a parte da data ou hora especificada. Se a segunda data ou hora for mais recente do que a primeira data ou hora, o resultado será positivo. Se a segunda data ou hora for mais antiga do que a primeira data ou hora, o resultado será negativo.

Tipo de retorno

BIGINT

Exemplos com uma coluna DATE

O exemplo a seguir encontra a diferença, em número de semanas, entre dois valores de data literais.

```
select datediff(week,'2009-01-01','2009-12-31') as numweeks;

numweeks
----------
```

```
52
(1 row)
```

O exemplo a seguir encontra a diferença, em horas, entre dois valores de data literais. Quando você não fornece o valor de hora para uma data, o padrão é 00:00:00.

```
select datediff(hour, '2023-01-01', '2023-01-03 05:04:03');

date_diff
----------
53
(1 row)
```

O exemplo a seguir encontra a diferença, em dias, entre dois valores literais de TIMESTAMETZ.

```
Select datediff(days, 'Jun 1,2008  09:59:59 EST', 'Jul 4,2008  09:59:59 EST')

date_diff
----------
33
```

O exemplo a seguir encontra a diferença, em dias, entre duas datas na mesma linha de uma tabela.

```
select * from date_table;

start_date |   end_date
-----------+-----------
2009-01-01 | 2009-03-23
2023-01-04 | 2024-05-04
(2 rows)

select datediff(day, start_date, end_date) as duration from date_table;

duration
---------
      81
     486
(2 rows)
```

O exemplo a seguir encontra a diferença, em número de trimestres, entre um valor literal no passado e a data de hoje. Este exemplo presume que a data atual seja 5 de junho de 2008. Você pode

nomear as partes da data por completo ou abreviá-las. O nome padrão da coluna para a função DATEDIFF é DATE_DIFF.

```
select datediff(qtr, '1998-07-01', current_date);

date_diff
-----------
40
(1 row)
```

O exemplo a seguir une as tabelas SALES e LISTING para calcular quantos dias os ingressos foram vendidos para as listagens 1000 a 1005 depois de serem listados. A espera mais longa para vendas dessas ofertas foi de 15 dias e a espera mais curta foi de menos de um dia (0 dias).

```
select priceperticket,
datediff(day, listtime, saletime) as wait
from sales, listing where sales.listid = listing.listid
and sales.listid between 1000 and 1005
order by wait desc, priceperticket desc;

priceperticket | wait
---------------+------
 96.00         |   15
 123.00        |   11
 131.00        |    9
 123.00        |    6
 129.00        |    4
 96.00         |    4
 96.00         |    0
(7 rows)
```

Este exemplo calcula o número médio de horas que os vendedores esperaram para todas as vendas de ingressos.

```
select avg(datediff(hours, listtime, saletime)) as avgwait
from sales, listing
where sales.listid = listing.listid;

avgwait
---------
465
```

```
(1 row)
```

Exemplos com uma coluna TIME

O TIME_TEST da tabela a seguir tem uma coluna TIME_VAL (tipo TIME) com três valores inseridos.

```
select time_val from time_test;

time_val
--------------------
20:00:00
00:00:00.5550
00:58:00
```

O exemplo a seguir localiza a diferença no número de horas entre a coluna TIME_VAL e um literal de tempo.

```
select datediff(hour, time_val, time '15:24:45') from time_test;

 date_diff
-----------
        -5
        15
        15
```

O exemplo a seguir localiza a diferença no número de minutos entre dois valores de tempo literal.

```
select datediff(minute, time '20:00:00', time '21:00:00') as nummins;

nummins
----------
60
```

Exemplos com uma coluna TIMETZ

O TIMETZ_TEST da tabela de exemplo a seguir tem uma coluna TIMETZ_VAL (tipo TIMETZ) com três valores inseridos.

```
select timetz_val from timetz_test;
```

```
timetz_val
------------------
04:00:00+00
00:00:00.5550+00
05:58:00+00
```

O exemplo a seguir localiza as diferenças no número de horas, entre um literal TIMETZ e timetz_val.

```
select datediff(hours, timetz '20:00:00 PST', timetz_val) as numhours from timetz_test;

numhours
----------
0
-4
1
```

O exemplo a seguir localiza a diferença no número de horas, entre dois valores TIMETZ literal.

```
select datediff(hours, timetz '20:00:00 PST', timetz '00:58:00 EST') as numhours;

numhours
----------
1
```

Função DATE_PART

DATE_PART extrai os valores de parte da data de uma expressão. DATE_PART é sinônimo da função PGDATE_PART.

Sintaxe

```
DATE_PART(datepart, {date|timestamp})
```

Argumentos

datepart

O literal ou a string de um identificador da parte específica do valor de data (por exemplo, ano, mês ou dia) sobre a qual a função atua. Para obter mais informações, consulte Partes da data para funções de data ou de timestamp.

{date|timestamp}

Uma coluna de data ou timestamp ou uma expressão que se converta implicitamente em uma data ou timestamp. A coluna ou expressão em date ou timestamp deve conter a parte da data especificada em datepart.

Tipo de retorno

DOUBLE

Exemplos

O nome padrão da coluna para a função DATE_PART é `pgdate_part`.

Para obter mais informações sobre os dados usados em alguns desses exemplos, consulte Banco de dados de exemplo.

O exemplo a seguir encontra o minuto de um literal do carimbo de data/hora.

```
SELECT DATE_PART(minute, timestamp '20230104 04:05:06.789');

pgdate_part
-----------
          5
```

O exemplo a seguir encontra o número da semana de um literal do carimbo de data/hora. O cálculo do número da semana segue o padrão ISO 8601. Para obter mais informações, consulte ISO 8601 na Wikipédia.

```
SELECT DATE_PART(week, timestamp '20220502 04:05:06.789');

pgdate_part
-----------
         18
```

O exemplo a seguir encontra o dia do mês de um literal do carimbo de data/hora.

```
SELECT DATE_PART(day, timestamp '20220502 04:05:06.789');

pgdate_part
-----------
```

```
         2
```

O exemplo a seguir encontra o dia da semana de um literal do carimbo de data/hora. O cálculo do número do dia da semana é um inteiro de 0 a 6, começando no domingo.

```
SELECT DATE_PART(dayofweek, timestamp '20220502 04:05:06.789');

pgdate_part
-----------
         1
```

O exemplo a seguir encontra o século de um literal do carimbo de data/hora. O cálculo do número do século segue o padrão ISO 8601. Para obter mais informações, consulte ISO 8601 na Wikipédia.

```
SELECT DATE_PART(century, timestamp '20220502 04:05:06.789');

pgdate_part
-----------
         21
```

O exemplo a seguir encontra o milênio de um literal de timestamp. O cálculo do milênio segue o padrão ISO 8601. Para obter mais informações, consulte ISO 8601 na Wikipédia.

```
SELECT DATE_PART(millennium, timestamp '20220502 04:05:06.789');

pgdate_part
-----------
         3
```

O exemplo a seguir encontra a quantidade de microssegundos de um literal de timestamp. O cálculo do número do microssegundos segue o padrão ISO 8601. Para obter mais informações, consulte ISO 8601 na Wikipédia.

```
SELECT DATE_PART(microsecond, timestamp '20220502 04:05:06.789');

pgdate_part
-----------
    789000
```

O exemplo a seguir encontra o mês de um literal da data.

```
SELECT DATE_PART(month, date '20220502');

pgdate_part
-----------
          5
```

O exemplo a seguir aplica a função DATE_PART à uma coluna em uma tabela.

```
SELECT date_part(w, listtime) AS weeks, listtime
FROM listing
WHERE listid=10


weeks |      listtime
------+---------------------
 25   | 2008-06-17 09:44:54
(1 row)
```

Você pode nomear partes da data completamente ou abreviá-las; nesse caso, w representa semanas.

A parte da data do dia da semana retorna um número inteiro de 0 a 6, começando com domingo. Use DATE_PART com dow (DAYOFWEEK) para visualizar eventos em um sábado.

```
SELECT date_part(dow, starttime) AS dow, starttime
FROM event
WHERE date_part(dow, starttime)=6
ORDER BY 2,1;

 dow |      starttime
-----+---------------------
   6 | 2008-01-05 14:00:00
   6 | 2008-01-05 14:00:00
   6 | 2008-01-05 14:00:00
   6 | 2008-01-05 14:00:00
...
(1147 rows)
```

Função DATE_PART_YEAR

A função DATE_PART_YEAR Extrai o ano a partir de uma data.

Sintaxe

```
DATE_PART_YEAR(date)
```

Argumento

date

Uma coluna de tipo de dados DATE ou uma expressão que é avaliada implicitamente como um tipo DATE.

Tipo de retorno

INTEGER

Exemplos

O exemplo a seguir encontra o ano de um literal da data.

```
SELECT DATE_PART_YEAR(date '20220502 04:05:06.789');

date_part_year
---------------
2022
```

O exemplo a seguir extrai o ano a partir da coluna CALDATE. Os valores na coluna CALDATE são datas. Para obter mais informações sobre os dados usados nesse exemplo, consulte Banco de dados de exemplo.

```
select caldate, date_part_year(caldate)
from date
order by
dateid limit 10;

 caldate   | date_part_year
-----------+----------------
2008-01-01 |           2008
2008-01-02 |           2008
2008-01-03 |           2008
2008-01-04 |           2008
2008-01-05 |           2008
```

```
2008-01-06 |           2008
2008-01-07 |           2008
2008-01-08 |           2008
2008-01-09 |           2008
2008-01-10 |           2008
(10 rows)
```

Função DATE_TRUNC

A função DATE_TRUNC trunca uma expressão de timestamp ou literal com base na parte da data especificada, tal como hora, dia ou mês.

Sintaxe

```
DATE_TRUNC('datepart', timestamp)
```

Argumentos

datepart

A parte da data para qual truncar o valor de timestamp. A entrada timestamp é truncada para que a entrada datepart seja precisa. Por exemplo, `month` trunca para o primeiro dia do mês. Os formatos válidos são:

- microssegundo, microssegundos
- milissegundo, milissegundos
- segundo, segundos
- minuto, minutos
- hora, horas
- dia, dias
- semana, semanas
- mês, meses
- trimestre, trimestres
- ano, anos
- década, décadas
- século, séculos
- milênio, milênios

Para obter mais informações sobre a abreviação de alguns formatos, consulte Partes da data para funções de data ou de timestamp

timestamp

Uma coluna de timestamp ou uma expressão que converta implicitamente para um timestamp.

Tipo de retorno

TIMESTAMP

Exemplos

Truncar o carimbo de data/hora de entrada para o segundo.

```
SELECT DATE_TRUNC('second', TIMESTAMP '20200430 04:05:06.789');
date_trunc
2020-04-30 04:05:06
```

Truncar timestamp para minuto.

```
SELECT DATE_TRUNC('minute', TIMESTAMP '20200430 04:05:06.789');
date_trunc
2020-04-30 04:05:00
```

Truncar timestamp para hora.

```
SELECT DATE_TRUNC('hour', TIMESTAMP '20200430 04:05:06.789');
date_trunc
2020-04-30 04:00:00
```

Truncar timestamp para dia.

```
SELECT DATE_TRUNC('day', TIMESTAMP '20200430 04:05:06.789');
date_trunc
2020-04-30 00:00:00
```

Truncar timestamp para o primeiro dia de um mês.

```
SELECT DATE_TRUNC('month', TIMESTAMP '20200430 04:05:06.789');
```

```
date_trunc
2020-04-01 00:00:00
```

Truncar timestamp para o primeiro dia de um trimestre.

```
SELECT DATE_TRUNC('quarter', TIMESTAMP '20200430 04:05:06.789');
date_trunc
2020-04-01 00:00:00
```

Truncar timestamp para o primeiro dia de um ano.

```
SELECT DATE_TRUNC('year', TIMESTAMP '20200430 04:05:06.789');
date_trunc
2020-01-01 00:00:00
```

Truncar timestamp para o primeiro dia de um século.

```
SELECT DATE_TRUNC('millennium', TIMESTAMP '20200430 04:05:06.789');
date_trunc
2001-01-01 00:00:00
```

Trunque o carimbo de data/hora de entrada para o a segunda-feira de uma semana.

```
select date_trunc('week', TIMESTAMP '20220430 04:05:06.789');
date_trunc
2022-04-25 00:00:00
```

No exemplo a seguir, a função DATE_TRUNC usa a parte da data “week” para retornar a data para a segunda-feira de cada semana.

```
select date_trunc('week', saletime), sum(pricepaid) from sales where
saletime like '2008-09%' group by date_trunc('week', saletime) order by 1;

date_trunc  |    sum
------------+-------------
2008-09-01  | 2474899
2008-09-08  | 2412354
2008-09-15  | 2364707
2008-09-22  | 2359351
2008-09-29  |  705249
```

Função EXTRACT

A função EXTRACT retorna a parte da data ou hora de um valor de TIMESTAMP, TIMESTAMPTZ, TIME, TIMETZ, INTERVAL YEAR TO MONTH ou INTERVAL DAY TO SECOND. Os exemplos incluem um dia, mês, ano, hora, minuto, segundo, milissegundo ou microssegundo de um timestamp.

Sintaxe

```
EXTRACT(datepart FROM source)
```

Argumentos

datepart

O subcampo de uma data ou hora que será extraído, como dia, mês, ano, hora, minuto, segundo, milissegundo ou microssegundo. Para os possíveis valores, consulte Partes da data para funções de data ou de timestamp.

Fonte de

Uma coluna ou expressão que é avaliada como um tipo de dado TIMESTAMP, TIMESTAMPTZ, TIME, TIMETZ, INTERVAL YEAR TO MONTH ou INTERVAL DAY TO SECOND.

Tipo de retorno

INTEGER se o valor de source for avaliado como TIMESTAMP, TIME, TIMETZ, INTERVAL YEAR TO MONTH ou INTERVAL DAY TO SECOND.

DOUBLE PRECISION se o valor de source for avaliado como TIMESTAMPTZ.

Exemplos com TIMESTAMP

O exemplo a seguir determina os números da semana para vendas em que o preço pago foi de $10.000 ou mais. Este exemplo usa os dados de TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

```
select salesid, extract(week from saletime) as weeknum
from sales
where pricepaid > 9999
order by 2;

salesid | weeknum
```

```
--------+---------
 159073 |       6
 160318 |       8
 161723 |      26
```

O exemplo a seguir retorna o valor de minutos de um valor de timestamp literal.

```
select extract(minute from timestamp '2009-09-09 12:08:43');

date_part
-----------
8
```

O exemplo a seguir retorna o número de milissegundos de um valor de timestamp literal.

```
select extract(ms from timestamp '2009-09-09 12:08:43.101');

date_part
-----------
101
```

Exemplos com TIMESTAMPTZ

O exemplo a seguir retorna o ano de um valor de timestamp literal.

```
select extract(year from timestamptz '1.12.1997 07:37:16.00 PST');

date_part
-----------
1997
```

Exemplos com TIME

O TIME_TEST da tabela a seguir tem uma coluna TIME_VAL (tipo TIME) com três valores inseridos.

```
select time_val from time_test;

time_val
---------------------
20:00:00
00:00:00.5550
00:58:00
```

O exemplo a seguir extrai os minutos de cada time_val.

```
select extract(minute from time_val) as minutes from time_test;

minutes
-----------
         0
         0
         58
```

O exemplo a seguir extrai as horas de cada time_val.

```
select extract(hour from time_val) as hours from time_test;

hours
-----------
         20
         0
         0
```

O exemplo a seguir extrai milissegundos de um valor literal.

```
select extract(ms from time '18:25:33.123456');

 date_part
-----------
     123
```

Exemplos com TIMETZ

O TIMETZ_TEST da tabela de exemplo a seguir tem uma coluna TIMETZ_VAL (tipo TIMETZ) com três valores inseridos.

```
select timetz_val from timetz_test;

timetz_val
------------------
04:00:00+00
00:00:00.5550+00
05:58:00+00
```

O exemplo a seguir extrai as horas de cada timetz_val.

```
select extract(hour from timetz_val) as hours from time_test;

hours
-----------
         4
         0
         5
```

O exemplo a seguir extrai milissegundos de um valor literal. Literais não são convertidos para UTC antes da extração ser processada.

```
select extract(ms from timetz '18:25:33.123456 EST');

 date_part
-----------
     123
```

O exemplo a seguir retorna a diferença de horas de um fuso horário em relação ao UTC de um valor de timetz literal.

```
select extract(timezone_hour from timetz '1.12.1997 07:37:16.00 PDT');

date_part
-----------
-7
```

Exemplos com INTERVAL YEAR TO MONTH e INTERVAL DAY TO SECOND

O exemplo a seguir extrai a parte do dia de 1 de INTERVAL DAY TO SECOND que define 36 horas, que é 1 dia e 12 horas.

```
select EXTRACT('days' from INTERVAL '36 hours' DAY TO SECOND)

 date_part
------------------
 1
```

O exemplo a seguir extrai a parte do mês de 3 de YEAR TO MONTH que define 15 meses, ou seja, 1 ano e 3 meses.

```
select EXTRACT('month' from INTERVAL '15 months' YEAR TO MONTH)

 date_part
------------------
 3
```

O exemplo a seguir extrai a parte do mês de 6 de 30 meses, que é 2 anos e 6 meses.

```
select EXTRACT('month' from INTERVAL '30' MONTH)

 date_part
------------------
 6
```

O exemplo a seguir extrai a parte da hora de 2 de 50 horas, que é 2 dias 2 horas.

```
select EXTRACT('hours' from INTERVAL '50' HOUR)

 date_part
------------------
 2
```

O exemplo a seguir extrai a parte dos minutos de 11 de 1 hora, 11 minutos e 11.123 segundos.

```
select EXTRACT('minute' from INTERVAL '70 minutes 70.123 seconds' MINUTE TO SECOND)

 date_part
------------------
 11
```

O exemplo a seguir extrai a parte dos segundos de 1.11 de 1 dia, 1 hora, 1 minuto e 1.11 segundos.

```
select EXTRACT('seconds' from INTERVAL '1 day 1:1:1.11' DAY TO SECOND)

 date_part
------------------
 1.11
```

O exemplo a seguir extrai o número total de horas em um INTERVAL. Cada parte é extraída e adicionada a um total.

```
select EXTRACT('days' from INTERVAL '50' HOUR) * 24 + EXTRACT('hours' from INTERVAL
 '50' HOUR)

 ?column?
------------------
 50
```

O exemplo a seguir extrai o número total de horas em um INTERVAL. Cada parte é extraída e adicionada a um total.

```
select EXTRACT('days' from INTERVAL '1 day 1:1:1.11' DAY TO SECOND) * 86400 +
       EXTRACT('hours' from INTERVAL '1 day 1:1:1.11' DAY TO SECOND) * 3600 +
       EXTRACT('minutes' from INTERVAL '1 day 1:1:1.11' DAY TO SECOND) * 60 +
       EXTRACT('seconds' from INTERVAL '1 day 1:1:1.11' DAY TO SECOND)

 ?column?
------------------
 90061.11
```

Função GETDATE

GETDATE retorna a atual data e hora no fuso horário da sessão atual (UTC por padrão). Retorna a data ou hora de início da instrução atual, mesmo quando está dentro de um bloco de transação.

Sintaxe

```
GETDATE()
```

Os parênteses são necessários.

Tipo de retorno

TIMESTAMP

Exemplos

O exemplo a seguir usa a função GETDATE para retornar o timestamp completo para a data atual.

```
select getdate();

timestamp
---------------------
```

```
2008-12-04 16:10:43
```

O exemplo a seguir usa a função GETDATE dentro da função TRUNC para retornar a data atual sem a hora.

```
select trunc(getdate());

trunc
------------
2008-12-04
```

Função INTERVAL_CMP

INTERVAL_CMP compara dois intervalos e retorna 1 se o primeiro intervalo for maior, -1 se o segundo intervalo for maior e 0 se os intervalos forem iguais. Para obter mais informações, consulte Exemplos de literais de intervalo sem sintaxe de qualificador.

Sintaxe

```
INTERVAL_CMP(interval1, interval2)
```

Argumentos

interval1

Um valor de intervalo literal.

interval2

Um valor de intervalo literal.

Tipo de retorno

INTEGER

Exemplos

O exemplo a seguir compara o valor de `3 days` com `1 year`.

```
select interval_cmp('3 days','1 year');

interval_cmp
```

```
--------------
-1
```

Este exemplo compara o valor `7 days` com `1 week`.

```
select interval_cmp('7 days','1 week');

interval_cmp
--------------
0
```

O exemplo a seguir compara o valor de `1 year` com `3 days`.

```
select interval_cmp('1 year','3 days');

interval_cmp
--------------
1
```

Função LAST_DAY

LAST_DAY retorna a data do último dia do mês que contenha date. O tipo de retorno é sempre DATE, independente do tipo de dado do argumento date.

Para obter mais informações sobre como recuperar partes específicas de uma data, consulte Função DATE_TRUNC.

Sintaxe

```
LAST_DAY( { date | timestamp } )
```

Argumentos

date | timestamp

Uma coluna de tipo de dados `DATE` ou `TIMESTAMP` ou uma expressão que é avaliada implicitamente como um tipo `DATE` ou `TIMESTAMP`.

Tipo de retorno

DATE

Exemplos

O exemplo a seguir retorna a data do último dia no mês atual.

```
select last_day(sysdate);

  last_day
------------
 2014-01-31
```

O exemplo a seguir retorna o número de ingressos vendido em cada um dos últimos 7 dias do mês. Os valores da coluna SALETIME são carimbos de data/hora.

```
select datediff(day, saletime, last_day(saletime)) as "Days Remaining", sum(qtysold)
from sales
where datediff(day, saletime, last_day(saletime)) < 7
group by 1
order by 1;

days remaining |  sum
---------------+-------
             0 | 10140
             1 | 11187
             2 | 11515
             3 | 11217
             4 | 11446
             5 | 11708
             6 | 10988
(7 rows)
```

Função MONTHS_BETWEEN

MONTHS_BETWEEN determina o número de meses entre duas datas.

Se a primeira data for mais recente do que a segunda, o resultado será positivo; caso contrário, o resultado será negativo.

Se um dos argumentos for nulo o resultado será NULL.

Sintaxe

```
MONTHS_BETWEEN( date1, date2 )
```

Argumentos

date1

Uma coluna de tipo de dados DATE ou uma expressão que é avaliada implicitamente como um tipo DATE.

date2

Uma coluna de tipo de dados DATE ou uma expressão que é avaliada implicitamente como um tipo DATE.

Tipo de retorno

FLOAT8

A porção inteira do número do resultado é baseada na diferença entre os valores de ano e de mês das datas. A parte fracionária do resultado é calculada a partir dos valores de dia e timestamp das datas e presume um mês de 31 dias.

Se date1 e date2 contêm a mesma data em um mês (por exemplo, 1/15/14 e 2/15/14) ou o último dia do mês (por exemplo, 8/31/14 e 9/30/14), o resultado é um número inteiro baseado nos valores de ano e mês das datas, independente de correspondência na porção do timestamp, se houver.

Exemplos

O exemplo a seguir retorna os meses entre 18/1/1969 e 18/3/1969.

```
select months_between('1969-01-18', '1969-03-18')
as months;

months
----------
-2
```

O exemplo a seguir retorna os meses entre 18/1/1969 e 18/1/1969.

```
select months_between('1969-01-18', '1969-01-18')
as months;
```

```
months
----------
0
```

O exemplo a seguir retorna os meses entre a primeira e a última exibição de um evento.

```
select eventname,
min(starttime) as first_show,
max(starttime) as last_show,
months_between(max(starttime),min(starttime)) as month_diff
from event
group by eventname
order by eventname
limit 5;

eventname         first_show             last_show              month_diff
-------------------------------------------------------------------------
.38 Special       2008-01-21 19:30:00.0  2008-12-25 15:00:00.0  11.12
3 Doors Down      2008-01-03 15:00:00.0  2008-12-01 19:30:00.0  10.94
70s Soul Jam      2008-01-16 19:30:00.0  2008-12-07 14:00:00.0  10.7
A Bronx Tale      2008-01-21 19:00:00.0  2008-12-15 15:00:00.0  10.8
A Catered Affair  2008-01-08 19:30:00.0  2008-12-19 19:00:00.0  11.35
```

Função NEXT_DAY

NEXT_DAY retorna a data da primeira instância do dia especificado que é mais recente do que a data fornecida.

Se o valor dia for o mesmo dia da semana da data fornecida, a próxima ocorrência desse dia será retornada.

Sintaxe

```
NEXT_DAY( { date | timestamp }, day )
```

Argumentos

date | timestamp

Uma coluna de tipo de dados DATE ou TIMESTAMP ou uma expressão que é avaliada implicitamente como um tipo DATE ou TIMESTAMP.

dia

Uma string que contém o nome de qualquer dia. O uso de letras maiúsculas não importa.

Os valores válidos são conforme se segue.

Dia	Valores
Domingo	Su, Sun, Sunday
Segunda-feira	M, Mo, Mon, Monday
Terça-feira	3ª, Ter, Terça, Terça-feira
Quarta-feira	4ª, Qua, Quarta, Quarta-feira
Quinta-feira	Th, Thu, Thurs, Thursday
Sexta-feira	F, Fr, Fri, Friday
Sábado	Sa, Sat, Saturday

Tipo de retorno

DATE

Exemplos

O seguinte exemplo retorna a data da primeira terça-feira após 8/20/2014.

```
select next_day('2014-08-20','Tuesday');

next_day
-----------
2014-08-26
```

O exemplo a seguir retorna a data da primeira terça-feira após 1.º/1/2008 às 5:54:44.

```
select listtime, next_day(listtime, 'Tue') from listing limit 1;

listtime            | next_day
--------------------+-----------
```

```
2008-01-01 05:54:44 | 2008-01-08
```

O exemplo a seguir obtém as datas de marketing de destino para o terceiro trimestre.

```
select username, (firstname ||' '|| lastname) as name,
eventname, caldate, next_day (caldate, 'Monday') as marketing_target
from sales, date, users, event
where sales.buyerid = users.userid
and sales.eventid = event.eventid
and event.dateid = date.dateid
and date.qtr = 3
order by marketing_target, eventname, name;

username  |     name          |     eventname        |    caldate   |
 marketing_target
----------+-------------------+----------------------+--------------
+-------------------
MBO26QSG  |   Callum Atkinson | .38 Special          |  2008-07-06  | 2008-07-07
WCR50YIU  |   Erasmus Alvarez | A Doll's House       |  2008-07-03  | 2008-07-07
CKT70OIE  |   Hadassah Adkins | Ana Gabriel          |  2008-07-06  | 2008-07-07
VVG07OUO  |   Nathan Abbott   | Armando Manzanero    |  2008-07-04  | 2008-07-07
GEW77SII  |   Scarlet Avila   | August: Osage County |  2008-07-06  | 2008-07-07
ECR71CVS  |   Caryn Adkins    | Ben Folds            |  2008-07-03  | 2008-07-07
KUW82CYU  |   Kaden Aguilar   | Bette Midler         |  2008-07-01  | 2008-07-07
WZE78DJZ  |   Kay Avila       | Bette Midler         |  2008-07-01  | 2008-07-07
HXY04NVE  |   Dante Austin    | Britney Spears       |  2008-07-02  | 2008-07-07
URY81YWF  |   Wilma Anthony   | Britney Spears       |  2008-07-02  | 2008-07-07
```

Função SYSDATE

SYSDATE retorna a atual data e hora no fuso horário da sessão atual (UTC por padrão).

> **Note**
>
> SYSDATE retorna a data e hora de início para a transação atual, não para o início da instrução atual.

Sintaxe

```
SYSDATE
```

Essa função não requer um argumento.

Tipo de retorno

TIMESTAMP

Exemplos

O exemplo a seguir usa a função SYSDATE para retornar o timestamp completo para a data atual.

```
select sysdate;

timestamp
----------------------------
2008-12-04 16:10:43.976353
```

O exemplo a seguir usa a função SYSDATE dentro da função TRUNC para retornar a data atual sem a hora.

```
select trunc(sysdate);

trunc
------------
2008-12-04
```

A consulta a seguir retorna informações de vendas para datas que caem entre a data em que a consulta é emitida e a data de 120 dias antes.

```
select salesid, pricepaid, trunc(saletime) as saletime, trunc(sysdate) as now
from sales
where saletime between trunc(sysdate)-120 and trunc(sysdate)
order by saletime asc;

 salesid | pricepaid |  saletime  |    now
---------+-----------+------------+------------
91535    |    670.00 | 2008-08-07 | 2008-12-05
91635    |    365.00 | 2008-08-07 | 2008-12-05
91901    |   1002.00 | 2008-08-07 | 2008-12-05
...
```

Função TIMEOFDAY

TIMEOFDAY é um alias especial usado para retornar o dia da semana, data e hora como um valor de string. Retorna a string de hora do dia para a instrução atual, mesmo quando está dentro de um bloco de transação.

Sintaxe

```
TIMEOFDAY()
```

Tipo de retorno

VARCHAR

Exemplos

O exemplo a seguir retorna a data e hora atuais usando a função TIMEOFDAY.

```
select timeofday();

timeofday
------------
Thu Sep 19 22:53:50.333525 2013 UTC
```

Função TIMESTAMP_CMP

Compara o valor de dois timestamps e retorna um número inteiro. Se os carimbos de data/hora forem idênticos, a função retornará `0`. Se o primeiro carimbo de data/hora for maior, a função retornará `1`. Se o segundo carimbo de data/hora for maior, a função retornará `-1`.

Sintaxe

```
TIMESTAMP_CMP(timestamp1, timestamp2)
```

Argumentos

timestamp1

Uma coluna de tipo de dados `TIMESTAMP` ou uma expressão que é avaliada implicitamente como um tipo `TIMESTAMP`.

timestamp2

Uma coluna de tipo de dados `TIMESTAMP` ou uma expressão que é avaliada implicitamente como um tipo `TIMESTAMP`.

Tipo de retorno

INTEGER

Exemplos

O exemplo a seguir compara os carimbos de data/hora e mostra os resultados da comparação.

```
SELECT TIMESTAMP_CMP('2008-01-24 06:43:29', '2008-01-24 06:43:29'),
 TIMESTAMP_CMP('2008-01-24 06:43:29', '2008-02-18 02:36:48'), TIMESTAMP_CMP('2008-02-18
 02:36:48', '2008-01-24 06:43:29');

timestamp_cmp  | timestamp_cmp | timestamp_cmp
---------------+---------------+---------------
             0 |            -1 |             1
```

O seguinte exemplo compara o LISTTIME e SALETIME de uma oferta. Observe que o valor referente a TIMESTAMP_CMP é de -1 para todas as listagens, pois o carimbo de data/hora da venda é posterior ao carimbo de data/hora das listagens.

```
select listing.listid, listing.listtime,
sales.saletime, timestamp_cmp(listing.listtime, sales.saletime)
from listing, sales
where listing.listid=sales.listid
order by 1, 2, 3, 4
limit 10;

 listid |      listtime       |      saletime       | timestamp_cmp
--------+---------------------+---------------------+---------------
      1 | 2008-01-24 06:43:29 | 2008-02-18 02:36:48 |            -1
      4 | 2008-05-24 01:18:37 | 2008-06-06 05:00:16 |            -1
      5 | 2008-05-17 02:29:11 | 2008-06-06 08:26:17 |            -1
      5 | 2008-05-17 02:29:11 | 2008-06-09 08:38:52 |            -1
      6 | 2008-08-15 02:08:13 | 2008-08-31 09:17:02 |            -1
     10 | 2008-06-17 09:44:54 | 2008-06-26 12:56:06 |            -1
     10 | 2008-06-17 09:44:54 | 2008-07-10 02:12:36 |            -1
     10 | 2008-06-17 09:44:54 | 2008-07-16 11:59:24 |            -1
```

```
     10 | 2008-06-17 09:44:54 | 2008-07-22 02:23:17 |            -1
     12 | 2008-07-25 01:45:49 | 2008-08-04 03:06:36 |            -1
(10 rows)
```

Este exemplo mostra que TIMESTAMP_CMP retorna um 0 para timestamps idênticos:

```
select listid, timestamp_cmp(listtime, listtime)
from listing
order by 1 , 2
limit 10;

 listid | timestamp_cmp
--------+---------------
      1 |             0
      2 |             0
      3 |             0
      4 |             0
      5 |             0
      6 |             0
      7 |             0
      8 |             0
      9 |             0
     10 |             0
(10 rows)
```

Função TIMESTAMP_CMP_DATE

TIMESTAMP_CMP_DATE compara o valor de um timestamp e uma data. Se os valores de carimbo de data/hora e data forem idênticos, a função retornará `0`. Se o carimbo de data/hora for maior cronologicamente, a função retornará 1. Se a data for maior, a função retornará `-1`.

Sintaxe

```
TIMESTAMP_CMP_DATE(timestamp, date)
```

Argumentos

timestamp

Uma coluna de tipo de dados `TIMESTAMP` ou uma expressão que é avaliada implicitamente como um tipo `TIMESTAMP`.

date

Uma coluna de tipo de dados DATE ou uma expressão que é avaliada implicitamente como um tipo DATE.

Tipo de retorno

INTEGER

Exemplos

O seguinte exemplo compara LISTTIME à data 2008-06-18. Ofertas feitas após esta data retornam 1; ofertas feitas antes desta data retornam -1. Os valores LISTTIME são carimbos de data/hora.

```
select listid, listtime,
timestamp_cmp_date(listtime, '2008-06-18')
from listing
order by 1, 2, 3
limit 10;


 listid |      listtime       | timestamp_cmp_date
--------+---------------------+--------------------
      1 | 2008-01-24 06:43:29 |                 -1
      2 | 2008-03-05 12:25:29 |                 -1
      3 | 2008-11-01 07:35:33 |                  1
      4 | 2008-05-24 01:18:37 |                 -1
      5 | 2008-05-17 02:29:11 |                 -1
      6 | 2008-08-15 02:08:13 |                  1
      7 | 2008-11-15 09:38:15 |                  1
      8 | 2008-11-09 05:07:30 |                  1
      9 | 2008-09-09 08:03:36 |                  1
     10 | 2008-06-17 09:44:54 |                 -1
(10 rows)
```

Função TIMESTAMP_CMP_TIMESTAMPTZ

TIMESTAMP_CMP_TIMESTAMPTZ compara o valor de uma expressão de timestamp à uma expressão de timestamp com fuso horário. Se os valores do carimbo de data/hora e do carimbo de data/hora com fuso horário forem idênticos, a função retornará 0. Se o carimbo de data/hora for maior cronologicamente, a função retornará 1. Se o carimbo de data/hora com fuso horário for maior, a função retornará –1.

Sintaxe

```
TIMESTAMP_CMP_TIMESTAMPTZ(timestamp, timestamptz)
```

Argumentos

timestamp

Uma coluna de tipo de dados TIMESTAMP ou uma expressão que é avaliada implicitamente como um tipo TIMESTAMP.

timestamptz

Uma coluna de tipo de dados TIMESTAMPTZ ou uma expressão que é avaliada implicitamente como um tipo TIMESTAMPTZ.

Tipo de retorno

INTEGER

Exemplos

O exemplo a seguir compara os carimbos de data/hora a carimbos de data/hora com fusos horários e mostra os resultados da comparação.

```
SELECT TIMESTAMP_CMP_TIMESTAMPTZ('2008-01-24 06:43:29', '2008-01-24 06:43:29+00'),
 TIMESTAMP_CMP_TIMESTAMPTZ('2008-01-24 06:43:29', '2008-02-18 02:36:48+00'),
 TIMESTAMP_CMP_TIMESTAMPTZ('2008-02-18 02:36:48', '2008-01-24 06:43:29+00');

timestamp_cmp_timestamptz  | timestamp_cmp_timestamptz | timestamp_cmp_timestamptz
---------------------------+---------------------------+--------------------------
             0             |            -1             |             1
```

Função TIMESTAMPTZ_CMP

TIMESTAMPTZ_CMP compara o valor de dois timestamps com valores de fuso horário e retorna um número inteiro. Se os carimbos de data/hora forem idênticos, a função retornará 0. Se o primeiro carimbo de data/hora for maior cronologicamente, a função retornará 1. Se o segundo carimbo de data/hora for maior, a função retornará -1.

Sintaxe

```
TIMESTAMPTZ_CMP(timestamptz1, timestamptz2)
```

Argumentos

timestamptz1

Uma coluna de tipo de dados TIMESTAMPTZ ou uma expressão que é avaliada implicitamente como um tipo TIMESTAMPTZ.

timestamptz2

Uma coluna de tipo de dados TIMESTAMPTZ ou uma expressão que é avaliada implicitamente como um tipo TIMESTAMPTZ.

Tipo de retorno

INTEGER

Exemplos

O exemplo a seguir compara os carimbos de data/hora com fusos horários e mostra os resultados da comparação.

```
SELECT TIMESTAMPTZ_CMP('2008-01-24 06:43:29+00', '2008-01-24 06:43:29+00'),
 TIMESTAMPTZ_CMP('2008-01-24 06:43:29+00', '2008-02-18 02:36:48+00'),
 TIMESTAMPTZ_CMP('2008-02-18 02:36:48+00', '2008-01-24 06:43:29+00');

timestamptz_cmp  | timestamptz_cmp | timestamptz_cmp
-----------------+-----------------+----------------
        0        |       -1        |       1
```

Função TIMESTAMPTZ_CMP_DATE

TIMESTAMPTZ_CMP_DATE compara o valor de um timestamp e uma data. Se os valores de carimbo de data/hora e data forem idênticos, a função retornará 0. Se o carimbo de data/hora for maior cronologicamente, a função retornará 1. Se a data for maior, a função retornará –1.

Sintaxe

```
TIMESTAMPTZ_CMP_DATE(timestamptz, date)
```

Argumentos

timestamptz

Uma coluna de tipo de dados TIMESTAMPTZ ou uma expressão que é avaliada implicitamente como um tipo TIMESTAMPTZ.

date

Uma coluna de tipo de dados DATE ou uma expressão que é avaliada implicitamente como um tipo DATE.

Tipo de retorno

INTEGER

Exemplos

O exemplo a seguir compara LISTTIME como um carimbo de data/hora com fuso horário e a data 2008-06-18. Ofertas feitas após esta data retornam 1; ofertas feitas antes desta data retornam -1.

```
select listid, CAST(listtime as timestamptz) as tstz,
timestamp_cmp_date(tstz, '2008-06-18')
from listing
order by 1, 2, 3
limit 10;


 listid |          tstz          | timestamptz_cmp_date
--------+------------------------+----------------------
      1 | 2008-01-24 06:43:29+00 |                   -1
      2 | 2008-03-05 12:25:29+00 |                   -1
      3 | 2008-11-01 07:35:33+00 |                    1
      4 | 2008-05-24 01:18:37+00 |                   -1
      5 | 2008-05-17 02:29:11+00 |                   -1
      6 | 2008-08-15 02:08:13+00 |                    1
      7 | 2008-11-15 09:38:15+00 |                    1
      8 | 2008-11-09 05:07:30+00 |                    1
      9 | 2008-09-09 08:03:36+00 |                    1
     10 | 2008-06-17 09:44:54+00 |                   -1
(10 rows)
```

Função TIMESTAMPTZ_CMP_TIMESTAMP

TIMESTAMPTZ_CMP_TIMESTAMP compara o valor de uma expressão de timestamp com fuso horário à uma expressão de timestamp. Se os valores do carimbo de data/hora com fuso horário e do carimbo de data/hora forem idênticos, a função retornará `0`. Se o carimbo de data/hora com fuso horário for maior cronologicamente, a função retornará 1. Se o carimbo de data/hora for maior, a função retornará `-1`.

Sintaxe

```
TIMESTAMPTZ_CMP_TIMESTAMP(timestamptz, timestamp)
```

Argumentos

timestamptz

Uma coluna de tipo de dados `TIMESTAMPTZ` ou uma expressão que é avaliada implicitamente como um tipo `TIMESTAMPTZ`.

timestamp

Uma coluna de tipo de dados `TIMESTAMP` ou uma expressão que é avaliada implicitamente como um tipo `TIMESTAMP`.

Tipo de retorno

INTEGER

Exemplos

O exemplo a seguir compara os carimbos de data/hora com fusos horários a carimbos de data/hora e mostra os resultados da comparação.

```
SELECT TIMESTAMPTZ_CMP_TIMESTAMP('2008-01-24 06:43:29+00', '2008-01-24 06:43:29'),
 TIMESTAMPTZ_CMP_TIMESTAMP('2008-01-24 06:43:29+00', '2008-02-18 02:36:48'),
 TIMESTAMPTZ_CMP_TIMESTAMP('2008-02-18 02:36:48+00', '2008-01-24 06:43:29');

timestamptz_cmp_timestamp  | timestamptz_cmp_timestamp | timestamptz_cmp_timestamp
---------------------------+---------------------------+--------------------------
             0             |            -1             |             1
```

Função TIMEZONE

TIMEZONE retorna um timestamp para o valor de fuso horário e de timestamp especificados.

Para obter informações e exemplos sobre como definir o fuso horário, consulte timezone.

Para obter informações e exemplos sobre como converter o fuso horário, consulte CONVERT_TIMEZONE.

Sintaxe

```
TIMEZONE('timezone', { timestamp | timestamptz })
```

Argumentos

timezone

O fuso horário para o valor de retorno. O fuso horário pode ser especificado como um nome de fuso horário (como **'Africa/Kampala'** ou **'Singapore'**) ou como uma abreviação de fuso horário (como **'UTC'** ou **'PDT'**). Para visualizar uma lista de nomes de fusos horários compatíveis, execute o comando a seguir.

```
select pg_timezone_names();
```

Para visualizar uma lista de abreviações de fusos horários compatíveis, execute o comando a seguir.

```
select pg_timezone_abbrevs();
```

Observe que o Amazon Redshift usa o banco de dados de fuso horário IANA como fonte autorizada de especificação de fuso horário. Para ter mais informações e exemplos, consulte Observações quanto ao uso de fuso horário.

timestamp | timestamptz

Uma expressão que resulta em um tipo TIMESTAMP ou TIMESTAMPTZ ou um valor que pode implicitamente ser convertido em um carimbo de data/hora com fuso horário.

Tipo de retorno

TIMESTAMPTZ quando usado com uma expressão TIMESTAMP.

TIMESTAMP quando usado com uma expressão TIMESTAMPTZ.

Exemplos

O seguinte retorna um carimbo de data/hora para o fuso horário UTC usando o carimbo de data/hora `2008-06-17 09:44:54` do fuso horário PST:

```
SELECT TIMEZONE('PST', '2008-06-17 09:44:54');

timezone
-----------------------
2008-06-17 17:44:54+00
```

O seguinte retorna um carimbo de data/hora para o fuso horário PST usando o carimbo de data/hora com fuso horário UTC `2008-06-17 09:44:54+00`:

```
SELECT TIMEZONE('PST', timestamptz('2008-06-17 09:44:54+00'));

timezone
-----------------------
2008-06-17 01:44:54
```

Função TO_TIMESTAMP

TO_TIMESTAMP converte uma string de TIMESTAMP em TIMESTAMPTZ. Para obter uma lista de funções adicionais de data e hora para o Amazon Redshift, consulte Perfis de data e hora.

Sintaxe

```
to_timestamp(timestamp, format)
```

```
to_timestamp (timestamp, format, is_strict)
```

Argumentos

timestamp

Uma string que representa um valor de timestamp no formato especificado por format. Se esse argumento for deixado vazio, o valor padrão de timestamp será `0001-01-01 00:00:00`.

formato

Um literal de string que define o formato do valor de timestamp. Formatos que incluem um fuso horário (**TZ**, **tz** ou **OF**) não têm suporte como entrada. Para os formatos de timestamp válidos, consulte Strings de formato datetime.

is_strict

Um valor booleano opcional que especifica se um erro será retornado se um valor de timestamp de entrada estiver fora do intervalo. Quando is_strict for definido como TRUE, um erro será retornado se houver um valor fora do intervalo. Quando is_strict estiver definido como FALSE, que é o padrão, então os valores de estouro são aceitos.

Tipo de retorno

TIMESTAMPTZ

Exemplos

O exemplo a seguir mostra como usar a função TO_TIMESTAMP para converter uma string TIMESTAMP em TIMESTAMPTZ.

```
select sysdate, to_timestamp(sysdate, 'YYYY-MM-DD HH24:MI:SS') as second;

timestamp                  | second
---------------------------   ---------------------
2021-04-05 19:27:53.281812 | 2021-04-05 19:27:53+00
```

É possível enviar parte de uma data com TO_TIMESTAMP. As partes restantes da data são definidas como valores padrão. A hora é incluída na saída:

```
SELECT TO_TIMESTAMP('2017','YYYY');

to_timestamp
---------------------------
2017-01-01 00:00:00+00
```

A seguinte instrução SQL converte a string “2011-12-18 24:38:15” para um TIMESTAMPTZ. O resultado é um TIMESTAMPTZ que cai no dia seguinte porque o número de horas é superior a 24 horas:

```
SELECT TO_TIMESTAMP('2011-12-18 24:38:15', 'YYYY-MM-DD HH24:MI:SS');

to_timestamp
----------------------
2011-12-19 00:38:15+00
```

A seguinte instrução SQL converte a string “2011-12-18 24:38:15” para um TIMESTAMPTZ. O resultado é um erro porque o valor de hora no timestamp é superior a 24 horas:

```
SELECT TO_TIMESTAMP('2011-12-18 24:38:15', 'YYYY-MM-DD HH24:MI:SS', TRUE);

ERROR:  date/time field time value out of range: 24:38:15.0
```

Função TRUNC

Trunca um TIMESTAMP e retorna um DATE.

Essa função também pode truncar um número. Para obter mais informações, consulte Função TRUNC.

Sintaxe

```
TRUNC(timestamp)
```

Argumentos

timestamp

Uma coluna de tipo de dados TIMESTAMP ou uma expressão que é avaliada implicitamente como um tipo TIMESTAMP.

Para retornar um valor de carimbo de data/hora com 00:00:00 como a hora, converta o resultado da função em um TIMESTAMP.

Tipo de retorno

DATE

Exemplos

O exemplo a seguir retorna a parte da data do resultado da função SYSDATE (que retorna um timestamp).

```
SELECT SYSDATE;

+----------------------------+
|         timestamp          |
+----------------------------+
| 2011-07-21 10:32:38.248109 |
+----------------------------+

SELECT TRUNC(SYSDATE);

+------------+
|   trunc    |
+------------+
| 2011-07-21 |
+------------+
```

O exemplo a seguir aplica a função TRUNC a uma coluna `TIMESTAMP`. O tipo de retorno é uma data.

```
SELECT TRUNC(starttime) FROM event
ORDER BY eventid LIMIT 1;

+------------+
|   trunc    |
+------------+
| 2008-01-25 |
+------------+
```

O exemplo a seguir retorna um valor de carimbo de data/hora com `00:00:00` como a hora ao converter o resultado da função TRUNC em um `TIMESTAMP`.

```
SELECT CAST((TRUNC(SYSDATE)) AS TIMESTAMP);

+---------------------+
|        trunc        |
+---------------------+
```

```
| 2011-07-21 00:00:00 |
+---------------------+
```

Partes da data para funções de data ou de timestamp

A tabela a seguir identifica os nomes e abreviações da parte da data e da hora que são aceitos como argumentos para as seguintes funções:

- DATEADD
- DATEDIFF
- DATE_PART
- EXTRACT

Parte da data ou parte da hora	Abreviações
milênio, milênios	mil, mils
século, séculos	c, cent, cents
década, décadas	dec, decs
epoch	epoch (compatível com EXTRACT)
ano, anos	y, yr, yrs
trimestre, trimestres	qtr, qtrs
mês, meses	mon, mons
semana, semanas	w
dia da semana	dayofweek, dow, dw, weekday (compatível com DATE_PART e Função EXTRACT) Retorna um número inteiro de 0 a 6, começando com domingo.

Parte da data ou parte da hora	Abreviações
	Note A parte da data DOW se comporta de maneira diferente da parte da data do dia da semana (D) usada para strings de formato de data e hora. D se baseia no números inteiros 1 a 7, onde domingo é 1. Para obter mais informações, consulte Strings de formato datetime.
dia do ano	dayofyear, doy, dy, yearday (compatível com EXTRACT)
dia, dias	d
hora, horas	h, hr, hrs
minuto, minutos	m, min, mins
segundo, segundos	s, sec, secs
milissegundo, milissegundos	ms, msec, msecs, msecond, mseconds, millisec, millisecs, millisecon
microssegundo, microssegundos	microsec, microsecs, microsecond, usecond, useconds, us, usec, usecs
timezone, timezone_hour, timezone_minute	Compatível com EXTRACT para timestamp somente com fuso horário (TIMESTAMPTZ).

Variações nos resultados com segundos, milissegundos e microssegundos

Pequenas diferenças nos resultados de consultas ocorrem quando diferentes funções de data especificam segundos, milissegundos ou microssegundos como partes da data:

- A função EXTRACT retorna números inteiros somente para a parte da data especificada, ignorando partes de data de níveis superiores e inferiores. Se a parte da data especificada é segundos, os milissegundos e os microssegundos não são incluídos no resultados. Se a parte da

data especificada é milissegundos, segundos e microssegundos não são incluídos. Se a parte da data especificada é microssegundos, segundos e milissegundos não são incluídos.

- A função DATE_PART retorna a parte completa de segundos do timestamp, independente da parte de data especificada, retornando um valor decimal ou um número inteiro conforme necessário.

Por exemplo, compare os resultados das consultas a seguir:

```
create table seconds(micro timestamp);

insert into seconds values('2009-09-21 11:10:03.189717');

select extract(sec from micro) from seconds;

date_part
-----------
3

select date_part(sec, micro) from seconds;

pgdate_part
-------------
3.189717
```

Observações de CENTURY, EPOCH, DECADE e MIL

CENTURY ou CENTURIES

O Amazon Redshift interpreta um CENTURY para começar com o ano ###1 e terminar com o ano ###0:

```
select extract (century from timestamp '2000-12-16 12:21:13');
date_part
-----------
20

select extract (century from timestamp '2001-12-16 12:21:13');
date_part
-----------
21
```

EPOCH

A implementação do Amazon Redshift de EPOCH é relativa a 1970-01-01 00:00:00.000000 independente do fuso horário onde o cluster reside. Você pode precisar deslocar os resultados pela diferença em horas dependendo do fuso horário onde o cluster está localizado.

O exemplo a seguir faz o seguinte:

1. Cria uma tabela chamada EVENT_EXAMPLE com base na tabela EVENT. Este comando CREATE AS usa a função DATE_PART para criar uma coluna de data (chamada PGDATE_PART por padrão) para armazenar o valor de epoch para cada evento.
2. Seleciona a coluna e o tipo de dados de EVENT_EXAMPLE a partir de PG_TABLE_DEF.
3. Seleciona EVENTNAME, STARTTIME e PGDATE_PART a partir da tabela EVENT_EXAMPLE para visualizar os diferentes formatos de data e hora.
4. Seleciona EVENTNAME e STARTTIME a partir da tabela EVENT EXAMPLE na forma atual. Converte valores de epoch em PGDATE_PART usando um intervalo de 1 segundo para um timestamp sem fuso horário e retorna os resultados em uma coluna chamada CONVERTED_TIMESTAMP.

```
create table event_example
as select eventname, starttime, date_part(epoch, starttime) from event;

select "column", type from pg_table_def where tablename='event_example';

     column    |            type
---------------+-----------------------------
 eventname     | character varying(200)
 starttime     | timestamp without time zone
 pgdate_part   | double precision
(3 rows)
```

```
select eventname, starttime, pgdate_part from event_example;

   eventname          |      starttime      | pgdate_part
----------------------+---------------------+-------------
 Mamma Mia!           | 2008-01-01 20:00:00 |  1199217600
 Spring Awakening     | 2008-01-01 15:00:00 |  1199199600
 Nas                  | 2008-01-01 14:30:00 |  1199197800
 Hannah Montana       | 2008-01-01 19:30:00 |  1199215800
 K.D. Lang            | 2008-01-01 15:00:00 |  1199199600
```

```
 Spamalot             | 2008-01-02 20:00:00 |  1199304000
 Macbeth              | 2008-01-02 15:00:00 |  1199286000
 The Cherry Orchard   | 2008-01-02 14:30:00 |  1199284200
 Macbeth              | 2008-01-02 19:30:00 |  1199302200
 Demi Lovato          | 2008-01-02 19:30:00 |  1199302200


select eventname,
starttime,
timestamp with time zone 'epoch' + pgdate_part * interval '1 second' AS
 converted_timestamp
from event_example;

       eventname      |      starttime      | converted_timestamp
----------------------+---------------------+---------------------
 Mamma Mia!           | 2008-01-01 20:00:00 | 2008-01-01 20:00:00
 Spring Awakening     | 2008-01-01 15:00:00 | 2008-01-01 15:00:00
 Nas                  | 2008-01-01 14:30:00 | 2008-01-01 14:30:00
 Hannah Montana       | 2008-01-01 19:30:00 | 2008-01-01 19:30:00
 K.D. Lang            | 2008-01-01 15:00:00 | 2008-01-01 15:00:00
 Spamalot             | 2008-01-02 20:00:00 | 2008-01-02 20:00:00
 Macbeth              | 2008-01-02 15:00:00 | 2008-01-02 15:00:00
 The Cherry Orchard   | 2008-01-02 14:30:00 | 2008-01-02 14:30:00
 Macbeth              | 2008-01-02 19:30:00 | 2008-01-02 19:30:00
 Demi Lovato          | 2008-01-02 19:30:00 | 2008-01-02 19:30:00
 ...
```

DECADE ou DECADES

O Amazon Redshift interpreta DECADE or DECADES DATEPART com base no calendário comum. Por exemplo, como o calendário comum começa a partir do ano 1, a primeira década (década 1) é 0001-01-01 a 0009-12-31 e a segunda década (década 2) é 0010-01-01 a 0019-12-31. Por exemplo, a década 201 vai de 2000-01-01 a 2009-12-31:

```
select extract(decade from timestamp '1999-02-16 20:38:40');
date_part
-----------
200

select extract(decade from timestamp '2000-02-16 20:38:40');
date_part
-----------
201
```

```
select extract(decade from timestamp '2010-02-16 20:38:40');
date_part
-----------
202
```

MIL ou MILS

O Amazon Redshift interpreta que um MIL começa no primeiro dia do ano #001 e termina no último dia do ano #000:

```
select extract (mil from timestamp '2000-12-16 12:21:13');
date_part
-----------
2

select extract (mil from timestamp '2001-12-16 12:21:13');
date_part
-----------
3
```

Funções de hash

Tópicos

- Função CHECKSUM
- Função farmFingerprint64
- Função FUNC_SHA1
- Função FNV_HASH
- Função MD5
- Função SHA
- Função SHA1
- Função SHA2
- MURMUR3_32_HASH

Uma função de hash é uma função matemática que converte um valor de entrada numérico em outro valor.

Função CHECKSUM

Computa um valor de soma de verificação para criação de um índice de hash.

Sintaxe

```
CHECKSUM(expression)
```

Argumento

expressão

A expressão de entrada deve ser um tipo de dados VARCHAR, INTEGER ou DECIMAL.

Tipo de retorno

A função CHECKSUM retorna um inteiro.

Exemplo

Os seguintes exemplos computam o valor da soma de verificação para a coluna COMMISSION:

```
select checksum(commission)
from sales
order by salesid
limit 10;

checksum
----------
10920
1140
5250
2625
2310
5910
11820
2955
8865
975
(10 rows)
```

Função farmFingerprint64

Calcula o valor do farmhash do argumento de entrada usando a função `Fingerprint64`.

Sintaxe

```
farmFingerprint64(expression)
```

Argumento

expressão

A expressão de entrada deve ser um tipo de dados `VARCHAR` ou `VARBYTE`.

Tipo de retorno

A função `farmFingerprint64` retorna um `BIGINT`.

Exemplo

O exemplo a seguir retorna o valor `farmFingerprint64` do `Amazon Redshift` que é inserido como um tipo de dados `VARCHAR`.

```
SELECT farmFingerprint64('Amazon Redshift');
```

```
  farmfingerprint64
---------------------
 8085098817162212970
```

O exemplo a seguir retorna o valor `farmFingerprint64` do `Amazon Redshift` que é inserido como um tipo de dados `VARBYTE`.

```
SELECT farmFingerprint64('Amazon Redshift'::varbyte);
```

```
  farmfingerprint64
---------------------
```

```
8085098817162212970
```

Função FUNC_SHA1

Sinônimo da função SHA1.

Consulte Função SHA1.

Função FNV_HASH

Calcula a função de hash não criptográfica FNV-1a de 64 bits para todos os tipos de dados básicos.

Sintaxe

```
FNV_HASH(value [, seed])
```

Argumentos

valor

O valor de entrada ao qual aplicar o hash. O Amazon Redshift usa a representação binária do valor para aplicar o hash ao valor de entrada; por exemplo, é aplicado hash aos valores INTEGER usando 4 bytes e aos valores BIGINT usando 8 bytes. Além disso, a aplicação do hash a entradas CHAR e VARCHAR não ignora espaços à esquerda.

seed

A semente BIGINT da função de hash é opcional. Se não for fornecida, o Amazon Redshift usará a semente FNV padrão. Isso permite combinar o hash de várias colunas sem conversões nem concatenações.

Tipo de retorno

BIGINT

Exemplo

Os exemplos a seguir retornam o hash FNV de um número, a string "Amazon Redshift" e a concatenação dos dois.

```
select fnv_hash(1);
```

```
        fnv_hash
----------------------
 -5968735742475085980
(1 row)
```

```
select fnv_hash('Amazon Redshift');
      fnv_hash
--------------------
 7783490368944507294
(1 row)
```

```
select fnv_hash('Amazon Redshift', fnv_hash(1));
       fnv_hash
----------------------
 -2202602717770968555
(1 row)
```

Observações de uso

- Para calcular o hash de uma tabela com várias colunas, é possível calcular o hash FNV da primeira coluna e transmiti-lo como uma semente para o hash da segunda coluna. Depois, ele passa o hash FNV da segunda coluna como uma semente para o hash da terceira coluna.

  O exemplo a seguir cria sementes para aplicar o hash a uma tabela com várias colunas.

  ```
  select fnv_hash(column_3, fnv_hash(column_2, fnv_hash(column_1))) from sample_table;
  ```

- A mesma propriedade pode ser usada para calcular o hash de uma concatenação de strings.

  ```
  select fnv_hash('abcd');
         fnv_hash
  ----------------------
   -281581062704388899
  (1 row)
  ```

  ```
  select fnv_hash('cd', fnv_hash('ab'));
        fnv_hash
  ----------------------
   -281581062704388899
  (1 row)
  ```

- A função de hash usa o tipo de entrada para determinar o número de bytes para hash. Use a fundição para impor um tipo específico, se necessário.

  Os exemplos a seguir usam diferentes tipos de entrada para produzir resultados diferentes.

```
select fnv_hash(1::smallint);
      fnv_hash
--------------------
 589727492704079044
(1 row)
```

```
select fnv_hash(1);
       fnv_hash
----------------------
 -5968735742475085980
(1 row)
```

```
select fnv_hash(1::bigint);
       fnv_hash
----------------------
 -8517097267634966620
(1 row)
```

Função MD5

Usa a função de hash criptográfica MD5 para converter uma string de comprimento variável em uma string de 32 caracteres que é uma representação de texto do valor hexadecimal de uma soma de verificação de 128 bits.

Sintaxe

```
MD5(string)
```

Argumentos

string

Uma string de comprimento variável.

Tipo de retorno

A função MD5 retorna uma string 32 caracteres que é uma representação de texto do valor hexadecimal de uma soma de verificação de 128 bits.

Exemplos

O seguinte exemplo mostra o valor de 128 bits para a string "Amazon Redshift":

```
select md5('Amazon Redshift');
md5
----------------------------------
f7415e33f972c03abd4f3fed36748f7a
(1 row)
```

Função SHA

Sinônimo da função SHA1.

Consulte Função SHA1.

Função SHA1

A função SHA1 usa a função de hash criptográfica SHA1 para converter uma string de comprimento variável em uma string de 40 caracteres que é uma representação de texto do valor hexadecimal de uma soma de verificação de 160 bits.

Sintaxe

SHA1 é um sinônimo de Função SHA e Função FUNC_SHA1.

```
SHA1(string)
```

Argumentos

string

Uma string de comprimento variável.

Tipo de retorno

A função SHA1 retorna uma string 40 caracteres que é uma representação de texto do valor hexadecimal de uma soma de verificação de 160 bits.

Exemplo

O seguinte exemplo retorna o valor de 160 bits para a palavra "Amazon Redshift":

```
select sha1('Amazon Redshift');
```

Função SHA2

A função SHA2 usa a função de hash criptográfica SHA2 para converter uma string de comprimento variável em uma string caracteres. A string de caracteres é uma representação de texto do valor hexadecimal da soma de verificação com o número especificado de bits.

Sintaxe

```
SHA2(string, bits)
```

Argumentos

string

Uma string de comprimento variável.

integer

O número de bits nas funções de hash. Os valores válidos são 0 (igual a 256), 224, 256, 384 e 512.

Tipo de retorno

A função SHA2 retorna uma string de caracteres que é uma representação de texto do valor hexadecimal da soma de verificação ou uma string vazia se o número de bits for inválido.

Exemplo

O seguinte exemplo retorna o valor de 256 bits para a palavra "Amazon Redshift":

```
select sha2('Amazon Redshift', 256);
```

MURMUR3_32_HASH

A função MURMUR3_32_HASH calcula o hash não criptográfico Murmur3A de 32 bits para todos os tipos de dados comuns, incluindo tipos numéricos e de string.

Sintaxe

```
MURMUR3_32_HASH(value [, seed])
```

Argumentos

valor

O valor de entrada para aplicar o hash. O Amazon Redshift faz hash da representação binária do valor de entrada. Esse comportamento é semelhante a Função FNV_HASH, mas o valor é convertido na representação binária especificada pela especificação de hash Murmur3 de 32 bits do Apache Iceberg.

seed

A semente INT da função de hash. Esse argumento é opcional. Se não for fornecida, o Amazon Redshift usará a semente padrão de 0. Isso permite combinar o hash de várias colunas sem conversões nem concatenações.

Tipo de retorno

A função retorna um INT.

Exemplo

Os exemplos a seguir retornam o hash Murmur3 de um número, a string “Amazon Redshift” e a concatenação dos dois.

```
select MURMUR3_32_HASH(1);

    MURMUR3_32_HASH
----------------------
 1392991556
```

```
(1 row)
```

```
select MURMUR3_32_HASH('Amazon Redshift');

    MURMUR3_32_HASH
----------------------
 -1563580564
(1 row)
```

```
select MURMUR3_32_HASH('Amazon Redshift', MURMUR3_32_HASH(1));

    MURMUR3_32_HASH
----------------------
 -1346554171
(1 row)
```

Observações de uso

Para calcular o hash de uma tabela com várias colunas, é possível calcular o hash Murmur3 da primeira coluna e transmiti-lo como uma semente para o hash da segunda coluna. Depois, ele passa o hash Murmur3 da segunda coluna como uma semente para o hash da terceira coluna.

O exemplo a seguir cria sementes para aplicar o hash a uma tabela com várias colunas.

```
select MURMUR3_32_HASH(column_3, MURMUR3_32_HASH(column_2, MURMUR3_32_HASH(column_1)))
 from sample_table;
```

A mesma propriedade pode ser usada para calcular o hash de uma concatenação de strings.

```
select MURMUR3_32_HASH('abcd');

   MURMUR3_32_HASH
----------------------
 1139631978
(1 row)
```

```
select MURMUR3_32_HASH('cd', MURMUR3_32_HASH('ab'));

   MURMUR3_32_HASH
----------------------
```

```
 1711522338
(1 row)
```

A função de hash usa o tipo de entrada para determinar o número de bytes para hash. Use a fundição para impor um tipo específico, se necessário.

Os exemplos a seguir usam diferentes tipos de entrada para produzir resultados diferentes.

```
select MURMUR3_32_HASH(1, MURMUR3_32_HASH(1));

   MURMUR3_32_HASH
---------------------
 -1193428387
(1 row)
```

```
select MURMUR3_32_HASH(1);

   MURMUR3_32_HASH
----------------------
 1392991556
(1 row)
```

```
select MURMUR3_32_HASH(1, MURMUR3_32_HASH(2));

   MURMUR3_32_HASH
----------------------
 1179621905
(1 row)
```

Funções HyperLogLog

Veja a seguir descrições das funções HyperLogLog para SQL compatíveis com o Amazon Redshift.

Tópicos

- Função HLL
- Função HLL_CREATE_SKETCH
- Função HLL_CARDINALITY
- Função HLL_COMBINE
- Função HLL_COMBINE_SKETCHES

Função HLL

A função HLL retorna a cardinalidade HyperLogLog dos valores de expressão de entrada. A função HLL funciona com quaisquer tipos de dados, exceto o tipo de dados HLLSKETCH. A função HLL ignora valores NULL. Quando não há linhas em uma tabela ou todas as linhas são NULL, a cardinalidade resultante é 0.

Sintaxe

```
HLL (aggregate_expression)
```

Argumento

aggregate_expression

Qualquer expressão válida que forneça o valor a uma agregação, como um nome de coluna. Esta função é compatível com qualquer tipo de dados como entrada, exceto HLLSKETCH, GEOMETRY, GEOGRAPHY e VARBYTE.

Tipo de retorno

A função HLL retorna um valor BIGINT ou INT8.

Exemplos

O seguinte exemplo retorna a cardinalidade da coluna `an_int` na tabela `a_table`.

```
CREATE TABLE a_table(an_int INT);
INSERT INTO a_table VALUES (1), (2), (3), (4);

SELECT hll(an_int) AS cardinality FROM a_table;
cardinality
-------------
4
```

Função HLL_CREATE_SKETCH

A função HLL_CREATE_SKETCH retorna um tipo de dados HLLSKETCH que encapsula os valores de expressão de entrada. A função HLL_CREATE_SKETCH funciona com qualquer tipo de dados e

ignora valores NULL. Quando não há linhas em uma tabela ou todas as linhas são NULL, o esboço resultante não tem pares de valor de índice, como `{"version":1,"logm":15,"sparse":{"indices":[],"values":[]}}`.

Sintaxe

```
HLL_CREATE_SKETCH (aggregate_expression)
```

Argumento

aggregate_expression

Qualquer expressão válida que forneça o valor a uma agregação, como um nome de coluna. Valores NULL são ignorados. Esta função é compatível com qualquer tipo de dados como entrada, exceto HLLSKETCH, GEOMETRY, GEOGRAPHY e VARBYTE.

Tipo de retorno

A função HLL_CREATE_SKETCH retorna um valor HLLSKETCH.

Exemplos

O exemplo a seguir retorna o tipo HLLSKETCH para coluna `an_int` na tabela `a_table`. Um objeto JSON é usado para representar um esboço do HyperLogLog esparso ao importar, exportar ou imprimir esboços. Uma representação de string (no formato Base64) é usada para representar um esboço de HyperLogLog denso.

```
CREATE TABLE a_table(an_int INT);
INSERT INTO a_table VALUES (1), (2), (3), (4);

SELECT hll_create_sketch(an_int) AS sketch FROM a_table;
sketch
-----------------------------------------------------------------------------------------------
{"version":1,"logm":15,"sparse":{"indices":
[20812342,20850007,22362299,47158030],"values":[1,2,1,1]}}
(1 row)
```

Função HLL_CARDINALITY

A função HLL_CARDINALITY retorna a cardinalidade do tipo de dados HLLSKETCH de entrada.

Sintaxe

```
HLL_CARDINALITY (hllsketch_expression)
```

Argumento

hllsketch_expression

Qualquer expressão válida que avalia para um tipo HLLSKETCH, tal como um nome de coluna. O valor de entrada é o tipo de dados HLLSKETCH.

Tipo de retorno

A função HLL_CARDINALITY retorna um valor BIGINT ou INT8.

Exemplos

O seguinte exemplo retorna a cardinalidade da coluna `sketch` na tabela `hll_table`.

```
CREATE TABLE a_table(an_int INT, b_int INT);
INSERT INTO a_table VALUES (1,1), (2,1), (3,1), (4,1), (1,2), (2,2), (3,2), (4,2),
 (5,2), (6,2);

CREATE TABLE hll_table (sketch HLLSKETCH);
INSERT INTO hll_table select hll_create_sketch(an_int) from a_table group by b_int;

SELECT hll_cardinality(sketch) AS cardinality FROM hll_table;
cardinality
-------------
6
4
(2 rows)
```

Função HLL_COMBINE

A função agregada HLL_COMBINE retorna um tipo de dados HLLSKETCH que combina todos os valores HLLSKETCH de entrada.

A combinação de dois ou mais esboços HyperLogLog é um novo HLLSKETCH que encapsula informações sobre a união dos valores distintos que cada esboço de entrada representa. Depois de

combinar esboços, o Amazon Redshift extrai a cardinalidade da união de dois ou mais conjuntos de dados. Para obter mais informações sobre como combinar vários esboços, consulte Exemplo: retorna um esboço do HyperLogLog da combinação de vários esboços.

Sintaxe

```
HLL_COMBINE (hllsketch_expression)
```

Argumento

hllsketch_expression

Qualquer expressão válida que avalia para um tipo HLLSKETCH, tal como um nome de coluna. O valor de entrada é o tipo de dados HLLSKETCH.

Tipo de retorno

A função HLL_COMBINE retorna um tipo HLLSKETCH.

Exemplos

O exemplo a seguir retorna os valores HLLSKETCH combinados na tabela `hll_table`.

```
CREATE TABLE a_table(an_int INT, b_int INT);
INSERT INTO a_table VALUES (1,1), (2,1), (3,1), (4,1), (1,2), (2,2), (3,2), (4,2),
 (5,2), (6,2);

CREATE TABLE hll_table (sketch HLLSKETCH);
INSERT INTO hll_table select hll_create_sketch(an_int) from a_table group by b_int;

SELECT hll_combine(sketch) AS sketches FROM hll_table;
sketches
--------------------------------------------------------------------------------------------------
{"version":1,"logm":15,"sparse":{"indices":
[20812342,20850007,22362299,40314817,42650774,47158030],"values":[1,2,1,3,2,1]}}
(1 row)
```

Função HLL_COMBINE_SKETCHES

A função HLL_COMBINE_SKETCHES é uma função escalar que recebe como entrada dois valores HLLSKETCH e combina-os em um único HLLSKETCH.

A combinação de dois ou mais esboços HyperLogLog é um novo HLLSKETCH que encapsula informações sobre a união dos valores distintos que cada esboço de entrada representa.

Sintaxe

```
HLL_COMBINE_SKETCHES (hllsketch_expression1, hllsketch_expression2)
```

Argumento

hllsketch_expression1 e hllsketch_expression2

Qualquer expressão válida que avalia para um tipo HLLSKETCH, tal como um nome de coluna.

Tipo de retorno

A função HLL_COMBINE_SKETCHES retorna um tipo HLLSKETCH.

Exemplos

O exemplo a seguir retorna os valores HLLSKETCH combinados na tabela `hll_table`.

```
WITH tbl1(x, y)
     AS (SELECT Hll_create_sketch(1),
                Hll_create_sketch(2)
         UNION ALL
         SELECT Hll_create_sketch(3),
                Hll_create_sketch(4)
         UNION ALL
         SELECT Hll_create_sketch(5),
                Hll_create_sketch(6)
         UNION ALL
         SELECT Hll_create_sketch(7),
                Hll_create_sketch(8)),
     tbl2(x, y)
     AS (SELECT Hll_create_sketch(9),
                Hll_create_sketch(10)
         UNION ALL
         SELECT Hll_create_sketch(11),
                Hll_create_sketch(12)
         UNION ALL
         SELECT Hll_create_sketch(13),
```

```
                Hll_create_sketch(14)
        UNION ALL
        SELECT Hll_create_sketch(15),
                Hll_create_sketch(16)
        UNION ALL
        SELECT Hll_create_sketch(NULL),
                Hll_create_sketch(NULL)),
     tbl3(x, y)
     AS (SELECT *
         FROM   tbl1
         UNION ALL
         SELECT *
         FROM   tbl2)
SELECT Hll_combine_sketches(x, y)
FROM   tbl3;
```

Funções JSON

Tópicos

- Função JSON_PARSE
- Função CAN_JSON_PARSE
- Função JSON_SERIALIZE
- Função JSON_SERIALIZE_TO_VARBYTE
- Funções JSON baseadas em texto

> **Note**
>
> Recomendamos usar as seguintes funções para trabalhar com JSON:
>
> - Função JSON_PARSE
> - Função CAN_JSON_PARSE
> - Função JSON_SERIALIZE
> - Função JSON_SERIALIZE_TO_VARBYTE
>
> Com o JSON_PARSE, basta converter o texto JSON em um valor do tipo SUPER uma vez na ingestão, depois disso, é possível operar com os valores SUPER. O Amazon Redshift analisa valores SUPER com maior eficiência do que VARCHAR, que é a saída para as

funções JSON baseadas em texto. Para ter mais informações sobre como trabalhar com o tipo de dado SUPER, consulte Dados semiestruturados no Amazon Redshift.

Quando você precisa armazenar um conjunto relativamente pequeno de pares de valores de chave, você pode economizar espaço armazenando os dados em formato JSON. Como strings JSON podem ser armazenados em uma única coluna, o uso de JSON pode ser mais eficiente que armazenar seus dados em formato tabular. Por exemplo, suponha que você tenha uma tabela esparsa onde você precise ter muitas colunas para representar totalmente todos os atributos possíveis, mas a maioria dos valores da coluna são NULL para qualquer linha ou coluna específica. Ao usar JSON para armazenamento, você pode ser capaz de armazenar os dados para uma linha em pares de valor:chave em uma única string JSON e eliminar as colunas da tabela esparsamente preenchida.

Além disso, é possível facilmente modificar strings JSON para armazenar pares de chave-valor quando o esquema JSON muda sem a necessidade de adicionar colunas a uma tabela.

Recomendamos usar JSON frugalmente. JSON não é uma boa escolha para armazenar conjuntos de dados maiores porque, ao armazenar dados díspares em uma única coluna, o JSON não usa a arquitetura de armazenamento de coluna do Amazon Redshift. Embora o Amazon Redshift ofereça suporte a funções JSON em colunas CHAR e VARCHAR, recomendamos usar SUPER para processar dados no formato de serialização JSON. SUPER usa uma representação sem esquema pós-análise que pode consultar dados hierárquicos de forma eficiente. Para obter mais informações sobre tipos de dados SUPER, consulte Dados semiestruturados no Amazon Redshift.

JSON usa strings de texto codificadas por UTF-8, portanto strings JSON podem ser armazenadas como tipos de dados CHAR ou VARCHAR.

As strings JSON devem ser adequadamente formatadas como JSON de acordo com as seguintes regras:

- O nível JSON raiz pode ser um objeto JSON ou uma matriz JSON. Um objeto JSON é um conjunto desordenado de pares de valor:chave separados por vírgula cercado por chaves.

  Por exemplo, `{"one":1, "two":2}` .
- Uma matriz JSON é um conjunto ordenado de valores separados por vírgula cercado por parênteses.

  Um exemplo é o seguinte: : `["first", {"one":1}, "second", 3, null]`

- Matrizes JSON usam um índice baseado em zero; o primeiro elemento em uma matriz fica na posição 0. Em um par de valores-chave JSON, a chave é uma string entre aspas duplas.
- Um valor JSON pode ser qualquer um dos seguintes:
  - Objeto JSON
  - array
  - string
    - Representado com o uso de aspas duplas.
  - número
    - Inclui números inteiros, decimais e flutuantes.
  - booleano
  - null
- Objetos vazios e matrizes vazias são valores JSON válidos.
- Os campos JSON diferenciam maiúsculas e minúsculas.
- O espaço em branco entre elementos estruturais JSON (tal como `{ }, [ ]`) é ignorado.

As funções JSON do Amazon Redshift o comando COPY do Amazon Redshift usam os mesmos métodos para trabalhar com dados JSON formatados. Para obter mais informações sobre como trabalhar com JSON, consulte COPY no formato JSON

Função JSON_PARSE

A função JSON_PARSE analisa dados no formato JSON e os converte na representação SUPER.

Para ingerir no tipo de dados SUPER usando o comando INSERT ou UPDATE, use a função JSON_PARSE. Quando você usa JSON_PARSE() para analisar strings JSON em valores SUPER, determinadas restrições se aplicam. Para obter informações adicionais, consulte Opções de análise para SUPER.

Sintaxe

```
JSON_PARSE( {json_string | binary_value} )
```

Argumentos

json_string

Uma expressão que retorna JSON serializado como um tipo VARBYTE ou VARCHAR.

binary_value

Um valor binário do tipo VARBYTE.

Tipo de retorno

SUPER

Exemplos

Para converter a matriz JSON `[10001,10002,"abc"]` no tipo de dados SUPER, use o exemplo a seguir.

```
SELECT JSON_PARSE('[10001,10002,"abc"]');

+---------------------+
|     json_parse      |
+---------------------+
| [10001,10002,"abc"] |
+---------------------+
```

Para garantir que a função converteu a matriz JSON no tipo de dados SUPER, use o exemplo a seguir. Para obter mais informações, consulte . Função JSON_TYPEOF

```
SELECT JSON_TYPEOF(JSON_PARSE('[10001,10002,"abc"]'));

+-------------+
| json_typeof |
+-------------+
| array       |
+-------------+
```

Função CAN_JSON_PARSE

A função CAN_JSON_PARSE analisa dados no formato JSON e retorna `true` se o resultado pode ser convertido em um valor SUPER usando a função JSON_PARSE.

Sintaxe

```
CAN_JSON_PARSE( {json_string | binary_value} )
```

Argumentos

json_string

Uma expressão que exibe JSON serializado no formulário VARCHAR.

binary_value

Um valor binário do tipo VARBYTE.

Tipo de retorno

BOOLEAN

Observações de uso

- CAN_JSON_PARSE exibe falso para strings vazias. Exibirá NULL se o argumento de entrada for nulo.

Exemplos

O exemplo a seguir mostra CAN_JSON_PARSE em execução em uma matriz JSON formada corretamente usando uma condição CASE. Ele exibe verdadeiro, então o Amazon Redshift executa a função JSON_PARSE no valor de exemplo.

```
SELECT CASE
            WHEN CAN_JSON_PARSE('[10001,10002,"abc"]')
            THEN JSON_PARSE('[10001,10002,"abc"]')
        END;

 case
---------------------
'[10001,10002,"abc"]'
```

O exemplo a seguir mostra CAN_JSON_PARSE em execução em um valor que não está no formato JSON usando uma condição CASE. Ele exibe falso, então o Amazon Redshift exibe o segmento na cláusula ELSE da condição CASE.

```
SELECT CASE
            WHEN CAN_JSON_PARSE('This is a string.')
            THEN JSON_PARSE('This is a string.')
            ELSE 'This is not JSON.'
        END;

 case
--------------------
"This is not JSON."
```

Função JSON_SERIALIZE

A função JSON_SERIALIZE serializa uma expressão SUPER em representação JSON textual para seguir RFC 8259. Para obter mais informações, consulte O formato de intercâmbio de dados JavaScript Object Notation (JSON).

O limite de tamanho SUPER é aproximadamente o mesmo que o limite de bloco, e o limite de VARCHAR é menor do que o limite de tamanho SUPER. Portanto, a função JSON_SERIALIZE retorna um erro quando o formato JSON excede o limite de VARCHAR do sistema. Para verificar o tamanho de uma expressão SUPER, consulte a função JSON_SIZE.

Sintaxe

```
JSON_SERIALIZE(super_expression)
```

Argumentos

super_expression

Uma expressão ou coluna SUPER.

Tipo de retorno

VARCHAR

> **Note**
>
> O valor de VARCHAR retornado é sempre uma string JSON não nula. Se super_expression for NULL, JSON_SERIALIZE retornará a string JSON `'null'`.

Exemplos

Para serializar um valor SUPER para uma string, use o exemplo a seguir.

```
SELECT JSON_SERIALIZE(JSON_PARSE('[10001,10002,"abc"]'));

+---------------------+
|   json_serialize    |
+---------------------+
| [10001,10002,"abc"] |
+---------------------+
```

Função JSON_SERIALIZE_TO_VARBYTE

A função JSON_SERIALIZE_TO_VARBYTE converte um valor SUPER em uma string JSON semelhante a JSON_SERIALIZE(), mas armazenada em um valor VARBYTE.

Sintaxe

```
JSON_SERIALIZE_TO_VARBYTE(super_expression)
```

Argumentos

super_expression

Uma expressão ou coluna SUPER.

Tipo de retorno

VARBYTE

Exemplos

Para serializar um valor SUPER e retorna o resultado no formato VARBYTE, use o exemplo a seguir.

```
SELECT JSON_SERIALIZE_TO_VARBYTE(JSON_PARSE('[10001,10002,"abc"]'));

+----------------------------------------+
|       json_serialize_to_varbyte        |
+----------------------------------------+
| 5b31303030312c31303030322c22616263225d |
```

```
+----------------------------------------+
```

Para serializar um valor SUPER e transmitir o resultado no formato VARCHAR, use o exemplo a seguir. Para obter mais informações, consulte Função CAST.

```
SELECT CAST((JSON_SERIALIZE_TO_VARBYTE(JSON_PARSE('[10001,10002,"abc"]'))) AS VARCHAR);

+---------------------------+
| json_serialize_to_varbyte |
+---------------------------+
| [10001,10002,"abc"]       |
+---------------------------+
```

Funções JSON baseadas em texto

As funções nesta seção analisam valores JSON como VARCHAR. Para analisar JSON, recomendamos usar as funções a seguir, que analisam valores JSON como SUPER. O Amazon Redshift analisa os valores SUPER com maior eficiência do que o VARCHAR.

- Função JSON_PARSE
- Função CAN_JSON_PARSE
- Função JSON_SERIALIZE
- Função JSON_SERIALIZE_TO_VARBYTE

Tópicos

- Função IS_VALID_JSON
- Função IS_VALID_JSON_ARRAY
- Função JSON_ARRAY_LENGTH
- Função JSON_EXTRACT_ARRAY_ELEMENT_TEXT
- Função JSON_EXTRACT_PATH_TEXT

Função IS_VALID_JSON

> **Note**
>
> CAN_JSON_PARSE e suas funções associadas analisam valores JSON como SUPER, que o Amazon Redshift analisa com maior eficiência do que VARCHAR.

Em vez de usar IS_VALID_JSON, recomendamos validar suas strings JSON usando o Função CAN_JSON_PARSE.

A função IS_VALID_JSON valida uma string JSON. A função retornará um valor booliano `true` se a string tiver um formato JSON válido ou `false` se a string for inválida. Para validar uma matriz JSON, use Função IS_VALID_JSON_ARRAY

Para obter mais informações, consulte Funções JSON.

Sintaxe

```
IS_VALID_JSON('json_string')
```

Argumentos

json_string

Uma string ou uma expressão que retorna uma string JSON.

Tipo de retorno

BOOLEAN

Exemplos

Para criar uma tabela e inserir strings JSON para fins de teste, use o exemplo a seguir.

```
CREATE TABLE test_json(id int IDENTITY(0,1), json_strings VARCHAR);

-- Insert valid JSON strings --
INSERT INTO test_json(json_strings) VALUES
('{"a":2}'),
('{"a":{"b":{"c":1}}}'),
('{"a": [1,2,"b"]}');

-- Insert invalid JSON strings --
INSERT INTO test_json(json_strings) VALUES
('{{}}'),
('{1:"a"}'),
```

```
('[1,2,3]');
```

Para validar as strings no exemplo anterior, use o exemplo a seguir.

```
SELECT id, json_strings, IS_VALID_JSON(json_strings)
FROM test_json
ORDER BY id;

+----+---------------------+---------------+
| id |    json_strings     | is_valid_json |
+----+---------------------+---------------+
|  0 | {"a":2}             | true          |
|  4 | {"a":{"b":{"c":1}}} | true          |
|  8 | {"a": [1,2,"b"]}    | true          |
| 12 | {{}}                | false         |
| 16 | {1:"a"}             | false         |
| 20 | [1,2,3]             | false         |
+----+---------------------+---------------+
```

Função IS_VALID_JSON_ARRAY

> **Note**
>
> O JSON_PARSE e suas funções associadas analisam os valores JSON como SUPER, que o Amazon Redshift analisa com maior eficiência do que o VARCHAR.
> Em vez de usar IS_VALID_JSON_ARRAY, recomendamos que você analise suas strings JSON usando o Função JSON_PARSE para ter um valor SUPER. Depois, use a função Função IS_ARRAY para confirmar se a matriz está formada corretamente.

A função IS_VALID_JSON_ARRAY valida um array JSON. A função retornará um valor booliano `true`, se a matriz tiver um formato JSON válido, ou `false`, se a matriz for inválida. Para validar uma string JSON, use Função IS_VALID_JSON

Para obter mais informações, consulte Funções JSON.

Sintaxe

```
IS_VALID_JSON_ARRAY('json_array')
```

Argumentos

json_array

Uma string ou uma expressão que retorna uma matriz JSON.

Tipo de retorno

BOOLEAN

Exemplos

Para criar uma tabela e inserir strings JSON para fins de teste, use o exemplo a seguir.

```
CREATE TABLE test_json_arrays(id int IDENTITY(0,1), json_arrays VARCHAR);

-- Insert valid JSON array strings --
INSERT INTO test_json_arrays(json_arrays)
VALUES('[]'),
('["a","b"]'),
('["a",["b",1,["c",2,3,null]]]');

-- Insert invalid JSON array strings --
INSERT INTO test_json_arrays(json_arrays)
VALUES('{"a":1}'),
('a'),
('[1,2,]');
```

Para validar as strings no exemplo anterior, use o exemplo a seguir.

```
SELECT json_arrays, IS_VALID_JSON_ARRAY(json_arrays)
FROM test_json_arrays ORDER BY id;

+------------------------------+---------------------+
|         json_arrays          | is_valid_json_array |
+------------------------------+---------------------+
| []                           | true                |
| ["a","b"]                    | true                |
| ["a",["b",1,["c",2,3,null]]] | true                |
| {"a":1}                      | false               |
| a                            | false               |
| [1,2,]                       | false               |
```

```
+-----------------------------+--------------------+
```

Função JSON_ARRAY_LENGTH

> **Note**
>
> O JSON_PARSE e suas funções associadas analisam os valores JSON como SUPER, que o Amazon Redshift analisa com maior eficiência do que o VARCHAR.
> Em vez de usar JSON_ARRAY_LENGTH, recomendamos que você analise suas strings JSON usando o Função JSON_PARSE para ter um valor SUPER. Depois, use o Função GET_ARRAY_LENGTH para ter o comprimento da matriz.

A função JSON_ARRAY_LENGTH retorna o número de elementos no array externo de uma string JSON. Se o argumento null_if_invalid for definido como `true` e a string JSON for inválida, a função retornará `NULL`, em vez de retornar um erro.

Para obter mais informações, consulte Funções JSON.

Sintaxe

```
JSON_ARRAY_LENGTH('json_array' [, null_if_invalid ] )
```

Argumentos

json_array

Uma matriz JSON adequadamente formatada.

null_if_invalid

(Opcional) Um valor `BOOLEAN` que especifica se `NULL` será ou não retornado caso a string de entrada JSON seja inválida, em vez de retornar um erro. Para retornar `NULL` se JSON for inválido, especifique `true` (`t`). Para retornar um erro se JSON for inválido, especifique `false` (`f`). O padrão é `false`.

Tipo de retorno

`INTEGER`

Exemplos

Para retornar o número de elementos na matriz, use o exemplo a seguir.

```
SELECT JSON_ARRAY_LENGTH('[11,12,13,{"f1":21,"f2":[25,26]},14]');

+-------------------+
| json_array_length |
+-------------------+
|                 5 |
+-------------------+
```

Para retornar um erro porque JSON é inválido, use o exemplo a seguir.

```
SELECT JSON_ARRAY_LENGTH('[11,12,13,{"f1":21,"f2":[25,26]},14');

ERROR: invalid json array object [11,12,13,{"f1":21,"f2":[25,26]},14
```

Para definir null_if_invalid como true a fim de que a instrução retorne NULL, em vez de retornar um erro para o JSON inválido, use o exemplo a seguir.

```
SELECT JSON_ARRAY_LENGTH('[11,12,13,{"f1":21,"f2":[25,26]},14',true);

+-------------------+
| json_array_length |
+-------------------+
| NULL              |
+-------------------+
```

Função JSON_EXTRACT_ARRAY_ELEMENT_TEXT

> **Note**
>
> O JSON_PARSE e suas funções associadas analisam os valores JSON como SUPER, que o Amazon Redshift analisa com maior eficiência do que o VARCHAR.
> Em vez de usar JSON_EXTRACT_ARRAY_ELEMENT_TEXT, recomendamos que você analise suas strings JSON usando o Função JSON_PARSE para ter um valor SUPER. Depois, consulte o elemento desejado usando o índice de matriz, com a sintaxe `value[element position]`. Para ter mais informações sobre como consultar elementos de matriz em valores SUPER, acesse Consultar dados semiestruturados.

A função JSON_EXTRACT_ARRAY_ELEMENT_TEXT retorna um elemento de array JSON no array mais externa de uma string JSON, usando um índice baseado em zero. O primeiro elemento em uma matriz fica na posição 0. Se o índice for negativo ou estiver fora do limite, JSON_EXTRACT_ARRAY_ELEMENT_TEXT retornará `NULL`. Se o argumento null_if_invalid for definido como `TRUE` e a string JSON for inválida, a função retornará `NULL`, em vez de retornar um erro.

Para obter mais informações, consulte Funções JSON.

Sintaxe

```
JSON_EXTRACT_ARRAY_ELEMENT_TEXT('json string', pos [, null_if_invalid ] )
```

Argumentos

json_string

Uma string JSON adequadamente formatada.

pos

Um `INTEGER` que representa o índice do elemento da matriz a ser retornado, usando um índice de matriz baseado em zero.

null_if_invalid

(Opcional) Um valor `BOOLEAN` que especifica se `NULL` será ou não retornado caso a string de entrada JSON seja inválida, em vez de retornar um erro. Para retornar `NULL` se JSON for inválido, especifique `true` (`t`). Para retornar um erro se JSON for inválido, especifique `false` (`f`). O padrão é `false`.

Tipo de retorno

`VARCHAR`

Uma string `VARCHAR` representando o elemento da matriz JSON referido por pos.

Exemplos

Para retornar o elemento da posição 2 da matriz, que é o terceiro elemento de um índice de matriz baseado em zero, use o exemplo a seguir.

```
SELECT JSON_EXTRACT_ARRAY_ELEMENT_TEXT('[111,112,113]', 2);

+--------------------------------+
| json_extract_array_element_text |
+--------------------------------+
|                            113 |
+--------------------------------+
```

Para retornar um erro porque JSON é inválido, use o exemplo a seguir.

```
SELECT JSON_EXTRACT_ARRAY_ELEMENT_TEXT('["a",["b",1,["c",2,3,null,]]]',1);

ERROR: invalid json array object ["a",["b",1,["c",2,3,null,]]]
```

Para definir null_if_invalid como true a fim de que a instrução retorne NULL, em vez de retornar um erro para o JSON inválido, use o exemplo a seguir.

```
SELECT JSON_EXTRACT_ARRAY_ELEMENT_TEXT('["a",["b",1,["c",2,3,null,]]]',1,true);

+--------------------------------+
| json_extract_array_element_text |
+--------------------------------+
| NULL                           |
+--------------------------------+
```

Considere o exemplo de declarações a seguir. Se a string JSON fornecida ou o índice for NULL, JSON_EXTRACT_ARRAY_ELEMENT_TEXT retornará NULL independentemente do valor de qualquer outro parâmetro.

```
--Statement where json_string is NULL.
SELECT json_extract_array_element_text(NULL, 0)

 json_extract_array_element_text
--------------------------------
                           NULL
```

```
--Statement where pos is NULL and json_string is invalid JSON.
SELECT json_extract_array_element_text('invalid_json', NULL);

 json_extract_array_element_text
---------------------------------
                            NULL

--Statement where json_string is NULL and null_if_invalid is FALSE.
SELECT json_extract_array_element_text(NULL, 0, FALSE);

 json_extract_array_element_text
---------------------------------
                            NULL
```

Considere o exemplo de declarações a seguir. Quando null_if_invalid é TRUE, JSON_EXTRACT_ARRAY_ELEMENT_TEXT retorna NULL quando json_string é um JSON inválido. Se null_if_invalid for FALSE ou não estiver definido, a função retornará um erro quando json_string for inválido.

```
--Statement with invalid JSON where null_if_invalid is TRUE.
SELECT json_extract_array_element_text('invalid_json', 0, TRUE);

 json_extract_array_element_text
---------------------------------
                            NULL

--Statement with invalid JSON where null_if_invalid is FALSE.
SELECT json_extract_array_element_text('invalid_json', 0);

ERROR:  JSON parsing error
```

Considere o exemplo a seguir, em que json_string é um JSON válido e pos se refere a um valor JSON `null`. Nesse caso, JSON_EXTRACT_ARRAY_ELEMENT_TEXT retorna NULL, independentemente do valor de null_if_invalid.

```
--Statement selecting a null value.
SELECT json_extract_array_element_text('[null]', 0);

  json_extract_array_element_text
---------------------------------
                             NULL
```

Função JSON_EXTRACT_PATH_TEXT

> **Note**
>
> O JSON_PARSE e suas funções associadas analisam os valores JSON como SUPER, que o Amazon Redshift analisa com maior eficiência do que o VARCHAR.
> Em vez de usar JSON_EXTRACT_PATH_TEXT, recomendamos que você analise suas strings JSON usando o Função JSON_PARSE para ter um valor SUPER. Depois, consulte o elemento desejado usando a sintaxe `value.attribute`. Para ter mais informações sobre como consultar elementos de matriz em valores SUPER, acesse Consultar dados semiestruturados.

A função JSON_EXTRACT_PATH_TEXT retorna o valor para o par de chave-valor referido por uma série de elementos de caminho em uma string JSON. O caminho JSON pode ser aninhado em até cinco níveis de profundidade. Os elementos do caminho diferenciam maiúsculas e minúsculas. Se um elemento do caminho não existir na string JSON, JSON_EXTRACT_PATH_TEXT retornará `NULL`.

Se o argumento null_if_invalid for definido como `TRUE` e a string JSON for inválida, a função retornará `NULL`, em vez de retornar um erro.

JSON_EXTRACT_PATH_TEXT tem um tamanho máximo de dados de 16 MB. Portanto, se algum registro JSON for maior que 16 MB, processá-lo com JSON_EXTRACT_PATH_TEXT vai gerar um erro.

Para obter informações sobre outras funções JSON, consulte Funções JSON. Para obter mais informações sobre como trabalhar com JSON, consulte COPY no formato JSON.

Sintaxe

```
JSON_EXTRACT_PATH_TEXT('json_string', 'path_elem' [,'path_elem'[, …] ]
 [, null_if_invalid ] )
```

Argumentos

json_string

Uma string JSON adequadamente formatada.

path_elem

Um elemento de caminho em uma string JSON. Um elemento de caminho é obrigatório. Elementos de caminho adicionais podem ser especificados com até cinco níveis de profundidade.

null_if_invalid

(Opcional) Um valor BOOLEAN que especifica se NULL será ou não retornado caso a string de entrada JSON seja inválida, em vez de retornar um erro. Para retornar NULL se JSON for inválido, especifique TRUE (t). Para retornar um erro se JSON for inválido, especifique FALSE (f). O padrão é FALSE.

Em uma string JSON, o Amazon Redshift reconhece \n como um caractere de nova linha e \t como um caractere de tabulação. Para carregar uma barra invertida, use uma barra invertida de escape (\\). Para obter mais informações, consulte Caracteres de escape em JSON.

Tipo de retorno

VARCHAR

Uma string VARCHAR representando o valor JSON referido pelos elementos de caminho.

Exemplos

Para retornar o valor do caminho 'f4', 'f6', use o exemplo a seguir.

```
SELECT JSON_EXTRACT_PATH_TEXT('{"f2":{"f3":1},"f4":{"f5":99,"f6":"star"}}','f4', 'f6');

+------------------------+
| json_extract_path_text |
+------------------------+
| star                   |
+------------------------+
```

Para retornar um erro porque JSON é inválido, use o exemplo a seguir.

```
SELECT JSON_EXTRACT_PATH_TEXT('{"f2":{"f3":1},"f4":{"f5":99,"f6":"star"}','f4', 'f6');

ERROR: invalid json object {"f2":{"f3":1},"f4":{"f5":99,"f6":"star"}
```

Para definir null_if_invalid como TRUE a fim de que a instrução retorne NULL para o JSON inválido em vez de retornar um erro, use o exemplo a seguir.

```
SELECT JSON_EXTRACT_PATH_TEXT('{"f2":{"f3":1},"f4":{"f5":99,"f6":"star"}','f4',
 'f6',true);

+------------------------+
| json_extract_path_text |
+------------------------+
| NULL                   |
+------------------------+
```

Considere o exemplo a seguir, que seleciona o valor para o caminho `'farm', 'barn', 'color'`, em que o valor recuperado está no terceiro nível, use o exemplo a seguir. Esse exemplo é formatado com uma ferramenta JSON Lint para facilitar a leitura.

```
SELECT JSON_EXTRACT_PATH_TEXT('{
    "farm": {
        "barn": {
            "color": "red",
            "feed stocked": true
        }
    }
}', 'farm', 'barn', 'color');
+------------------------+
| json_extract_path_text |
+------------------------+
| red                    |
+------------------------+
```

Para retornar NULL porque o elemento `'color'` está ausente, use o exemplo a seguir. Esse exemplo é formatada com uma ferramenta JSON Lint.

```
SELECT JSON_EXTRACT_PATH_TEXT('{
    "farm": {
        "barn": {}
    }
}', 'farm', 'barn', 'color');

+------------------------+
| json_extract_path_text |
+------------------------+
```

```
| NULL                    |
+-------------------------+
```

Se o JSON for válido, tentar extrair um elemento ausente retornará `NULL`.

Para retornar o valor do caminho `'house', 'appliances', 'washing machine', 'brand'`, use o exemplo a seguir.

```
SELECT JSON_EXTRACT_PATH_TEXT('{
  "house": {
    "address": {
      "street": "123 Any St.",
      "city": "Any Town",
      "state": "FL",
      "zip": "32830"
    },
    "bathroom": {
      "color": "green",
      "shower": true
    },
    "appliances": {
      "washing machine": {
        "brand": "Any Brand",
        "color": "beige"
      },
      "dryer": {
        "brand": "Any Brand",
        "color": "white"
      }
    }
  }
}', 'house', 'appliances', 'washing machine', 'brand');

+-------------------------+
| json_extract_path_text |
+-------------------------+
| Any Brand              |
+-------------------------+
```

O exemplo abaixo cria uma tabela de exemplo e a preenche com valores SUPER e, em seguida, retorna o valor do caminho `'f2'` para ambas as linhas.

```
CREATE TABLE json_example(id INT, json_text SUPER);
```

```
INSERT INTO json_example VALUES
(1, JSON_PARSE('{"f2":{"f3":1},"f4":{"f5":99,"f6":"star"}}')),
(2, JSON_PARSE('{
    "farm": {
        "barn": {
            "color": "red",
            "feed stocked": true
        }
    }
}'));

SELECT * FROM json_example;
id          | json_text
------------+--------------------------------------------
1           | {"f2":{"f3":1},"f4":{"f5":99,"f6":"star"}}
2           | {"farm":{"barn":{"color":"red","feed stocked":true}}}


SELECT id, JSON_EXTRACT_PATH_TEXT(JSON_SERIALIZE(json_text), 'f2') FROM json_example;

id          | json_text
------------+--------------------------------------------
1           | {"f3":1}
2           |
```

Considere o exemplo de declarações a seguir. O path_elem fornecido é NULL, então JSON_EXTRACT_PATH_TEXT retorna NULL independentemente do valor de qualquer outro parâmetro.

```
--Statement where path_elem is NULL and json_string is valid JSON.
SELECT JSON_EXTRACT_PATH_TEXT('{"f2":{"f3":1},"f4":{"f5":99,"f6":"star"}}',NULL);

 json_extract_path_text
------------------------
                  NULL

--Statement where only one path_elem is NULL.
SELECT JSON_EXTRACT_PATH_TEXT('{"f2":{"f3":1},"f4":{"f5":99,"f6":"star"}}','f4',NULL);

 json_extract_path_text
------------------------
                  NULL
```

```
--Statement where path_elem is NULL and json_string is invalid JSON.
SELECT json_extract_path_text('invalid_json', NULL);

 json_extract_path_text
------------------------
                  NULL

--Statement where path_elem is NULL and null_if_invalid is FALSE.
SELECT json_extract_path_text(NULL, 0, FALSE);

 json_extract_path_text
------------------------
                  NULL
```

Considere o exemplo de declarações a seguir. Quando null_if_invalid é TRUE, JSON_EXTRACT_PATH_TEXT retorna NULL quando json_string é um JSON inválido. Se null_if_invalid for FALSE ou não estiver definido, a função retornará um erro quando json_string for inválido.

```
--Statement with invalid JSON where null_if_invalid is TRUE.
SELECT json_extract_path_text('invalid_json', 0, TRUE);

 json_extract_path_text
------------------------
                  NULL

--Statement with invalid JSON where null_if_invalid is FALSE.
SELECT json_extract_path_text('invalid_json', 0, FALSE);

ERROR:  JSON parsing error
```

Considere os exemplos a seguir, em que json_string é um JSON válido e path_elem se refere a um valor JSON `null`. Nesse caso, JSON_EXTRACT_PATH_TEXT retorna NULL. Da mesma forma, quando path_elem se refere a um valor não existente, JSON_EXTRACT_PATH_TEXT retorna NULL, independentemente do valor de null_if_invalid.

```
--Statement selecting a null value.
SELECT json_extract_path_text('[null]', 0);

  json_extract_path_text
-------------------------
```

```
                    NULL

--Statement selecting a non-existing value.
SELECT json_extract_path_text('{}', 'a');

  json_extract_path_text
-------------------------
                    NULL
```

Funções de machine learning

Usando o machine learning (ML) do Amazon Redshift, é possível treinar modelos de ML usando instruções SQL e chamá-los em consultas SQL para previsão. A explicabilidade do modelo do Amazon Redshift ML contém valores relevantes de recursos para ajudar a entender como cada atributo dos dados de treinamento contribui para o resultado previsto.

Veja a seguir descrições das funções de machine learning para SQL compatíveis com o Amazon Redshift.

Tópicos

- Função EXPLAIN_MODEL

Função EXPLAIN_MODEL

A função EXPLAIN_MODEL retorna um tipo de dados SUPER que contém um relatório de explicabilidade do modelo em formato JSON. O relatório de explicabilidade contém informações sobre o valor Shapley para todos os recursos do modelo.

A função EXPLAIN_MODEL atualmente é compatível apenas com os modelos XGBoost AUTO ON ou AUTO OFF.

Quando o relatório de explicabilidade não está disponível, a função retorna status exibindo o andamento do modelo. Entre eles estão `Waiting for training job to complete`, `Waiting for processing job to complete` e `Processing job failed`.

Quando você executa a instrução CREATE MODEL, o estado da explicação se torna `Waiting for training job to complete`. Quando o modelo foi treinado e uma solicitação de explicação é enviada, o estado de explicação se torna `Waiting for processing job to complete`. Quando a explicação do modelo for concluída com êxito, o relatório completo de explicabilidade estará disponível. Caso contrário, o estado se torna `Processing job failed`.

Ao executar a instrução CREATE MODEL, você pode usar o parâmetro opcional MAX_RUNTIME para especificar a quantidade máxima de tempo que o treinamento deve levar. Quando a criação do modelo atingir esse período, o Amazon Redshift interromperá a criação do modelo. Se você atingir esse limite de tempo ao criar um modelo de piloto automático, o Amazon Redshift retornará o melhor modelo até o momento. A explicabilidade do modelo fica disponível depois que o treinamento de modelo é concluído, portanto, se MAX_RUNTIME estiver definido com um período curto, o relatório de explicabilidade poderá não estar disponível. O tempo de treinamento varia e depende da complexidade do modelo, do tamanho dos dados e de outros fatores.

Sintaxe

```
EXPLAIN_MODEL ('schema_name.model_name')
```

Argumento

schema_name

O nome do esquema. Se nenhum schema_name for especificado, será selecionado o esquema atual.

model_name

O nome do modelo. O nome do modelo em um esquema deve ser exclusivo.

Tipo de retorno

A função EXPLAIN_MODEL retorna um tipo de dados SUPER, conforme mostrado a seguir.

```
{"version":"1.0","explanations":{"kernel_shap":{"label0":{"global_shap_values":
{"x0":0.05,"x1":0.10,"x2":0.30,"x3":0.15},"expected_value":0.50}}}}
```

Exemplos

O seguinte exemplo retorna o estado de explicação `waiting for training job to complete`.

```
select explain_model('customer_churn_auto_model');
                 explain_model
-------------------------------------------------------
{"explanations":"waiting for training job to complete"}
(1 row)
```

Quando a explicação do modelo for concluída com êxito, o relatório completo de explicabilidade estará disponível como mostrado a seguir.

```
select explain_model('customer_churn_auto_model');
                                        explain_model
-------------------------------------------------------------------------------------------------
{"version":"1.0","explanations":{"kernel_shap":{"label0":{"global_shap_values":
{"x0":0.05386043365892927,"x1":0.10801289723274592,"x2":0.23227865827017378,"x3":0.067668513394
(1 row)
```

Como a função EXPLAIN_MODEL retorna o tipo de dados SUPER, é possível consultar o relatório de explicabilidade. Ao fazer isso, você pode extrair valores `global_shap_values`, `expected_value` ou valores Shapley específicos de recursos.

O exemplo a seguir extrai `global_shap_values` para o modelo.

```
select json_table.report.explanations.kernel_shap.label0.global_shap_values from
 (select explain_model('customer_churn_auto_model') as report) as json_table;
                                                    global_shap_values
-------------------------------------------------------------------------------------------------
{"state":0.10983770427197151,"account_length":0.1772441398408543,"area_code":0.0862682396863959
(1 row)
```

O exemplo a seguir extrai `global_shap_values` para o recurso x0.

```
select json_table.report.explanations.kernel_shap.label0.global_shap_values.x0 from
 (select explain_model('customer_churn_auto_model') as report) as json_table;
          x0
------------------------
  0.05386043365892927
(1 row)
```

Se o modelo for criado em um esquema específico e você tiver acesso ao modelo criado, poderá consultar a explicação do modelo conforme mostrado a seguir.

```
-- Check the current schema
SHOW search_path;
   search_path
-----------------
  $user, public
(1 row)
```

```
-- If you have the privilege to access the model explanation
-- in `test_schema`
SELECT explain_model('test_schema.test_model_name');
                        explain_model
-------------------------------------------------------------
{"explanations":"waiting for training job to complete"}
(1 row)
```

Funções matemáticas

Tópicos

- Símbolos de operadores matemáticos
- Função ABS
- Função ACOS
- Função ASIN
- Função ATAN
- Função ATAN2
- Função CBRT
- Função CEILING (ou CEIL)
- Função COS
- Função COT
- Função DEGREES
- Função DEXP
- Função DLOG1
- Função DLOG10
- Função EXP
- Função FLOOR
- Função LN
- Função LOG
- Função MOD
- Função PI
- Função POWER

- Função RADIANS
- Função RANDOM
- Função ROUND
- Função SIN
- Função SIGN
- Função SQRT
- Função TAN
- Função TRUNC

Esta seção descreve os operadores matemáticos e funções com suporte no Amazon Redshift.

Símbolos de operadores matemáticos

A tabela a seguir lista os operadores matemáticos compatíveis.

Operadores compatíveis

Operador	Descrição	Exemplo	Resultado
+	adição	2 + 3	5
-	subtração	2 - 3	-1
*	multiplicação	2 x 3	6
/	divisão	4 / 2	2
%	módulo	5 % 4	1
^	exponenciação	2,0 ^ 3,0	8
|/	raiz quadrada	| / 25,0	5
||/	raiz cúbica	|| / 27,0	3

Operador	Descrição	Exemplo	Resultado
@	valor absoluto	@ -5,0	5
<<	deslocamento bitwise à esquerda	1 << 4	16
>>	deslocamento bitwise à direita	8 >> 2	2
&	bitwise e	8 & 2	0

Exemplos

Os exemplos a seguir usam o banco de dados de exemplo de TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para calcular a comissão paga mais uma taxa de manuseio de USD 2,00 para determinada transação, use o exemplo a seguir.

```
SELECT
    commission,
    (commission + 2.00) AS comm
FROM
    sales
WHERE
    salesid = 10000;

+------------+-------+
| commission | comm  |
+------------+-------+
|      28.05 | 30.05 |
+------------+-------+
```

Para calcular 20% do preço de venda para determinada transação, use o exemplo a seguir.

```
SELECT pricepaid, (pricepaid * .20) as twentypct
```

```
FROM sales
WHERE salesid=10000;

+-----------+-----------+
| pricepaid | twentypct |
+-----------+-----------+
|       187 |      37.4 |
+-----------+-----------+
```

Para prever as vendas de ingressos com base em um padrão de crescimento contínuo, use o exemplo a seguir. Neste exemplo, a subconsulta retorna o número de ingressos vendidos em 2008. Esse resultado é multiplicado exponencialmente por uma taxa de crescimento contínuo de 5% por 10 anos.

```
SELECT (SELECT SUM(qtysold) FROM sales, date
WHERE sales.dateid=date.dateid AND year=2008)^((5::float/100)*10) AS qty10years;

+------------------+
|    qty10years    |
+------------------+
| 587.664019657491 |
+------------------+
```

Para encontrar o preço total pago e a comissão para vendas com um ID de data maior ou igual a 2000, use o exemplo a seguir. Então, subtraia a comissão total do preço total pago.

```
SELECT SUM(pricepaid) AS sum_price, dateid,
SUM(commission) AS sum_comm, (SUM(pricepaid) - SUM(commission)) AS value
FROM sales
WHERE dateid >= 2000
GROUP BY dateid
ORDER BY dateid
LIMIT 10;

+-----------+--------+----------+-----------+
| sum_price | dateid | sum_comm |   value   |
+-----------+--------+----------+-----------+
|    305885 |   2000 | 45882.75 | 260002.25 |
|    316037 |   2001 | 47405.55 | 268631.45 |
|    358571 |   2002 | 53785.65 | 304785.35 |
|    366033 |   2003 | 54904.95 | 311128.05 |
|    307592 |   2004 |  46138.8 |  261453.2 |
```

```
|    333484 |   2005 |  50022.6 |  283461.4 |
|    317670 |   2006 |  47650.5 |  270019.5 |
|    351031 |   2007 | 52654.65 | 298376.35 |
|    313359 |   2008 | 47003.85 | 266355.15 |
|    323675 |   2009 | 48551.25 | 275123.75 |
+-----------+--------+----------+-----------+
```

Função ABS

ABS calcula o valor absoluto de um número, que pode ser um literal ou uma expressão que avalie para um número.

Sintaxe

```
ABS(number)
```

Argumentos

Número da

Número ou expressão que avalia para um número. Ele também pode ser do tipo `SMALLINT`, `INTEGER`, `BIGINT`, `DECIMAL`, `FLOAT4`, `FLOAT8` ou `SUPER`.

Tipo de retorno

ABS retorna o mesmo tipo de dados que seu argumento.

Exemplos

Para calcular o valor absoluto de `-38`, use o exemplo a seguir.

```
SELECT ABS(-38);

+-----+
| abs |
+-----+
|  38 |
+-----+
```

Para calcular o valor absoluto de `(14-76)`, use o exemplo a seguir.

```
SELECT ABS(14-76);

+-----+
| abs |
+-----+
|  62 |
+-----+
```

Função ACOS

ACOS é uma função trigonométrica que retorna o arco cosseno de um número. O valor de retorno é em radianos, entre `0` e `PI`.

Sintaxe

```
ACOS(number)
```

Argumentos

Número da

O parâmetro de entrada é um número `DOUBLE PRECISION`.

Tipo de retorno

`DOUBLE PRECISION`

Exemplos

Para retornar o arco cosseno de `-1`, use o exemplo a seguir.

```
SELECT ACOS(-1);

+-------------------+
|       acos        |
+-------------------+
| 3.141592653589793 |
+-------------------+
```

Para converter o arco cosseno de `.5` para o número equivalente de graus, use o exemplo a seguir.

```
SELECT (ACOS(.5) * 180/(SELECT PI())) AS degrees;

+------------------+
|      degrees     |
+------------------+
| 60.00000000000001 |
+------------------+
```

Função ASIN

ASIN é uma função trigonométrica que retorna o arco seno de um número. O valor de retorno é em radianos, entre `PI/2` e `-PI/2`.

Sintaxe

```
ASIN(number)
```

Argumentos

Número da

O parâmetro de entrada é um número `DOUBLE PRECISION`.

Tipo de retorno

`DOUBLE PRECISION`

Exemplos

Para retornar o arco seno de 1, use o exemplo a seguir.

```
SELECT ASIN(1) AS halfpi;

+--------------------+
|       halfpi       |
+--------------------+
| 1.5707963267948966 |
+--------------------+
```

Para converter o arco seno de `.5` para o número equivalente de graus, use o exemplo a seguir.

```
SELECT (ASIN(.5) * 180/(SELECT PI())) AS degrees;

+--------------------+
|      degrees       |
+--------------------+
| 30.000000000000004 |
+--------------------+
```

Função ATAN

ATAN é uma função trigonométrica que retorna a arco tangente de um número. O valor de retorno é em radianos, entre `-PI` e `PI`.

Sintaxe

```
ATAN(number)
```

Argumentos

Número da

O parâmetro de entrada é um número `DOUBLE PRECISION`.

Tipo de retorno

`DOUBLE PRECISION`

Exemplos

Para retornar a arco tangente de 1 e a multiplica por 4, use o exemplo a seguir.

```
SELECT ATAN(1) * 4 AS pi;

+-------------------+
|        pi         |
+-------------------+
| 3.141592653589793 |
+-------------------+
```

Para converter o arco tangente de 1 para o número equivalente de graus, use o exemplo a seguir.

```
SELECT (ATAN(1) * 180/(SELECT PI())) AS degrees;

+---------+
| degrees |
+---------+
|      45 |
+---------+
```

Função ATAN2

ATAN2 é uma função trigonométrica que retorna a arco tangente de um número dividido por outro número. O valor de retorno é em radianos, entre `PI/2` e `-PI/2`.

Sintaxe

```
ATAN2(number1, number2)
```

Argumentos

number1

Um número `DOUBLE PRECISION`.

number2

Um número `DOUBLE PRECISION`.

Tipo de retorno

`DOUBLE PRECISION`

Exemplos

Para retornar a arco tangente de `2/2` e a multiplica por `4`, use o exemplo a seguir.

```
SELECT ATAN2(2,2) * 4 AS PI;

+-------------------+
|        pi         |
+-------------------+
| 3.141592653589793 |
```

```
+-------------------+
```

Para converter o arco tangente de `1/0` (que é avaliado como 0) para o número equivalente de graus, use o exemplo a seguir.

```
SELECT (ATAN2(1,0) * 180/(SELECT PI())) AS degrees;

+---------+
| degrees |
+---------+
|      90 |
+---------+
```

Função CBRT

A função CBRT é uma função matemática que calcula a raiz cúbica de determinado número.

Sintaxe

```
CBRT(number)
```

Argumentos

CBRT usa um número `DOUBLE PRECISION` como um argumento.

Tipo de retorno

`DOUBLE PRECISION`

Exemplos

Os exemplos a seguir usa o banco de dados de exemplo de TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para calcular a raiz cúbica da comissão paga para determinada transação, use o exemplo a seguir.

```
SELECT CBRT(commission) FROM sales WHERE salesid=10000;

+--------------------+
|        cbrt        |
+--------------------+
| 3.0383953904884344 |
```

```
+--------------------+
```

Função CEILING (ou CEIL)

A função CEILING ou CEIL é usada para arredondar um número para o número inteiro seguinte. (A função Função FLOOR arredonda um número para o número inteiro anterior.)

Sintaxe

```
{CEIL | CEILING}(number)
```

Argumentos

Número da

O número ou expressão avaliada como um número. Ele também pode ser do tipo SMALLINT, INTEGER, BIGINT, DECIMAL, FLOAT4, FLOAT8 ou SUPER.

Tipo de retorno

CEILING e CEIL retornam o mesmo tipo de dados que seu argumento.

Quando a entrada é do tipo SUPER, a saída mantém o mesmo tipo dinâmico que a entrada enquanto o tipo estático permanece o tipo SUPER. Quando o tipo dinâmico de SUPER não é um número, o Amazon Redshift retorna um nulo.

Exemplos

Os exemplos a seguir usa o banco de dados de exemplo de TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para calcular o teto da comissão paga para determinada transação de vendas, use o exemplo a seguir.

```
SELECT CEILING(commission) FROM sales
WHERE salesid=10000;

+---------+
| ceiling |
+---------+
|      29 |
```

```
+---------+
```

Função COS

COS é uma função trigonométrica que retorna o cosseno de um número. O valor de retorno é em radianos, entre `-1` e `1`, incluindo ambos.

Sintaxe

```
COS(double_precision)
```

Argumentos

Número da

O parâmetro de entrada é um número `DOUBLE PRECISION`.

Tipo de retorno

A função COS retorna um número `DOUBLE PRECISION`.

Exemplos

Para retornar o cosseno de `0`, use o exemplo a seguir.

```
SELECT COS(0);

+-----+
| cos |
+-----+
|   1 |
+-----+
```

Para retornar o cosseno de `pi`, use o exemplo a seguir.

```
SELECT COS(PI());

+-----+
| cos |
+-----+
|  -1 |
```

```
+-----+
```

Função COT

COT é uma função trigonométrica que retorna a cotangente de um número. O parâmetro de entrada deve ser diferente de zero.

Sintaxe

```
COT(number)
```

Argumento

Número da

O parâmetro de entrada é um número DOUBLE PRECISION.

Tipo de retorno

DOUBLE PRECISION

Exemplos

Para retornar a cotangente de 1, use o exemplo a seguir.

```
SELECT COT(1);

+--------------------+
|        cot         |
+--------------------+
| 0.6420926159343306 |
+--------------------+
```

Função DEGREES

Converte um ângulo em radianos para seu equivalente em graus.

Sintaxe

```
DEGREES(number)
```

Argumento

Número da

O parâmetro de entrada é um número `DOUBLE PRECISION`.

Tipo de retorno

`DOUBLE PRECISION`

Exemplos

Para retornar o grau equivalente de 0,5 radiano, use o exemplo a seguir.

```
SELECT DEGREES(.5);

+-------------------+
|      degrees      |
+-------------------+
| 28.64788975654116 |
+-------------------+
```

Para converter PI radianos em graus, use o exemplo a seguir.

```
SELECT DEGREES(pi());

+---------+
| degrees |
+---------+
|     180 |
+---------+
```

Função DEXP

A função DEXP retorna o valor exponencial em notação científica para um número de precisão dupla. A única diferença entre as funções DEXP e EXP é que o parâmetro para DEXP deve ser um `DOUBLE PRECISION`.

Sintaxe

```
DEXP(number)
```

Argumento

Número da

O parâmetro de entrada é um número `DOUBLE PRECISION`.

Tipo de retorno

`DOUBLE PRECISION`

Exemplo

Os exemplos a seguir usa o banco de dados de exemplo de TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Use a função DEXP para prever as vendas de ingressos com base em um padrão de crescimento contínuo. Neste exemplo, a subconsulta retorna o número de ingressos vendidos em 2008. Esse resultado é multiplicado pelo resultado da função DEXP, que especifica uma taxa de crescimento contínuo de 7% por 10 anos.

```
SELECT (SELECT SUM(qtysold)
FROM sales, date
WHERE sales.dateid=date.dateid
AND year=2008) * DEXP((7::FLOAT/100)*10) qty2010;

+-------------------+
|      qty2010      |
+-------------------+
| 695447.4837722216 |
+-------------------+
```

Função DLOG1

A função DLOG1 retorna o logaritmo natural do parâmetro de entrada. Sinônimo de Função LN.

Função DLOG10

A função DLOG10 retorna o logaritmo de base 10 do parâmetro de entrada.

Sinônimo de Função LOG.

Sintaxe

```
DLOG10(number)
```

Argumento

Número da

O parâmetro de entrada é um número DOUBLE PRECISION.

Tipo de retorno

DOUBLE PRECISION

Exemplo

Para retornar o logaritmo de base 10 do número 100, use o exemplo a seguir.

```
SELECT DLOG10(100);

+--------+
| dlog10 |
+--------+
|      2 |
+--------+
```

Função EXP

A função EXP implementa a função exponencial de uma expressão numérica, ou a base de um logaritmo natural, e, elevada à potência da expressão. A função EXP é o inverso de Função LN.

Sintaxe

```
EXP(expression)
```

Argumento

expressão

A expressão de entrada deve ser um tipo de dados INTEGER, DECIMAL ou DOUBLE PRECISION.

Tipo de retorno

`DOUBLE PRECISION`

Exemplo

Os exemplos a seguir usa o banco de dados de exemplo de TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Use a função EXP para prever as vendas de ingressos com base em um padrão de crescimento contínuo. Neste exemplo, a subconsulta retorna o número de ingressos vendidos em 2008. Esse resultado é multiplicado pelo resultado da função EXP, que especifica uma taxa de crescimento contínuo de 7% por 10 anos.

```
SELECT (SELECT SUM(qtysold)
FROM sales, date
WHERE sales.dateid=date.dateid
AND year=2008) * EXP((7::FLOAT/100)*10) qty2018;

+-------------------+
|      qty2018      |
+-------------------+
| 695447.4837722216 |
+-------------------+
```

Função FLOOR

A função FLOOR arredonda um número para o número inteiro anterior.

Sintaxe

```
FLOOR(number)
```

Argumento

Número da

O número ou expressão avaliada como um número. Ele também pode ser do tipo `SMALLINT`, `INTEGER`, `BIGINT`, `DECIMAL`, `FLOAT4`, `FLOAT8` ou `SUPER`.

Tipo de retorno

FLOOR retorna o mesmo tipo de dados que seu argumento.

Quando a entrada é do tipo SUPER, a saída mantém o mesmo tipo dinâmico que a entrada enquanto o tipo estático permanece o tipo SUPER. Quando o tipo dinâmico de SUPER não é um número, o Amazon Redshift retorna NULL.

Exemplos

Os exemplos a seguir usam o banco de dados de exemplo de TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para mostrar o valor da comissão paga por determinada transação de vendas antes e depois de usar a função FLOOR, use o exemplo a seguir.

```
SELECT commission
FROM sales
WHERE salesid=10000;

+------------+
| commission |
+------------+
|      28.05 |
+------------+

SELECT FLOOR(commission)
FROM sales
WHERE salesid=10000;

+-------+
| floor |
+-------+
|    28 |
+-------+
```

Função LN

Retorna o logaritmo natural do parâmetro de entrada.

Sinônimo de Função DLOG1.

Sintaxe

```
LN(expression)
```

Argumento

expressão

A coluna ou expressão de destino na qual a função opera.

> **Note**
>
> Esta função retorna um erro para alguns tipos de dados se a expressão fizer referência a uma tabela criada pelo usuário do Amazon Redshift ou a uma tabela de sistema STL ou STV do Amazon Redshift.

As expressões com os seguintes tipos de dados produzem um erro se fizerem referência a uma tabela criada por usuário ou uma tabela de sistema. As expressões com esses tipos de dados executam exclusivamente no nó de liderança:

- `BOOLEAN`
- `CHAR`
- `DATE`
- `DECIMAL` ou `NUMERIC`
- `TIMESTAMP`
- `VARCHAR`

Expressões com os seguintes tipos de dados executam com êxito em tabelas criadas por usuário ou tabelas de sistema STL ou STV:

- `BIGINT`
- `DOUBLE PRECISION`
- `INTEGER`
- `REAL`
- `SMALLINT`

Tipo de retorno

A função LN retorna o mesmo tipo que a expressão de entrada.

Exemplos

Para retornar o logaritmo natural, ou logaritmo de base e do número 2,718281828, use o exemplo a seguir.

```
SELECT LN(2.718281828);

+--------------------+
|         ln         |
+--------------------+
| 0.9999999998311267 |
+--------------------+
```

Observe que a resposta é quase igual a 1.

Os exemplos a seguir usa o banco de dados de exemplo de TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para retornar o logaritmo natural dos valores na coluna userid da tabela USERS, use o exemplo a seguir.

```
SELECT username, LN(userid) FROM users ORDER BY userid LIMIT 10;

+----------+--------------------+
| username |         ln         |
+----------+--------------------+
| JSG99FHE |                  0 |
| PGL08LJI | 0.6931471805599453 |
| IFT66TXU | 1.0986122886681098 |
| XDZ38RDD | 1.3862943611198906 |
| AEB55QTM | 1.6094379124341003 |
| NDQ15VBM |  1.791759469228055 |
| OWY35QYB | 1.9459101490553132 |
| AZG78YIP | 2.0794415416798357 |
| MSD36KVR | 2.1972245773362196 |
| WKW41AIW |  2.302585092994046 |
+----------+--------------------+
```

Função LOG

Retorna o logaritmo de um número.

Se você estiver usando essa função para calcular o logaritmo de base 10, também poderá usar Função DLOG10.

Sintaxe

```
LOG([base, ]argument)
```

Parâmetros

base

(Opcional) A base da função de logaritmo. Esse número deve ser positivo e não pode ser igual a 1. Se esse parâmetro for omitido, o Amazon Redshift calculará o logaritmo de base 10 do argumento.

argument

O argumento da função de logaritmo. Esse número deve ser positivo. Se o valor de argument for 1, a função retornará `0`.

Tipo de retorno

A função LOG retorna um número `DOUBLE PRECISION`.

Exemplos

Para encontrar o logaritmo de base 2 de 100, use o exemplo a seguir.

```
SELECT LOG(2, 100);
+-------------------+
|        log        |
+-------------------+
| 6.643856189774725 |
+-------------------+
```

Para encontrar o logaritmo de base 10 de 100, use o exemplo a seguir. Observe que, se você omitir o parâmetro de base, o Amazon Redshift assumirá uma base de 10.

```
SELECT LOG(100);

+-----+
| log |
+-----+
|   2 |
+-----+
```

Função MOD

Retorna o resto de dois números, também chamado de operação modulo. Para calcular o resultado, o primeiro parâmetro é dividido pelo segundo.

Sintaxe

```
MOD(number1, number2)
```

Argumentos

number1

O primeiro parâmetro de entrada é um número INTEGER, SMALLINT, BIGINT ou DECIMAL. Se um dos parâmetros for do tipo DECIMAL, o outro parâmetro também deverá ser do tipo DECIMAL. Se um dos parâmetros for um INTEGER, o outro parâmetro poderá ser um INTEGER, SMALLINT ou BIGINT. Ambos os parâmetros também podem ser SMALLINT ou BIGINT, mas um parâmetro não poderá ser SMALLINT se o outro for BIGINT.

number2

O segundo parâmetro é um número INTEGER, SMALLINT, BIGINT ou DECIMAL. As mesmas regras de tipo de dados são válidas para number2 e number1.

Tipo de retorno

O tipo de retorno da função MOD é o mesmo tipo numérico que os parâmetros de entrada se ambos os parâmetros de entrada forem do mesmo tipo. Se um dos parâmetro de entrada for INTEGER, porém, o tipo de retorno também será INTEGER. Os tipos de retorno válidos são DECIMAL, INT, SMALLINT e BIGINT.

Observações de uso

Você pode usar % como um operador de módulo.

Exemplos

Para retornar o resto da divisão de um número por outro, use o exemplo a seguir.

```
SELECT MOD(10, 4);

+-----+
| mod |
+-----+
|   2 |
+-----+
```

Para devolver um resultado DECIMAL ao usar a função MOD, use o exemplo a seguir.

```
SELECT MOD(10.5, 4);

+-----+
| mod |
+-----+
| 2.5 |
+-----+
```

Para transmitir um número antes de executar a função MOD, use o exemplo a seguir. Para obter mais informações, consulte Função CAST.

```
SELECT MOD(CAST(16.4 AS INTEGER), 5);

+-----+
| mod |
+-----+
|   1 |
+-----+
```

Para verificar se o primeiro parâmetro é par dividindo-o por 2, use o exemplo a seguir.

```
SELECT mod(5,2) = 0 AS is_even;
```

```
+---------+
| is_even |
+---------+
| false   |
+---------+
```

Para usar % como operador de módulo, use o exemplo a seguir.

```
SELECT 11 % 4 as remainder;

 +-----------+
| remainder |
+-----------+
|         3 |
+-----------+
```

Os exemplos a seguir usa o banco de dados de exemplo de TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para retornar informações das categorias com números ímpares na tabela CATEGORY, use o exemplo a seguir.

```
SELECT catid, catname
FROM category
WHERE MOD(catid,2)=1
ORDER BY 1,2;

+-------+-----------+
| catid |  catname  |
+-------+-----------+
|     1 | MLB       |
|     3 | NFL       |
|     5 | MLS       |
|     7 | Plays     |
|     9 | Pop       |
|    11 | Classical |
+-------+-----------+
```

Função PI

A função PI retorna o valor de pi para 14 casas decimais.

Sintaxe

```
PI()
```

Tipo de retorno

DOUBLE PRECISION

Exemplos

Para retornar o valor de pi, use o exemplo a seguir.

```
SELECT PI();

+-------------------+
|        pi         |
+-------------------+
| 3.141592653589793 |
+-------------------+
```

Função POWER

A função POWER é uma função exponencial que eleva uma expressão numérica para a potência de uma segunda expressão numérica. Por exemplo, 2 elevado à terceira potência é calculado como POWER(2,3), com um resultado de 8.

Sintaxe

```
{POW | POWER}(expression1, expression2)
```

Argumentos

expression1

Expressão numérica a ser elevada. Deve ser um tipo de dados INTEGER, DECIMAL ou FLOAT.

expression2

Potência a elevar a expression1. Deve ser um tipo de dados INTEGER, DECIMAL ou FLOAT.

Tipo de retorno

DOUBLE PRECISION

Exemplos

Os exemplos a seguir usam o banco de dados de exemplo de TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

No exemplo a seguir, a função POWER é usada para prever a situação das vendas de ingressos nos próximos 10 anos com base no número de ingressos vendidos em 2008 (o resultado da subconsulta). A taxa de crescimento é definida como 7% ao ano neste exemplo.

```
SELECT (SELECT SUM(qtysold) FROM sales, date
WHERE sales.dateid=date.dateid
AND year=2008) * POW((1+7::FLOAT/100),10) qty2010;

+-------------------+
|      qty2010      |
+-------------------+
| 679353.7540885945 |
+-------------------+
```

O seguinte exemplo é uma variação do exemplo anterior, com a taxa de crescimento de 7% ao ano, mas o intervalo está definido como meses (120 meses ao longo de 10 anos).

```
SELECT (SELECT SUM(qtysold) FROM sales, date
WHERE sales.dateid=date.dateid
AND year=2008) * POW((1+7::FLOAT/100/12),120) qty2010;

+-----------------+
|     qty2010     |
+-----------------+
| 694034.54678046 |
+-----------------+
```

Função RADIANS

A função RADIANS converte um ângulo em graus em seu equivalente em radianos.

Sintaxe

```
RADIANS(number)
```

Argumento

Número da

O parâmetro de entrada é um número DOUBLE PRECISION.

Tipo de retorno

DOUBLE PRECISION

Exemplos

Para retornar o radiano equivalente de 180 graus, use o exemplo a seguir.

```
SELECT RADIANS(180);

+-------------------+
|      radians      |
+-------------------+
| 3.141592653589793 |
+-------------------+
```

Função RANDOM

A função RANDOM gera um valor aleatório entre 0,0 (inclusive) e 1,0 (exclusivo).

Sintaxe

```
RANDOM()
```

Tipo de retorno

DOUBLE PRECISION

Observações de uso

A função RANDOM pode ser semeada usando o valor de seed SET TO. Quando semeada, cada fatia computacional gera sua própria sequência pseudoaleatória reproduzível. No entanto, os resultados gerais da consulta não são determinísticos em um ambiente MPP distribuído porque você não pode prever qual fatia processará qual linha. O valor de seed afeta o gerador de números aleatórios em cada nó de computação, mas a distribuição das linhas entre as fatias varia.

Exemplos

Para calcular um valor aleatório entre 0 e 99, use o exemplo a seguir. Se o número aleatório é de 0 a 1, essa consulta produz um número aleatório de 0 a 100.

```
SELECT CAST(RANDOM() * 100 AS INT);

+------+
| int4 |
+------+
|   59 |
+------+
```

Este exemplo usa o comando SET para definir um valor de SEED de forma que RANDOM gere uma sequência previsível de números.

Para retornar três números inteiros RANDOM sem definir o valor de SEED, use o exemplo a seguir.

```
SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|    6 |
+------+

SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|   68 |
+------+

SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|   56 |
+------+
```

Para definir o valor de SEED como `.25` e retornar mais três números RANDOM, use o exemplo a seguir.

```
SET SEED TO .25;
SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|   21 |
+------+

SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|   79 |
+------+

SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|   12 |
+------+
```

Para redefinir o valor de SEED como `.25` e verificar se RANDOM retorna os mesmos resultados que as três chamadas anteriores, use o exemplo a seguir.

```
SET SEED TO .25;
SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|   21 |
+------+

SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|   79 |
+------+

SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
```

```
+------+
|   12 |
+------+
```

Os exemplos a seguir usam o banco de dados de exemplo de TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para recuperar uma amostra aleatória uniforme de dez itens da tabela SALES, use o exemplo a seguir.

```
SELECT *
FROM sales
ORDER BY RANDOM()
LIMIT 10;

+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
| salesid | listid | sellerid | buyerid | eventid | dateid | qtysold | pricepaid |
 commission |      saletime       |
+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
|   45422 |  51114 |     5983 |   24482 |    4369 |   2118 |       1 |       195 |
 29.25 | 2008-10-19 05:20:07 |
|   42481 |  47638 |     4573 |    6198 |    6479 |   1987 |       4 |      1140 |
   171 | 2008-06-10 09:39:19 |
|   31494 |  34759 |    18895 |    4719 |    7753 |   2090 |       4 |      1024 |
 153.6 | 2008-09-21 03:44:26 |
|  119388 | 136685 |    21815 |   41905 |    2071 |   1884 |       1 |       359 |
 53.85 | 2008-02-27 10:43:10 |
|  166990 | 225037 |    18529 |    7628 |     746 |   2113 |       1 |      2009 |
 301.35 | 2008-10-14 10:07:44 |
|   11146 |  12096 |    42685 |    6619 |    1876 |   2123 |       1 |        29 |
  4.35 | 2008-10-24 06:23:54 |
|  148537 | 172056 |    15102 |   11787 |    6122 |   1923 |       2 |       480 |
    72 | 2008-04-07 03:58:23 |
|   68945 |  78387 |     7359 |   18323 |    6636 |   1910 |       1 |       457 |
 68.55 | 2008-03-25 08:31:03 |
|   52796 |  59576 |     9909 |   15102 |    7958 |   1951 |       1 |       479 |
 71.85 | 2008-05-05 02:25:08 |
|   90684 | 103522 |    38052 |   21549 |    7384 |   2117 |       1 |       313 |
 46.95 | 2008-10-18 05:43:11 |
+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
```

Para recuperar uma amostra aleatória de dez itens, mas escolher os itens em proporção aos respectivos preços, use o exemplo a seguir. Por exemplo, um item cujo preço é o dobro de outro tem duas vezes mais probabilidade de aparecer nos resultados de consulta.

```
SELECT *
FROM sales
ORDER BY -LOG(RANDOM()) / pricepaid
LIMIT 10;

+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
| salesid | listid | sellerid | buyerid | eventid | dateid | qtysold | pricepaid |
 commission |      saletime       |
+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
|  158340 | 208208 |    17082 |   42018 |    1211 |   2160 |       4 |      6852 |
 1027.8 | 2008-11-30 12:21:43 |
|   53250 |  60069 |    12644 |    7066 |    7942 |   1838 |       4 |      1528 |
 229.2 | 2008-01-12 11:24:56 |
|   22929 |  24938 |    47314 |    6503 |     179 |   2000 |       3 |       741 |
 111.15 | 2008-06-23 08:04:50 |
|  164980 | 221181 |     1949 |   19670 |    1471 |   1906 |       1 |      1330 |
 199.5 | 2008-03-21 07:59:51 |
|  159641 | 211179 |    44897 |   16652 |    7458 |   2128 |       1 |      1019 |
 152.85 | 2008-10-29 02:02:15 |
|   73143 |  83439 |     5716 |    5727 |    7314 |   1903 |       1 |       248 |
  37.2 | 2008-03-18 11:07:42 |
|   84778 |  96749 |    46608 |   32980 |    3883 |   1999 |       2 |       958 |
 143.7 | 2008-06-22 12:13:31 |
|  171096 | 232929 |    43683 |    8536 |    8353 |   1870 |       1 |       929 |
 139.35 | 2008-02-13 01:36:36 |
|   74212 |  84697 |    39809 |   15569 |    5525 |   2105 |       2 |       896 |
 134.4 | 2008-10-06 11:47:50 |
|  158011 | 207556 |    25399 |   16881 |     232 |   2088 |       2 |      2526 |
 378.9 | 2008-09-19 06:00:26 |
+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
```

Função ROUND

A função ROUND arredonda números para o inteiro ou decimal mais próximo.

A função ROUND pode incluir opcionalmente um segundo argumento como um INTEGER para indicar o número de casas decimais para arredondamento, em qualquer direção. Quando você não fornece o segundo argumento, a função arredonda para o número inteiro mais próximo. Quando o segundo argumento inteiro é especificado, a função arredonda para o número mais próximo com inteiro casas decimais de precisão.

Sintaxe

```
ROUND(number [ , integer ] )
```

Argumentos

Número da

Um número ou expressão avaliada como um número. Ele pode ser do tipo DECIMAL, FLOAT8 ou SUPER. O Amazon Redshift pode converter implicitamente outros tipos de dados numéricos.

integer

(Opcional) Um INTEGER que indica o número de casas decimais para arredondamento em ambas as direções. O tipo de dados SUPER não é compatível com esse argumento.

Tipo de retorno

ROUND retorna o mesmo tipo de dados numéricos que a entrada number.

Quando a entrada é do tipo SUPER, a saída mantém o mesmo tipo dinâmico que a entrada enquanto o tipo estático permanece o tipo SUPER. Quando o tipo dinâmico de SUPER não é um número, o Amazon Redshift retorna NULL.

Exemplos

Os exemplos a seguir usam o banco de dados de exemplo de TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para arredondar a comissão paga para determinada transação para o número inteiro mais próximo, use o exemplo a seguir.

```
SELECT commission, ROUND(commission)
FROM sales WHERE salesid=10000;

+------------+-------+
```

```
| commission | round |
+------------+-------+
|      28.05 |    28 |
+------------+-------+
```

Para arredondar a comissão paga para determinada transação para a primeira casa decimal, use o exemplo a seguir.

```
SELECT commission, ROUND(commission, 1)
FROM sales WHERE salesid=10000;

+------------+-------+
| commission | round |
+------------+-------+
|      28.05 |  28.1 |
+------------+-------+
```

Para estender a precisão na direção oposta à do exemplo anterior, use o exemplo a seguir.

```
SELECT commission, ROUND(commission, -1)
FROM sales WHERE salesid=10000;

+------------+-------+
| commission | round |
+------------+-------+
|      28.05 |    30 |
+------------+-------+
```

Função SIN

SIN é uma função trigonométrica que retorna o seno de um número. O valor de retorno está entre `-1` e `1`.

Sintaxe

```
SIN(number)
```

Argumento

Número da

Um número `DOUBLE PRECISION` em radianos.

Tipo de retorno

`DOUBLE PRECISION`

Exemplos

Para retornar o seno de `-PI`, use o exemplo a seguir.

```
SELECT SIN(-PI());

+-------------------------+
|           sin           |
+-------------------------+
| -0.00000000000000012246 |
+-------------------------+
```

Função SIGN

A função SIGN retorna o sinal (positivo ou negativo) de um número. O resultado da função SIGN será 1 se o argumento for positivo, `-1` se o argumento for negativo ou `0` se o argumento for `0`.

Sintaxe

```
SIGN(number)
```

Argumento

Número da

Número ou expressão que avalia para um número. Ele pode ser tipo `DECIMAL`, `FLOAT8` ou `SUPER`. O Amazon Redshift pode converter outros tipos de dados de acordo com as regras de conversão implícitas.

Tipo de retorno

SIGN retorna o mesmo tipo de dados numéricos que o argumento de entrada. Se a entrada for `DECIMAL`, a saída será `DECIMAL(1,0)`.

Quando a entrada é do tipo `SUPER`, a saída mantém o mesmo tipo dinâmico que a entrada enquanto o tipo estático permanece o tipo `SUPER`. Quando o tipo dinâmico de `SUPER` não é um número, o Amazon Redshift retorna um `NULL`.

Exemplos

O exemplo a seguir mostra que a coluna d na tabela t2 tem `DOUBLE PRECISION` como seu tipo, já que a entrada é `DOUBLE PRECISION` e que a coluna n na tabela t2 tem `NUMERIC(1,0)` como saída, visto que a entrada é `NUMERIC`.

```
CREATE TABLE t1(d DOUBLE PRECISION, n NUMERIC(12, 2));
INSERT INTO t1 VALUES (4.25, 4.25), (-4.25, -4.25);
CREATE TABLE t2 AS SELECT SIGN(d) AS d, SIGN(n) AS n FROM t1;
SELECT table_name, column_name, data_type FROM SVV_REDSHIFT_COLUMNS WHERE
 table_name='t1' OR table_name='t2';

+------------+-------------+----------------------+
| table_name | column_name |       data_type      |
+------------+-------------+----------------------+
| t1         | d           | double precision     |
| t1         | n           | numeric(12,2)        |
| t2         | d           | double precision     |
| t2         | n           | numeric(1,0)         |
| t1         | col1        | character varying(20) |
+------------+-------------+----------------------+
```

Os exemplos a seguir usa o banco de dados de exemplo de TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para determinar o sinal da comissão paga por determinada transação na tabela SALES, use o exemplo a seguir.

```
SELECT commission, SIGN(commission)
FROM sales WHERE salesid=10000;

+------------+------+
| commission | sign |
+------------+------+
|      28.05 |    1 |
+------------+------+
```

Função SQRT

A função SQRT retorna a raiz quadrada de um valor `NUMERIC`. A raiz quadrada é um número multiplicado por si mesmo para obter o valor fornecido.

Sintaxe

```
SQRT(expression)
```

Argumento

expressão

A expressão deve ter um tipo de dados INTEGER, DECIMAL ou FLOAT ou que seja convertido implicitamente nesses tipos de dados. A expressão pode incluir funções.

Tipo de retorno

DOUBLE PRECISION

Exemplos

Para retornar a raiz quadrada de 16, use o exemplo a seguir.

```
SELECT SQRT(16);

+------+
| sqrt |
+------+
|    4 |
+------+
```

Para retornar a raiz quadrada da string 16 usando uma conversão de tipo implícita, use o exemplo a seguir.

```
SELECT SQRT('16');

+------+
| sqrt |
+------+
|    4 |
+------+
```

Para retornar a raiz quadrada de 16,4 depois de usar a função ROUND, use o exemplo a seguir.

```
SELECT SQRT(ROUND(16.4));
```

```
+------+
| sqrt |
+------+
|    4 |
+------+
```

Para retornar o comprimento do raio quando dada a área de um círculo, use o exemplo a seguir. Ele calcula o raio em polegadas, por exemplo, quando dada a área em polegadas quadradas. A área na amostra é 20.

```
SELECT SQRT(20/PI()) AS radius;

+--------------------+
|      radius        |
+--------------------+
| 2.5231325220201604 |
+--------------------+
```

Os exemplos a seguir usam o banco de dados de exemplo de TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para retornar a raiz quadrada de valores de COMMISSION da tabela SALES, use o exemplo a seguir. A coluna COMMISSION é uma coluna `DECIMAL`. Este exemplo mostra como você pode usar a função em uma consulta com uma lógica condicional mais complexa.

```
SELECT SQRT(commission)
FROM sales WHERE salesid < 10 ORDER BY salesid;

+--------------------+
|        sqrt        |
+--------------------+
| 10.449880382090505 |
| 3.3763886032268267 |
|  7.245688373094719 |
|  5.123475382979799 |
|  4.806245936279167 |
|  7.687652437513028 |
| 10.871982339941507 |
| 5.4359911699707535 |
|   9.41541289588513 |
+--------------------+
```

Para retornar a raiz quadrada arredondada para o mesmo conjunto de valores de COMMISSION, use o exemplo a seguir.

```
SELECT ROUND(SQRT(commission))
FROM sales WHERE salesid < 10 ORDER BY salesid;

+-------+
| round |
+-------+
|    10 |
|     3 |
|     7 |
|     5 |
|     5 |
|     8 |
|    11 |
|     5 |
|     9 |
+-------+
```

Função TAN

TAN é uma função trigonométrica que retorna a tangente de um número. O argumento de entrada é um número (em radianos).

Sintaxe

```
TAN(number)
```

Argumento

Número da

Um número DOUBLE PRECISION.

Tipo de retorno

DOUBLE PRECISION

Exemplos

Para retornar a tangente de zero, use o exemplo a seguir.

```
SELECT TAN(0);

+-----+
| tan |
+-----+
|   0 |
+-----+
```

Função TRUNC

A função TRUNC trunca números para o inteiro ou decimal anterior.

A função TRUNC pode incluir opcionalmente um segundo argumento como um `INTEGER` para indicar o número de casas decimais para arredondamento, em qualquer direção. Quando você não fornece o segundo argumento, a função arredonda para o número inteiro mais próximo. Quando o segundo argumento inteiro é especificado, a função arredonda para o número mais próximo com inteiro casas decimais de precisão.

Essa função também pode truncar um `TIMESTAMP` e retornar um `DATE`. Para obter mais informações, consulte Função TRUNC.

Sintaxe

```
TRUNC(number [ , integer ])
```

Argumentos

Número da

Um número ou uma expressão avaliada como um número. Ele pode ser do tipo `DECIMAL`, `FLOAT8` ou `SUPER`. O Amazon Redshift pode converter outros tipos de dados de acordo com as regras de conversão implícitas.

integer

(Opcional) Um `INTEGER` que indica o número de casas decimais de precisão, em um dos sentidos. Se nenhum inteiro for fornecido, o número será truncado como um número inteiro; se um inteiro for especificado, o número será truncado para a casa decimal especificada. Isso não é compatível com o tipo de dados `SUPER`.

Tipo de retorno

TRUNC retorna o mesmo tipo de dados que a entrada número.

Quando a entrada é do tipo SUPER, a saída mantém o mesmo tipo dinâmico que a entrada enquanto o tipo estático permanece o tipo SUPER. Quando o tipo dinâmico de SUPER não é um número, o Amazon Redshift retorna NULL.

Exemplos

Alguns dos exemplos a seguir usam o banco de dados de amostra de TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para truncar a comissão paga para determinada transação de vendas, use o exemplo a seguir.

```
SELECT commission, TRUNC(commission)
FROM sales WHERE salesid=784;

+------------+-------+
| commission | trunc |
+------------+-------+
|     111.15 |   111 |
+------------+-------+
```

Para truncar o mesmo valor de comissão para a primeira casa decimal, use o exemplo a seguir.

```
SELECT commission, TRUNC(commission,1)
FROM sales WHERE salesid=784;

+------------+-------+
| commission | trunc |
+------------+-------+
|     111.15 | 111.1 |
+------------+-------+
```

Para truncar a comissão com um valor negativo para o segundo argumento, use o exemplo a seguir. Observe que 111.15 é arredondado para 110.

```
SELECT commission, TRUNC(commission,-1)
FROM sales WHERE salesid=784;

+------------+-------+
```

```
| commission | trunc |
+------------+-------+
|     111.15 |   110 |
+------------+-------+
```

Funções de objetos

Veja a seguir as funções de objeto do SQL compatíveis com o Amazon Redshift para criar e operar em objetos do tipo SUPER:

Tópicos

- Função GET_NUMBER_ATTRIBUTES
- Função LOWER_ATTRIBUTE_NAMES
- Função OBJECT
- Função OBJECT_TRANSFORM
- Função UPPER_ATTRIBUTE_NAMES

Função GET_NUMBER_ATTRIBUTES

Exibe a contagem de pares de chave-valor existentes no nível raiz de um objeto de dicionário.

Sintaxe

```
GET_NUMBER_ATTRIBUTES( super_expression )
```

Argumentos

super_expression

Uma expressão SUPER em forma de dicionário.

Tipo de retorno

A função GET_NUMBER_ATTRIBUTES exibe um tipo INT.

Observação

Essa função conta somente atributos diretos e não inclui pares em dicionários aninhados.

Exemplo

O exemplo a seguir mostra a função GET_NUMBER_ATTRIBUTES.

```
SELECT GET_NUMBER_ATTRIBUTES(JSON_PARSE('{"a": 1, "b": 2, "c": 3}'));
 get_number_attributes
-----------------------
                     3
(1 row)
```

A função GET_NUMBER_ATTRIBUTES opera somente no primeiro nível do dicionário.

```
SELECT GET_NUMBER_ATTRIBUTES(JSON_PARSE('{"a": 1, "b": {"c": 3}}'));
 get_number_attributes
-----------------------
                     2
(1 row)
```

Função LOWER_ATTRIBUTE_NAMES

Converte todos os nomes de atributos aplicáveis em um valor SUPER em minúsculas, usando a mesma rotina de conversão de maiúsculas e minúsculas de Função LOWER. LOWER_ATTRIBUTE_NAMES aceita caracteres UTF-8 multibyte, até o máximo de quatro bytes por caractere.

Para converter nomes de atributos SUPER em maiúsculas, use Função UPPER_ATTRIBUTE_NAMES.

Sintaxe

```
LOWER_ATTRIBUTE_NAMES( super_expression )
```

Argumentos

super_expression

Uma expressão SUPER.

Tipo de retorno

SUPER

Observações de uso

No Amazon Redshift, os identificadores de coluna tradicionalmente não diferenciam maiúsculas de minúsculas e são convertidos em minúsculas. Se você ingerir dados de formatos de dados que diferenciam maiúsculas e minúsculas, como JSON, os dados poderão conter nomes de atributos com maiúsculas e minúsculas.

Considere o exemplo a seguir.

```
CREATE TABLE t1 (s) AS SELECT JSON_PARSE('{"AttributeName": "Value"}');


SELECT s.AttributeName FROM t1;

attributename
-------------
NULL


SELECT s."AttributeName" FROM t1;

attributename
-------------
NULL
```

O Amazon Redshift exibe NULL para as duas consultas. Para consultar `AttributeName`, use LOWER_ATTRIBUTE_NAMES para converter os nomes dos atributos dos dados em minúsculas. Considere o exemplo a seguir.

```
CREATE TABLE t2 (s) AS SELECT LOWER_ATTRIBUTE_NAMES(s) FROM t1;


SELECT s.attributename FROM t2;

attributename
-------------
"Value"


SELECT s.AttributeName FROM t2;

attributename
```

```
-------------
"Value"


SELECT s."attributename" FROM t2;

attributename
-------------
"Value"


SELECT s."AttributeName" FROM t2;

attributename
-------------
"Value"
```

Uma opção relacionada para trabalhar com nomes de atributos de objetos com maiúsculas e minúsculas é a opção de configuração `enable_case_sensitive_super_attribute`, que permite ao Amazon Redshift reconhecer maiúsculas e minúsculas em nomes de atributos SUPER. Essa pode ser uma solução alternativa ao uso de LOWER_ATTRIBUTE_NAMES. Para ter mais informações sobre `enable_case_sensitive_super_attribute`, acesse enable_case_sensitive_super_attribute.

Exemplos

Converter nomes de atributos SUPER em minúsculas

O exemplo a seguir usa LOWER_ATTRIBUTE_NAMES para converter os nomes dos atributos de todos os valores SUPER em uma tabela.

```
-- Create a table and insert several SUPER values.
CREATE TABLE t (i INT, s SUPER);

INSERT INTO t VALUES
  (1, NULL),
  (2, 'A'::SUPER),
  (3, JSON_PARSE('{"AttributeName": "B"}')),
  (4, JSON_PARSE(
     '[{"Subobject": {"C": "C"},
       "Subarray": [{"D": "D"}, "E"]
      }]'));
```

```
-- Convert all attribute names to lowercase.
UPDATE t SET s = LOWER_ATTRIBUTE_NAMES(s);

SELECT i, s FROM t ORDER BY i;

 i |                         s
---+---------------------------------------------------
 1 | NULL
 2 | "A"
 3 | {"attributename":"B"}
 4 | [{"subobject":{"c":"C"},"subarray":[{"d":"D"}, "E"]}]
```

Observe como LOWER_ATTRIBUTE_NAMES funciona.

- Valores NULL e valores SUPER escalares, como ''A'', permanecem inalterados.
- Em um objeto SUPER, todos os nomes de atributos são alterados para minúsculas, mas valores de atributos como ''B'' permanecem inalterados.
- LOWER_ATTRIBUTE_NAMES aplica-se recursivamente a qualquer objeto SUPER aninhado dentro de uma matriz SUPER ou dentro de outro objeto.

Usar LOWER_ATTRIBUTE_NAMES em um objeto SUPER com nomes de atributos duplicados

Se um objeto SUPER contiver atributos cujos nomes sejam diferentes apenas em termos de maiúsculas e minúsculas, LOWER_ATTRIBUTE_NAMES gerará um erro. Considere o exemplo a seguir.

```
SELECT LOWER_ATTRIBUTE_NAMES(JSON_PARSE('{"A": "A", "a": "a"}'));

error:   Invalid input
code:    8001
context: SUPER value has duplicate attributes after case conversion.
```

Função OBJECT

Cria um objeto do tipo de dados SUPER.

Sintaxe

```
OBJECT ( [ key1, value1 ], [ key2, value2 ...] )
```

Argumentos

key1, key2

Expressões avaliadas em strings do tipo VARCHAR.

value1, value2

Expressões de qualquer tipo de dado do Amazon Redshift, exceto os tipos de data e hora, já que o Amazon Redshift não transmite os tipos de data e hora para o tipo de dados SUPER. Para obter mais informações sobre tipos de data e hora, consulte Tipos de datetime.

As expressões `value` em um objeto não precisam ser do mesmo tipo de dados.

Tipo de retorno

SUPER

Exemplo

```
-- Creates an empty object.
select object();

object
--------
{}
(1 row)

-- Creates objects with different keys and values.
select object('a', 1, 'b', true, 'c', 3.14);

object
---------------------------
{"a":1,"b":true,"c":3.14}
(1 row)

select object('a', object('aa', 1), 'b', array(2,3), 'c', json_parse('{}'));

object
---------------------------------
{"a":{"aa":1},"b":[2,3],"c":{}}
(1 row)
```

```
-- Creates objects using columns from a table.
create table bar (k varchar, v super);
insert into bar values ('k1', json_parse('[1]')), ('k2', json_parse('{}'));
select object(k, v) from bar;

object
------------
{"k1":[1]}
{"k2":{}}
(2 rows)

-- Errors out because DATE type values can't be converted to SUPER type.
select object('k', '2008-12-31'::date);

ERROR:  OBJECT could not convert type date to super
```

Função OBJECT_TRANSFORM

Transforma um objeto SUPER.

Sintaxe

```
OBJECT_TRANSFORM(
  input
  [KEEP path1, ...]
  [SET
    path1, value1,
    ...,  ...
  ]
)
```

Argumentos

input

Uma expressão resolvida em um objeto do tipo SUPER.

KEEP

Todos os valores path especificados nessa cláusula são mantidos e transportados para o objeto de saída.

A cláusula é opcional.

path1, path2...

Literais de string constantes, no formato de componentes de caminho com aspas duplas delimitados por pontos finais. Por exemplo, `'"a"."b"."c"'` não é um valor de caminho válido. Isso se aplica ao parâmetro path nas cláusulas KEEP e SET.

SET

Pares path e value para modificar um caminho existente ou adicionar um novo caminho, além de definir o valor desse caminho no objeto de saída.

A cláusula é opcional.

value1, value2...

Expressões resolvidas como valores do tipo SUPER. Os tipos numérico, de texto e booliano podem ser resolvidos como SUPER.

Tipo de retorno

`SUPER`

Observações de uso

OBJECT_TRANSFORM retorna um objeto do tipo SUPER contendo os valores do caminho de entrada que foram especificados em KEEP e os pares de caminho e valor que foram especificados em SET.

Se KEEP e SET estiverem vazios, OBJECT_TRANSFORM vai retornar entrada.

Se a entrada não for um objeto do tipo SUPER, OBJECT_TRANSFORM retornará a entrada, independentemente de eventuais valores KEEP ou SET.

Exemplo

O exemplo a seguir transforma um objeto SUPER em outro objeto SUPER.

```
CREATE TABLE employees (
    col_person SUPER
);

INSERT INTO employees
VALUES
    (
```

```
        json_parse('
            {
                "name": {
                    "first": "John",
                    "last": "Doe"
                },
                "age": 25,
                "ssn": "111-22-3333",
                "company": "Company Inc.",
                "country": "U.S."
            }
        ')
    ),
    (
        json_parse('
            {
                "name": {
                    "first": "Jane",
                    "last": "Appleseed"
                },
                "age": 34,
                "ssn": "444-55-7777",
                "company": "Organization Org.",
                "country": "Ukraine"
            }
        ')
    )
;

SELECT
    OBJECT_TRANSFORM(
        col_person
        KEEP
            '"name"."first"',
            '"age"',
            '"company"',
            '"country"'
        SET
            '"name"."first"', UPPER(col_person.name.first::TEXT),
            '"age"', col_person.age + 5,
            '"company"', 'Amazon'
    ) AS col_person_transformed
FROM employees;
```

```
--This result is formatted for ease of reading.
                  col_person_transformed
-------------------------------------------------------------
{
    "name": {
        "first": "JOHN"
    },
    "age": 30,
    "company": "Amazon",
    "country": "U.S."
}
{
    "name": {
        "first": "JANE"
    },
    "age": 39,
    "company": "Amazon",
    "country": "Ukraine"
}
```

Função UPPER_ATTRIBUTE_NAMES

Converte todos os nomes de atributos aplicáveis em um valor SUPER em maiúsculas, usando a mesma rotina de conversão de maiúsculas e minúsculas de Função UPPER. UPPER_ATTRIBUTE_NAMES é compatível com caracteres UTF-8 multibyte, até o máximo de quatro bytes por caractere.

Para converter nomes de atributos SUPER em minúsculas, use Função LOWER_ATTRIBUTE_NAMES.

Sintaxe

```
UPPER_ATTRIBUTE_NAMES( super_expression )
```

Argumentos

super_expression

Uma expressão SUPER.

Tipo de retorno

SUPER

Exemplos

Converter nomes de atributos SUPER em maiúsculas

O exemplo a seguir usa UPPER_ATTRIBUTE_NAMES para converter os nomes dos atributos de todos os valores SUPER em uma tabela.

```
-- Create a table and insert several SUPER values.
CREATE TABLE t (i INT, s SUPER);

INSERT INTO t VALUES
  (1, NULL),
  (2, 'a'::SUPER),
  (3, JSON_PARSE('{"AttributeName": "b"}')),
  (4, JSON_PARSE(
     '[{"Subobject": {"c": "c"},
        "Subarray": [{"d": "d"}, "e"]
      }]'));

-- Convert all attribute names to uppercase.
UPDATE t SET s = UPPER_ATTRIBUTE_NAMES(s);

SELECT i, s FROM t ORDER BY i;

 i |                         s
---+---------------------------------------------------
 1 | NULL
 2 | "a"
 3 | {"ATTRIBUTENAME":"B"}
 4 | [{"SUBOBJECT":{"C":"c"},"SUBARRAY":[{"D":"d"}, "e"]}]
```

Observe como UPPER_ATTRIBUTE_NAMES funciona.

- Valores NULL e valores SUPER escalares, como "a", permanecem inalterados.
- Em um objeto SUPER, todos os nomes de atributos são alterados para maiúsculas, mas valores de atributos como "b" permanecem inalterados.
- UPPER_ATTRIBUTE_NAMES aplica-se recursivamente a qualquer objeto SUPER aninhado dentro de uma matriz SUPER ou dentro de outro objeto.

Usar UPPER_ATTRIBUTE_NAMES em um objeto SUPER com nomes de atributos duplicados

Se um objeto SUPER contiver atributos cujos nomes sejam diferentes apenas em termos de maiúsculas e minúsculas, UPPER_ATTRIBUTE_NAMES gerará um erro. Considere o exemplo a seguir.

```
SELECT UPPER_ATTRIBUTE_NAMES(JSON_PARSE('{"A": "A", "a": "a"}'));

error:   Invalid input
code:    8001
context: SUPER value has duplicate attributes after case conversion.
```

Funções espaciais

Os relacionamentos entre objetos geométricos são baseados no Dimensionally Extended nine-Intersection Model (DE-9IM). O modelo define predicados como equals, contains e covers. Para obter mais informações sobre a definição de relacionamentos espaciais, consulte DE-9IM na Wikipedia.

Para obter mais informações sobre como usar dados espaciais com o Amazon Redshift, consulte Consultar dados espaciais no Amazon Redshift.

O Amazon Redshift fornece funções espaciais que funcionam com os tipos de dados `GEOMETRY` e `GEOGRAPHY`. A seguir estão listadas as funções compatíveis com o tipo de dados `GEOGRAPHY`:

- ST_Area
- ST_AsEWKT
- ST_AsGeoJSON
- ST_AsHexEWKB
- ST_AsHexWKB
- ST_AsText
- ST_Distance
- ST_GeogFromText
- St_GeogFromWKB
- ST_Length
- ST_NPoints

- ST_Perimeter

A seguir está listado o conjunto completo de funções espaciais compatíveis com o Amazon Redshift.

Tópicos

- AddBBox
- DropBBox
- GeometryType
- H3_Boundary
- H3_Center
- H3_FromLongLat
- H3_FromPoint
- H3_IsValid
- H3_Polyfill
- H3_Resolution
- H3_ToChildren
- H3_ToParent
- ST_AddPoint
- ST_Angle
- ST_Area
- ST_AsBinary
- ST_AsEWKB
- ST_AsEWKT
- ST_AsGeoJSON
- ST_AsHexWKB
- ST_AsHexEWKB
- ST_AsText
- ST_Azimuth
- ST_Boundary
- ST_Buffer

- ST_Centroid
- ST_Collect
- ST_Contains
- ST_ContainsProperly
- ST_ConvexHull
- ST_CoveredBy
- ST_Covers
- ST_Crosses
- ST_Dimension
- ST_Disjoint
- ST_Distance
- ST_DistanceSphere
- ST_DWithin
- ST_EndPoint
- ST_Envelope
- ST_Equals
- ST_ExteriorRing
- ST_Force2D
- ST_Force3D
- ST_Force3DM
- ST_Force3DZ
- ST_Force4D
- ST_GeoHash
- ST_GeogFromText
- St_GeogFromWKB
- ST_GeometryN
- ST_GeometryType
- ST_GeomFromEWKB
- ST_GeomFromEWKT
- ST_GeomFromGeoHash

- ST_GeomFromGeoJSON
- ST_GeomFromGeoSquare
- ST_GeomFromText
- ST_GeomFromWKB
- ST_GeoSquare
- ST_InteriorRingN
- ST_Intersects
- ST_Intersection
- ST_IsPolygonCCW
- ST_IsPolygonCW
- ST_IsClosed
- ST_IsCollection
- ST_IsEmpty
- ST_IsRing
- ST_IsSimple
- ST_IsValid
- ST_Length
- ST_LengthSphere
- ST_Length2D
- ST_LineFromMultiPoint
- ST_LineInterpolatePoint
- ST_M
- ST_MakeEnvelope
- ST_MakeLine
- ST_MakePoint
- ST_MakePolygon
- ST_MemSize
- ST_MMax
- ST_MMin

- ST_Multi
- ST_NDims
- ST_NPoints
- ST_NRings
- ST_NumGeometries
- ST_NumInteriorRings
- ST_NumPoints
- ST_Perimeter
- ST_Perimeter2D
- ST_Point
- ST_PointN
- ST_Points
- ST_Polygon
- ST_RemovePoint
- ST_Reverse
- ST_SetPoint
- ST_SetSRID
- ST_Simplify
- ST_SRID
- ST_StartPoint
- ST_Touches
- ST_Transform
- ST_Union
- ST_Within
- ST_X
- ST_XMax
- ST_XMin
- ST_Y
- ST_YMax
- ST_YMin

- ST_Z
- ST_ZMax
- ST_ZMin
- SupportsBBox

AddBBox

AddBBox retorna uma cópia da geometria de entrada que suporta codificação com uma caixa delimitadora pré-computada. Para obter mais informações sobre o suporte para caixas delimitadoras, consulte Caixa delimitadora.

Sintaxe

```
AddBBox(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

GEOMETRY

Nulo será retornado se geom for nulo.

Exemplos

O SQL a seguir retorna uma cópia de uma geometria de polígono de entrada que suporta ser codificado com uma caixa delimitadora.

```
SELECT ST_AsText(AddBBox(ST_GeomFromText('POLYGON((0 0,1 0,0 1,0 0))')));
```

```
 st_astext
----------
```

```
 POLYGON((0 0,1 0,0 1,0 0))
```

DropBBox

DropBBox retorna uma cópia da geometria de entrada que não suporta codificação com uma caixa delimitadora pré-computada. Para obter mais informações sobre o suporte para caixas delimitadoras, consulte Caixa delimitadora.

Sintaxe

```
DropBBox(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

GEOMETRY

Nulo será retornado se geom for nulo.

Exemplos

O SQL a seguir retorna uma cópia de uma geometria de polígono de entrada que não suporta ser codificado com uma caixa delimitadora.

```
SELECT ST_AsText(DropBBox(ST_GeomFromText('POLYGON((0 0,1 0,0 1,0 0))')));
```

```
 st_astext
----------
 POLYGON((0 0,1 0,0 1,0 0))
```

GeometryType

GeometryType retorna o subtipo de uma geometria de entrada como uma string.

Sintaxe

```
GeometryType(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

VARCHAR que representa o subtipo de geom.

Nulo será retornado se geom for nulo.

Os valores retornados são os seguintes.

Valor de string retornado para geometrias 2D, 3DZ, 4D	Valor de string retornado para geometrias 3DM	Subtipo de geometria
POINT	POINTM	Retornado se geom for um subtipo POINT
LINESTRING	LINESTRINGM	Retornado se geom for um subtipo LINESTRING
POLYGON	POLYGONM	Retornado se geom for um subtipo POLYGON
MULTIPOINT	MULTIPOINTM	Retornado se geom for um subtipo MULTIPOINT
MULTILINESTRING	MULTILINESTRINGM	Retornado se geom for um subtipo MULTILINESTRING
MULTIPOLYGON	MULTIPOLYGONM	Retornado se geom for um subtipo MULTIPOLYGON

Valor de string retornado para geometrias 2D, 3DZ, 4D	Valor de string retornado para geometrias 3DM	Subtipo de geometria
GEOMETRYCOLLECTION	GEOMETRYCOLLECTIONM	Retornado se geom for um subtipo GEOMETRYCOLLECTION

Exemplos

O SQL a seguir converte a representação de um texto bem-conhecido (WKT - well-known text) de um polígono e retorna o subtipo GEOMETRY como uma string.

```
SELECT GeometryType(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'));
```

```
geometrytype
-------------
 POLYGON
```

H3_Boundary

H3_Boundary retorna o limite de um ID de célula H3 de um índice de entrada. Para obter informações sobre a indexação H3, consulte H3.

Sintaxe

```
H3_Boundary(index)
```

Argumentos

Índice do

Um valor de tipo de dados BIGINT ou VARCHAR que representa o índice de uma célula H3. Ou uma expressão que seja avaliada como um desses tipos de dados.

Tipo de retorno

POLYGON: representa o polígono com identificador do sistema de referência espacial (SRID) de 0.

Se o índice for inválido, será retornado um erro.

Exemplos

O comando SQL a seguir insere um VARCHAR que representa o índice de uma célula H3 e retorna um POLYGON com SRID 0 que representa o limite da célula H3 de entrada. A saída de H3_Boundary é inserida em ST_AwEWKT para exibição em uma representação de texto bem conhecido estendido (EWKT).

```
SELECT ST_AsEWKT(H3_Boundary('8025fffffffffff'));
```

```
 st_asewkt
-----------------------------------------------------------------------------------------------------------
 POLYGON((87.7729641223953 52.6542030078627,86.7082098104731
 40.3127667561966,98.2285042557705 33.6210697806835,110.694610548823
 37.163896485796,116.212895637138 47.3094513028131,106.40349563788
 56.210610737585,87.7729641223953 52.6542030078627))
```

O comando SQL a seguir insere um BIGINT que representa o índice de uma célula H3 e retorna um POLYGON com SRID 0 que representa o limite da célula H3 de entrada. A saída de H3_Boundary é inserida em ST_AwEWKT para exibição em uma representação de texto bem conhecido estendido (EWKT).

```
SELECT ST_AsEWKT(H3_Boundary(577129255373111295));
```

```
 st_asewkt
-----------------------------------------------------------------------------------------------------------
POLYGON((87.7729641223953 52.6542030078627,86.7082098104731
 40.3127667561966,98.2285042557705 33.6210697806835,110.694610548823
 37.163896485796,116.212895637138 47.3094513028131,106.40349563788
 56.210610737585,87.7729641223953 52.6542030078627))
```

H3_Center

H3_Center retorna o centroide de um ID de célula H3 de um índice de entrada. Para obter informações sobre a indexação H3, consulte H3.

Sintaxe

```
H3_Center(index)
```

Argumentos

Índice do

Um valor de tipo de dados BIGINT ou VARCHAR que representa o índice de uma célula H3. Ou uma expressão que seja avaliada como um desses tipos de dados.

Tipo de retorno

POINT: representa o centroide da célula H3 com identificador do sistema de referência espacial (SRID) de 0.

Se o índice for inválido, será retornado um erro.

Exemplos

O comando SQL a seguir insere um VARCHAR que representa o índice de uma célula H3 e retorna um POINT com SRID de 0 que representa o centroide da célula H3 de entrada.

```
SELECT H3_Center('8025fffffffffff');
```

```
 h3_center
--------------------------------------------
 010100000070707A550B605940AEE9D70B327E4640
```

O comando SQL a seguir insere um BIGINT que representa o índice de uma célula H3 e retorna um POINT com SRID 0 que representa o centroide da célula H3 de entrada.

```
SELECT H3_Center(577129255373111295);
```

```
 h3_center
--------------------------------------------
```

```
 010100000070707A550B605940AEE9D70B327E4640
```

O comando SQL a seguir insere um VARCHAR que representa o índice de uma célula H3 e retorna um POINT com SRID 0 que representa o centroide da célula H3 de entrada. A saída de H3_Center é inserida em ST_AwEWKT para exibição em uma representação de texto bem conhecido estendido (EWKT).

```
SELECT ST_AsEWKT(H3_Center('8075fffffffffff'));
```

```
 st_asewkt
-----------------------------------------
POINT(-5.24539029677733 2.30088211162675)
```

H3_FromLongLat

H3_FromLongLat retorna o ID da célula H3 correspondente de uma longitude, uma latitude e uma resolução de entrada. Para obter informações sobre a indexação H3, consulte H3.

Sintaxe

```
H3_FromLongLat(longitude, latitude, resolution)
```

Argumentos

longitude

Um valor de tipo de dados DOUBLE PRECISION ou uma expressão que é avaliada como um tipo DOUBLE PRECISION.

latitude

Um valor de tipo de dados DOUBLE PRECISION ou uma expressão que é avaliada como um tipo DOUBLE PRECISION.

resolução

Um valor do tipo de dados INTEGER ou uma expressão avaliada como um tipo INTEGER. O valor representa a resolução do sistema de grade H3. O valor deve ser um número inteiro entre 0 e 15, inclusive. Com 0 sendo o mais aproximado e 15 sendo o mais preciso.

Tipo de retorno

`BIGINT`: representa o ID da célula H3.

Se resolução estiver fora dos limites, um erro será retornado.

Exemplos

O SQL a seguir retorna o ID da célula H3 de longitude `0`, latitude `0` e resolução `10`.

```
SELECT H3_FromLongLat(0, 0, 10);
```

```
 h3_fromlonglat
-------------------
 623560421467684863
```

H3_FromPoint

H3_FromPoint exibe o ID da célula H3 correspondente de uma resolução e de um ponto de geometria de entrada. Para obter informações sobre a indexação H3, consulte H3.

Sintaxe

```
H3_FromPoint(geom, resolution)
```

Argumentos

geom

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`. A geom deve ser um `POINT`.

resolução

Um valor do tipo de dados `INTEGER` ou uma expressão avaliada como um tipo `INTEGER`. O valor representa a resolução do sistema de grade H3. O valor deve ser um número inteiro entre 0 e 15, inclusive. Com `0` sendo o mais aproximado e 15 sendo o mais preciso.

Tipo de retorno

`BIGINT`: representa o ID da célula H3.

Se geom não for uma POINT, será retornado um erro.

Se resolução estiver fora dos limites, um erro será retornado.

Se geom estiver vazio, NULL será retornado.

Exemplos

O SQL a seguir retorna o ID da célula H3 do ponto 0,0 e a resolução 10.

```
SELECT H3_FromPoint(ST_GeomFromText('POINT(0 0)'), 10);
```

```
 h3_frompoint
--------------------
 623560421467684863
```

H3_IsValid

H3_IsValid será true se a entrada representar um ID de célula H3, caso contrário, será false. Para obter informações sobre a indexação H3, consulte H3.

Sintaxe

```
H3_IsValid(index)
```

Argumentos

Índice do

Um valor de tipo de dados BIGINT ou VARCHAR ou uma expressão que é avaliada como um desses tipos de dados.

Tipo de retorno

BOOLEAN: true se a entrada representar um ID de célula H3 válido, caso contrário, false.

Se índice for NULL, então, será exibido NULL.

Exemplos

O SQL a seguir insere um VARCHAR que representa um ID de célula H3 e exibe true.

```
SELECT H3_IsValid('8025fffffffffff');
```

```
 h3_isvalid
------------
 true
```

O SQL a seguir insere um BIGINT que representa um ID de célula H3 e exibe true.

```
SELECT H3_IsValid(577129255373111295);
```

```
 h3_isvalid
------------
 true
```

O SQL a seguir insere um ID de célula H3 inválido e exibe false.

```
SELECT H3_IsValid('');
```

```
 h3_isvalid
------------
 false
```

H3_Polyfill

H3_Polyfill retorna os IDs de célula H3 correspondentes aos hexágonos e pentágonos contidos no polígono de entrada da resolução indicada. Para obter informações sobre a indexação H3, consulte H3.

Sintaxe

```
H3_Polyfill(geom, resolution)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY. A geom deve ser um POLYGON.

resolução

Um valor do tipo de dados INTEGER ou uma expressão avaliada como um tipo INTEGER. O valor representa a resolução do sistema de grade H3. O valor deve ser um número inteiro entre 0 e 15, inclusive. Com 0 sendo o mais aproximado e 15 sendo o mais preciso.

Tipo de retorno

SUPER: representa uma lista de IDs de células H3.

Se geom não for uma POLYGON, será retornado um erro.

Se resolução estiver fora dos limites, um erro será retornado.

Se geom estiver vazio, NULL será retornado.

Exemplos

O SQL a seguir retorna uma matriz do tipo de dados SUPER dos IDs de célula H3 de um polígono e uma resolução 4.

```
SELECT H3_Polyfill(ST_GeomFromText('POLYGON((0 0, 0 1, 1 1, 1 0, 0 0))'), 4);
```

```
 h3_polyfill
-------------------------------------------------------------------------------------------------
 [596538848238895103,596538805289222143,596538856828829695,596538813879156735,5965379205259591
```

H3_Resolution

H3_Resolution exibe a resolução de um ID de célula H3 de um índice de entrada. A resolução é um número inteiro de 0 (mais baixa) a 15 (mais fina). Para obter informações sobre a indexação H3, consulte H3.

Sintaxe

```
H3_Resolution(index)
```

Argumentos

Índice do

Um valor de tipo de dados BIGINT ou VARCHAR que representa o índice de uma célula H3, ou uma expressão que determina o valor de um desses tipos de dados.

Tipo de retorno

INTEGER: representa a resolução do ID da célula H3 de entrada.

Se índice for NULL, então, será exibido NULL.

Se o índice for inválido, será retornado um erro.

Exemplos

O comando SQL a seguir insere um VARCHAR que representa o índice de uma célula H3 e exibe um INTEGER que representa a resolução da célula H3 de entrada.

```
SELECT H3_Resolution('8025fffffffffff');
```

```
 h3_resolution
---------------
 0
```

O comando SQL a seguir insere um BIGINT que representa o índice de uma célula H3 e exibe um INTEGER que representa a resolução da célula H3 de entrada.

```
SELECT H3_Resolution(614553222213795839);
```

```
 h3_resolution
---------------
 8
```

H3_ToChildren

H3_toChildren exibe uma lista de IDs de células H3 filhas em uma resolução especificada para determinado índice H3. Para obter informações sobre a indexação H3, consulte H3.

Sintaxe

```
H3_ToChildren(index, resolution)
```

Argumentos

Índice do

Um valor de tipo de dados BIGINT ou VARCHAR que representa o índice de uma célula H3, ou uma expressão que determina o valor de um desses tipos de dados.

resolução

Um valor do tipo de dados INTEGER ou uma expressão avaliada como um tipo INTEGER. O valor representa a resolução dos IDs de células filhas. O valor deve ser um número inteiro entre a resolução do índice de entrada e 15, inclusive.

Tipo de retorno

SUPER: representa uma lista de IDs de células H3.

Se o índice ou a resolução forem NULL, então NULL será exibido.

Se o índice for inválido, será retornado um erro.

Se a resolução não estiver entre a resolução do índice e 15, inclusive, um erro será exibido.

Se o tamanho de saída exceder o limite máximo de tamanho SUPER, um erro será exibido.

Exemplos

O comando SQL a seguir insere um VARCHAR que representa o índice de uma célula H3 e um INTEGER que representa a resolução desejada de todos os filhos e exibe uma matriz SUPER contendo os filhos na resolução 6.

```
SELECT H3_ToChildren('85283473fffffff', 6);
```

```
 h3_tochildren
---------------------------------------------------------------------------------------------
 [604189641121202175,604189641255419903,604189641389637631,604189641523855359,6041896416580730
```

O comando SQL a seguir insere um BIGINT que representa o índice de uma célula H3 e um INTEGER que representa a resolução desejada de todos os filhos e exibe uma matriz SUPER contendo os filhos na resolução 6.

```
SELECT H3_ToChildren(599686042433355775, 6);
```

```
 h3_tochildren
---------------------------------------------------------------------------------------------
 [604189641121202175,604189641255419903,604189641389637631,604189641523855359,6041896416580730
```

Observação: uma diferença de 7 ou menos entre a resolução e a resolução do índice é segura.

O exemplo a seguir demonstra uma solução alternativa para consultas que excedem o limite de tamanho SUPER. Quando a diferença de resolução entre o índice H3 de entrada e a resolução filha desejada é muito grande (maior que 7). Esse procedimento resolve o problema expandindo gradualmente os filhos em etapas menores (máximo de cinco níveis de resolução por vez) e armazenando os resultados finais em uma tabela criada pelo usuário.

```
CREATE OR REPLACE PROCEDURE generate_h3_children()
LANGUAGE plpgsql
AS $$
BEGIN
    -- Drop and create h3_children table that will contain the results
    DROP TABLE IF EXISTS h3_children;
    CREATE TABLE h3_children (
        h3_index BIGINT,
        child_res INTEGER,
        children SUPER
    );

    -- Create temporary table for steps
    DROP TABLE IF EXISTS h3_steps;
    CREATE TABLE h3_steps (
        h3_index BIGINT,
        current_res INTEGER,
        target_res INTEGER,
```

```
        h3_array SUPER
    );

    -- Initial insert into h3_steps
    INSERT INTO h3_steps
    SELECT h3_index, H3_Resolution(h3_index), child_res, ARRAY(h3_index)
    FROM h3_indexes; -- Insert from your table with h3_index and child_res as columns

    -- Loop until we reach target resolution
    -- We expect at most 3 iterations considering that we can start at resolution
    -- 0 and target/child resolution equal to 15 (0 -> 5 -> 10 -> 15)
    WHILE EXISTS (
        SELECT 1
        FROM h3_steps h
        GROUP BY h3_index, target_res
        HAVING MAX(current_res) < target_res
    ) LOOP
        -- Populate the h3_steps table with the tables that need to
        -- reach closer to the target res
        INSERT INTO h3_steps
        SELECT
            h.h3_index,
            LEAST(h.current_res + 5, h.target_res), -- Do not exceed target res
            h.target_res,
            -- Take the children of the child cell at resolution current_res of the
            -- h3_index
            H3_ToChildren(c.child::BIGINT, LEAST(h.current_res + 5, h.target_res))
        FROM h3_steps h, UNNEST(h.h3_array) AS c(child)
        WHERE h.current_res < h.target_res
        AND h.current_res = (SELECT MAX(current_res)
                             FROM h3_steps
                             WHERE h3_index = h.h3_index
        );
    END LOOP;

    -- Store final results
    INSERT INTO h3_children
    SELECT h3_index AS h3_index, target_res AS child_res, h3_array AS children
    FROM h3_steps
    WHERE current_res = target_res;
END;
$$;

-- Create the source table for H3_ToChildren queries
```

```
CREATE TABLE h3_indexes (
    h3_index BIGINT,
    child_res INTEGER,
    PRIMARY KEY (h3_index, child_res)
);
INSERT INTO h3_indexes (h3_index, child_res)
VALUES (x'8001fffffffffff'::BIGINT, 11);

-- Execute the procedure
CALL generate_h3_children();

-- View results
SELECT * FROM h3_children;
```

H3_ToParent

H3_ToParent exibe o ID de célula H3 principal em uma resolução principal especificada para determinado índice H3. Para obter informações sobre a indexação H3, consulte H3.

Sintaxe

```
H3_ToParent(index, resolution)
```

Argumentos

Índice do

Um valor de tipo de dados `BIGINT` ou `VARCHAR` que representa o índice de uma célula H3, ou uma expressão que determina o valor de um desses tipos de dados.

resolução

Um valor do tipo de dados `INTEGER` ou uma expressão avaliada como um tipo `INTEGER`. O valor representa a resolução do ID de célula principal. O valor deve ser 0 e a resolução do índice, inclusive.

Tipo de retorno

`BIGINT`: representa o ID de célula H3 do principal.

Se o índice ou a resolução forem NULL, então NULL será exibido.

Se o índice for inválido, será retornado um erro.

Se a resolução for menor que 0 ou maior que a resolução do índice, um erro será exibido.

Exemplos

O comando SQL a seguir insere um VARCHAR que representa o índice de uma célula H3 e um INTEGER que representa a resolução desejada do principal desejado e exibe um BIGINT que representa o principal na resolução 0 da célula H3 de entrada.

```
SELECT H3_ToParent('85283473fffffff', 0);
```

```
 h3_toparent
--------------------
 577199624117288959
```

O comando SQL a seguir insere um BIGINT que representa o índice de uma célula H3 e um INTEGER que representa a resolução desejada do principal desejado e exibe um BIGINT que representa o principal na resolução 0 da célula H3 de entrada.

```
SELECT H3_ToParent(646078419604526808, 8);
```

```
 h3_toparent
--------------------
 614553222213795839
```

ST_AddPoint

ST_AddPoint retorna uma geometria de linestring que é a mesma geometria de entrada com adição de um ponto. Se um índice for fornecido, o ponto será adicionado na posição do índice. Se o índice for -1 ou não for fornecido, o ponto será anexado à linestring.

O índice é baseado em zero. O Spatial Reference System Identifier (SRID – Identificador do sistema de referência espacial) do resultado é o mesmo da geometria de entrada.

A dimensão da geometria retornada é a mesma do valor de geom1. Se geom1 e geom2 têm dimensões diferentes, geom2 é projetado para a dimensão de geom1.

Sintaxe

```
ST_AddPoint(geom1, geom2)
```

```
ST_AddPoint(geom1, geom2, index)
```

Argumentos

geom1

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY. O subtipo deve ser LINESTRING.

geom2

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY. O subtipo deve ser POINT. O ponto pode ser o ponto vazio.

Índice do

Um valor do tipo de dados INTEGER que representa a posição de um índice baseado em zero.

Tipo de retorno

GEOMETRY

Se geom1, geom2 ou index for null, será retornado null.

Se geom2 é o ponto vazio, então uma cópia do geom1 é retornado.

Se geom1 não for uma LINESTRING, será retornado um erro.

Se geom2 não for um POINT, será retornado um erro.

Se index estiver fora do intervalo, será retornado um erro. Os valores válidos para a posição do índice são -1 ou um valor entre 0 e ST_NumPoints(geom1).

Exemplos

O SQL a seguir adiciona um ponto a uma linestring para torná-la uma linestring fechada.

```
WITH tmp(g) AS (SELECT ST_GeomFromText('LINESTRING(0 0,10 0,10 10,5 5,0 5)',4326))
SELECT ST_AsEWKT(ST_AddPoint(g, ST_StartPoint(g))) FROM tmp;
```

```
 st_asewkt
-------------------------------------------------
 SRID=4326;LINESTRING(0 0,10 0,10 10,5 5,0 5,0 0)
```

O SQL a seguir adiciona um ponto a uma posição específica em uma linestring.

```
WITH tmp(g) AS (SELECT ST_GeomFromText('LINESTRING(0 0,10 0,10 10,5 5,0 5)',4326))
SELECT ST_AsEWKT(ST_AddPoint(g, ST_SetSRID(ST_Point(5, 10), 4326), 3)) FROM tmp;
```

```
 st_asewkt
-------------------------------------------------
 SRID=4326;LINESTRING(0 0,10 0,10 10,5 10,5 5,0 5)
```

ST_Angle

ST_Angle retorna o ângulo em radianos entre pontos medidos no sentido horário da seguinte forma:

- Se forem introduzidos três pontos, o ângulo retornado P1-P2-P3 é medido como se o ângulo fosse obtido girando de P1 para P3 em torno de P2 no sentido horário.
- Se quatro pontos forem inseridos, o ângulo retornado no sentido horário formado pelas linhas direcionadas P1-P2 e P3-P4 é retornado. Se a entrada for um caso degenerado (ou seja, P1 é igual a P2 ou P3 igual a P4), então null é retornado.

O valor de retorno é em radianos e no intervalo [0, 2π).

ST_Angle opera em projeções 2D das geometrias de entrada.

Sintaxe

```
ST_Angle(geom1, geom2, geom3)
```

```
ST_Angle(geom1, geom2, geom3, geom4)
```

Argumentos

geom1

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`. O subtipo deve ser `POINT`.

geom2

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`. O subtipo deve ser `POINT`.

geom3

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`. O subtipo deve ser `POINT`.

geom4

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`. O subtipo deve ser `POINT`.

Tipo de retorno

`DOUBLE PRECISION`.

Se geom1 for igual a geom2 ou geom2 for igual a geom3, será exibido null.

Se geom1, geom2, geom3 ou geom4 forem nulos, será exibido null.

Se geom1, geom2, geom3 ou geom4 forem um ponto vazio, será exibido um erro.

Se geom1, geom2, geom3 e geom4 não tiverem o mesmo valor de identificador do sistema de referência espacial (SRID), será exibido um erro.

Exemplos

O SQL a seguir retorna o ângulo convertido em graus de três pontos de entrada.

```
SELECT ST_Angle(ST_Point(1,1), ST_Point(0,0), ST_Point(1,0)) / Pi() * 180.0 AS angle;
```

```
 angle
---------------
```

```
    45
```

O SQL a seguir retorna o ângulo convertido em graus de quatro pontos de entrada.

```
SELECT ST_Angle(ST_Point(1,1), ST_Point(0,0), ST_Point(1,0), ST_Point(2,0)) / Pi() *
 180.0 AS angle;
```

```
 angle
---------------
   225
```

ST_Area

Para uma geometria de entrada, o ST_Area retorna a área cartesiana da projeção 2D. As unidades de área são as mesmas unidades em que as coordenadas da geometria de entrada são expressas. Para pontos, linestrings, multipontos e multilinestrings, a função retorna 0. Para coleções de geometrias, retorna a soma das áreas das geometrias presentes na coleção.

Para uma geografia de entrada, o ST_Area retorna a área geodésica da projeção 2D de uma geografia plana de entrada computada no esferoide determinado pelo SRID. A unidade de comprimento é o metro quadrado. A função retorna zero (0) para pontos, multipontos e geografias lineares. Quando a entrada for uma coleção de geometrias, a função retornará a soma das áreas das geografias planas na coleção.

Sintaxe

```
ST_Area(geo)
```

Argumentos

geo

Um valor de tipo de dados `GEOMETRY` ou `GEOGRAPHY` ou uma expressão que é avaliada como tipo `GEOMETRY` ou `GEOGRAPHY`.

Tipo de retorno

`DOUBLE PRECISION`

Será retornado null, se geo for nulo.

Exemplos

O SQL a seguir retorna a área cartesiana de um multipolígono.

```
SELECT ST_Area(ST_GeomFromText('MULTIPOLYGON(((0 0,10 0,0 10,0 0)),((10 0,20 0,20 10,10
 0)))'));
```

```
 st_area
---------
     100
```

O SQL a seguir retorna a área de um polígono em uma geografia.

```
SELECT ST_Area(ST_GeogFromText('polygon((34 35, 28 30, 25 34, 34 35))'));
```

```
     st_area
------------------
 201824655743.383
```

O SQL a seguir retorna zero para uma geografia linear.

```
SELECT ST_Area(ST_GeogFromText('multipoint(0 0, 1 1, -21.32 121.2)'));
```

```
 st_area
---------
       0
```

ST_AsBinary

ST_AsBinary retorna a representação do binário hexadecimal conhecido de uma geometria de entrada. Para geometrias 3DZ, 3DM e 4D, ST_AsBinary usa o valor padrão Open Geospatial Consortium (OGC) para o tipo de geometria.

Sintaxe

```
ST_AsBinary(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

VARBYTE

Nulo será retornado se geom for nulo.

Exemplos

O seguinte SQL retorna a representação de hexadecimal WKB de um polígono.

```
SELECT ST_AsBinary(ST_GeomFromText('POLYGON((0 0,0 1,1 1,1 0,0 0))',4326));
```

```
st_asbinary
--------------------------------
0103000000010000000500000000000000000000000000000000000000000000000000F03F00000
```

ST_AsEWKB

ST_AsEWKB retorna a representação binária conhecida estendida (EWKB) de uma geometria de entrada. Para geometrias 3DZ, 3DM e 4D, ST_AsEWKB usa o valor padrão Open Geospatial Consortium (OGC) para o tipo de geometria.

Sintaxe

```
ST_AsEWKB(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

VARBYTE

Nulo será retornado se geom for nulo.

Exemplos

O seguinte SQL retorna a representação EWKB hexadecimal de um polígono.

```
SELECT ST_AsEWKB(ST_GeomFromText('POLYGON((0 0,0 1,1 1,1 0,0 0))',4326));
```

```
st_asewkb
--------------------------------
0103000020E6100000010000000500000000000000000000000000000000000000000000000000F0
```

ST_AsEWKT

ST_AsEWKT retorna a representação de texto reconhecido estendido (EWKT) de uma geometria ou geografia de entrada. Para geometrias 3DZ, 3DM e 4D, ST_AsEWKT acrescenta Z, M ou ZM ao valor WKT para o tipo de geometria.

Sintaxe

```
ST_AsEWKT(geo)
```

```
ST_AsEWKT(geo, precision)
```

Argumentos

geo

Um valor de tipo de dados GEOMETRY ou GEOGRAPHY ou uma expressão que é avaliada como tipo GEOMETRY ou GEOGRAPHY.

precisão

Um valor de de tipo de dados INTEGER. Para geometrias, as coordenadas geo são exibidas usando a precisão especificada de 1 a 20. Se precision não for especificado, o padrão será 15. Para geometrias, as coordenadas geo são exibidas usando a precisão especificada. Se precision não for especificado, o padrão será 15.

Tipo de retorno

VARCHAR

Será retornado null, se geo for nulo.

Nulo será retornado se precision for nulo.

Se o resultado for maior que VARCHAR de 64 KB, um erro será retornado.

Exemplos

O seguinte SQL retorna a representação EWKT de uma linestring.

```
SELECT ST_AsEWKT(ST_GeomFromText('LINESTRING(3.141592653589793
 -6.283185307179586,2.718281828459045 -1.414213562373095)', 4326));
```

```
st_asewkt
-------------------------------
 SRID=4326;LINESTRING(3.14159265358979 -6.28318530717959,2.71828182845905
 -1.41421356237309)
```

O seguinte SQL retorna a representação EWKT de uma linestring. As coordenadas das geometrias são exibidas com seis dígitos de precisão.

```
SELECT ST_AsEWKT(ST_GeomFromText('LINESTRING(3.141592653589793
 -6.283185307179586,2.718281828459045 -1.414213562373095)', 4326), 6);
```

```
st_asewkt
-------------------------------
 SRID=4326;LINESTRING(3.14159 -6.28319,2.71828 -1.41421)
```

O SQL a seguir retorna a representação de EWKT de uma geografia.

```
SELECT ST_AsEWKT(ST_GeogFromText('LINESTRING(110 40, 2 3, -10 80, -7 9)'));
```

```
                  st_asewkt
---------------------------------------------
 SRID=4326;LINESTRING(110 40,2 3,-10 80,-7 9)
```

ST_AsGeoJSON

ST_AsGeoJSON retorna a representação GeoJSON de uma geometria ou geografia de entrada. Para mais informações sobre GeoJSON, consulte GeoJSON na Wikipédia.

Para geometrias 3DZ e 4D, a geometria de saída é uma projeção 3DZ da geometria 3DZ ou 4D de entrada. Ou seja, as coordenadas x, y, e z estão presentes na saída. Para geometrias 3DM, a geometria de saída é uma projeção 2D da geometria 3DM de entrada. Ou seja, apenas as coordenadas x e y estão presentes na saída.

Para geografias de entrada, ST_AsGeoJSON retorna a representação GeoJSON de uma geografia de entrada. As coordenadas da geografia são exibidas usando a precisão especificada.

Sintaxe

```
ST_AsGeoJSON(geo)
```

```
ST_AsGeoJSON(geo, precision)
```

Argumentos

geo

Um valor de tipo de dados GEOMETRY ou GEOGRAPHY ou uma expressão que é avaliada como tipo GEOMETRY ou GEOGRAPHY.

precisão

Um valor de de tipo de dados INTEGER. Para geometrias, as coordenadas geo são exibidas usando a precisão especificada de 1 a 20. Se precision não for especificado, o padrão será 15. Para geometrias, as coordenadas geo são exibidas usando a precisão especificada. Se precision não for especificado, o padrão será 15.

Tipo de retorno

VARCHAR

Será retornado null, se geo for nulo.

Nulo será retornado se precision for nulo.

Se o resultado for maior que VARCHAR de 64 KB, um erro será retornado.

Exemplos

O seguinte SQL retorna a representação GeoJSON de uma linestring.

```
SELECT ST_AsGeoJSON(ST_GeomFromText('LINESTRING(3.141592653589793
 -6.283185307179586,2.718281828459045 -1.414213562373095)'));
```

```
st_asgeojson
-----------------------------------------------------------------------------------------------
 {"type":"LineString","coordinates":[[3.14159265358979,-6.28318530717959],
[2.71828182845905,-1.4142135623730 9]]}
```

O seguinte SQL retorna a representação GeoJSON de uma linestring. As coordenadas das geometrias são exibidas com seis dígitos de precisão.

```
SELECT ST_AsGeoJSON(ST_GeomFromText('LINESTRING(3.141592653589793
 -6.283185307179586,2.718281828459045 -1.414213562373095)'), 6);
```

```
st_asgeojson
-------------------------------------------------------------------------
  {"type":"LineString","coordinates":[[3.14159,-6.28319],[2.71828,-1.41421]]}
```

O SQL a seguir retorna a representação GeoJSON de uma geografia.

```
SELECT ST_AsGeoJSON(ST_GeogFromText('LINESTRING(110 40, 2 3, -10 80, -7 9)'));
```

```
                             st_asgeojson
------------------------------------------------------------------------
 {"type":"LineString","coordinates":[[110,40],[2,3],[-10,80],[-7,9]]}
```

ST_AsHexWKB

ST_AsHexWKB retorna a representação hexadecimal binária reconhecida (WKB) de uma geometria ou geografia de entrada usando caracteres hexadecimais ASCII (0 a 9, A a F). Para geometrias ou geografias 3DZ, 3DM e 4D, ST_AsHexWKB usa o valor padrão Open Geospatial Consortium (OGC) para o tipo de geometria ou geografia.

Sintaxe

```
ST_AsHexWKB(geo)
```

Argumentos

geo

Um valor de tipo de dados GEOMETRY ou GEOGRAPHY ou uma expressão que é avaliada como tipo GEOMETRY ou GEOGRAPHY.

Tipo de retorno

VARCHAR

Será retornado null, se geo for nulo.

Se o resultado for maior que VARCHAR de 64 KB, um erro será retornado.

Exemplos

O SQL a seguir retorna a representação WKB hexadecimal de um polígono em uma geometria.

```
SELECT ST_AsHexWKB(ST_GeomFromText('POLYGON((0 0,0 1,1 1,1 0,0 0))',4326));
```

```
st_ashexwkb
--------------------------------
0103000000010000000500000000000000000000000000000000000000000000000000F03F00000
```

O SQL a seguir retorna a representação WKB hexadecimal de um polígono em uma geografia.

```
SELECT ST_AsHexWKB(ST_GeogFromText('POLYGON((0 0,0 1,1 1,1 0,0 0))'));
```

```
st_ashexwkb
--------------------------------
0103000000010000000500000000000000000000000000000000000000000000000000F03F00000
```

ST_AsHexEWKB

ST_AsHexEWKB retorna a representação hexadecimal binária reconhecida estendida (EWKB) de uma geometria ou geografia de entrada usando caracteres hexadecimais ASCII (0 a 9, A a F). Para geometrias e geografias 3DZ, 3DM e 4D, ST_AsHexEWKB usa o valor de WKB estendido PostGIS para o tipo de geometria ou geografia.

Sintaxe

```
ST_AsHexEWKB(geo)
```

Argumentos

geo

Um valor de tipo de dados GEOMETRY ou GEOGRAPHY ou uma expressão que é avaliada como tipo GEOMETRY ou GEOGRAPHY.

Tipo de retorno

VARCHAR

Será retornado null, se geo for nulo.

Se o resultado for maior que VARCHAR de 64 KB, um erro será retornado.

Exemplos

O SQL a seguir retorna a representação EWKB hexadecimal de um polígono em uma geometria.

```
SELECT ST_AsHexEWKB(ST_GeomFromText('POLYGON((0 0,0 1,1 1,1 0,0 0))',4326));
```

```
st_ashexewkb
--------------------------------
0103000020E6100000010000000500000000000000000000000000000000000000000000000000F
```

O SQL a seguir retorna a representação EWKB hexadecimal de um polígono em uma geografia.

```
SELECT ST_AsHexEWKB(ST_GeogFromText('POLYGON((0 0,0 1,1 1,1 0,0 0))'));
```

```
st_ashexewkb
--------------------------------
0103000020E6100000010000000500000000000000000000000000000000000000000000000000F
```

ST_AsText

ST_AsText retorna a representação de texto reconhecido (WKT) de uma geometria ou geografia de entrada. Para geometrias ou geografias 3DZ, 3DM e 4D, ST_AsEWKT acrescenta Z, M ou ZM ao valor de WKT para o tipo de geometria ou geografia.

Sintaxe

```
ST_AsText(geo)
```

```
ST_AsText(geo, precision)
```

Argumentos

geo

Um valor de tipo de dados GEOMETRY ou GEOGRAPHY ou uma expressão que é avaliada como tipo GEOMETRY ou GEOGRAPHY.

precisão

Um valor de de tipo de dados INTEGER. Para geometrias, as coordenadas geo são exibidas usando a precisão especificada de 1 a 20. Se precision não for especificado, o padrão será 15. As coordenadas de geo são exibidas usando a precisão especificada. Se precision não for especificado, o padrão será 15.

Tipo de retorno

VARCHAR

Será retornado null, se geo for nulo.

Nulo será retornado se precision for nulo.

Se o resultado for maior que VARCHAR de 64 KB, um erro será retornado.

Exemplos

O seguinte SQL retorna a representação WKT de uma linestring.

```
SELECT ST_AsText(ST_GeomFromText('LINESTRING(3.141592653589793
 -6.283185307179586,2.718281828459045 -1.414213562373095)', 4326));
```

```
st_astext
-------------------------------
LINESTRING(3.14159265358979 -6.28318530717959,2.71828182845905 -1.41421356237309)
```

O seguinte SQL retorna a representação WKT de uma linestring. As coordenadas das geometrias são exibidas com seis dígitos de precisão.

```
SELECT ST_AsText(ST_GeomFromText('LINESTRING(3.141592653589793
 -6.283185307179586,2.718281828459045 -1.414213562373095)', 4326), 6);
```

```
st_astext
---------------------------------------------
 LINESTRING(3.14159 -6.28319,2.71828 -1.41421)
```

O SQL a seguir retorna a representação de WKT de uma geografia.

```
SELECT ST_AsText(ST_GeogFromText('LINESTRING(110 40, 2 3, -10 80, -7 9)'));
```

```
             st_astext
-----------------------------------
 LINESTRING(110 40,2 3,-10 80,-7 9)
```

ST_Azimuth

ST_Azimuth retorna o azimute cartesiano baseado no norte de dois pontos de entrada.

Sintaxe

```
ST_Azimuth(point1, point2)
```

Argumentos

point1

Um valor de `POINT` de tipo de dados `GEOMETRY`. O identificador do sistema de referência espacial (SRID - spatial reference system identifier) do point1 deve corresponder ao SRID do point2.

point2

Um valor de `POINT` de tipo de dados `GEOMETRY`. O SRID de point2 deve corresponder ao SRID de point1.

Tipo de retorno

Um número que é um ângulo em radianos do tipo de dados `DOUBLE PRECISION`. Os valores variam de 0 (inclusivo) a 2 pi (exclusivo).

Um erro será retornado se point1 ou point2 não for um ponto.

Nulo será retornado se point1 ou point2 for nulo.

Nulo será retornado se point1 e point2 forem iguais.

Um erro será retornado se point1 ou point2 não for um ponto.

Um erro será retornado se geom1 e geom2 não tiverem o mesmo valor do identificador do sistema referência espacial (SRID -spatial reference system identifier).

Exemplos

O seguinte SQL retorna o azimute dos pontos de entrada.

```
SELECT ST_Azimuth(ST_Point(1,2), ST_Point(5,6));
```

```
st_azimuth
-------------------
 0.7853981633974483
```

ST_Boundary

ST_Boundary retorna o limite de uma geometria de entrada da seguinte forma:

- Se a geometria de entrada estiver vazia (isto é, ela não contém pontos), ela é retornada como está.
- Se a geometria de entrada for um ponto ou um multiponto não vazio, uma coleção de geometria vazia será retornada.
- Se a entrada for uma linestring ou uma multilinestring, então um multiponto contendo todos os pontos no limite é retornado. O multiponto pode estar vazio).
- Se a entrada for um polígono que não possui anéis interiores, então uma linestring fechada representando seu limite é retornada.
- Se a entrada é um polígono que tem anéis interiores, ou é um multipolígono, então uma multilinestring é retornada. A multilinestring contém todos os limites de todos os anéis na geometria de área como linhas fechadas.

Para determinar a igualdade de pontos, ST_Boundary opera na projeção 2D da geometria de entrada. Se a geometria de entrada estiver vazia, será retornado uma cópia na mesma dimensão da entrada. Para geometrias 3DM e 4D não vazias, suas coordenadas `m` são descartadas. No caso especial de 3DZ e 4D multilinestrings, as coordenadas `z` dos pontos de limite da multilinestring são calculadas como as médias dos valores z distintos dos pontos de limite de linestring com a mesma projeção 2D.

Sintaxe

```
ST_Boundary(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

GEOMETRY

Nulo será retornado se geom for nulo.

Se geom for uma GEOMETRYCOLLECTION, será retornado um erro.

Exemplos

O SQL a seguir retorna o limite do polígono de entrada como uma multilinestring.

```
SELECT ST_AsEWKT(ST_Boundary(ST_GeomFromText('POLYGON((0 0,10 0,10 10,0 10,0 0),(1 1,1
 2,2 1,1 1))')));
```

```
st_asewkt
--------------------
 MULTILINESTRING((0 0,10 0,10 10,0 10,0 0),(1 1,1 2,2 1,1 1))
```

ST_Buffer

ST_Buffer retorna uma forma geométrica 2D que representa todos os pontos cuja distância da forma geométrica inserida projetada no plano cartesiano xy é menor ou igual à distância inserida.

Sintaxe

```
ST_Buffer(geom, distance)
```

```
ST_Buffer(geom, distance, number_of_segments_per_quarter_circle)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

distance

Um valor de tipo de dado DOUBLE PRECISION que representa a distância (ou raio) do buffer.

number_of_segments_per_quarter_circle

Um valor de de tipo de dados INTEGER. Esse valor determina o número de pontos para aproximar um quarto de círculo em torno de cada vértice da forma geométrica inserida. Valores negativos assumem o valor zero por padrão. O padrão é 8.

Tipo de retorno

GEOMETRY

A função ST_Buffer retorna uma forma geométrica bidimensional (2D) no plano cartesiano xy.

Se geom for uma GEOMETRYCOLLECTION, será retornado um erro.

Exemplos

O comando SQL a seguir retorna o buffer da linestring inserida.

```
SELECT ST_AsEwkt(ST_Buffer(ST_GeomFromText('LINESTRING(1 2,5 2,5 8)'), 2));
```

```
               st_asewkt
POLYGON((-1 2,-0.96157056080646 2.39018064403226,-0.847759065022573
 2.76536686473018,-0.662939224605089 3.11114046603921,-0.414213562373093
 3.4142135623731,-0.111140466039201 3.66293922460509,0.234633135269824
 3.84775906502257,0.609819355967748 3.96157056080646,1 4,3 4,3 8,3.03842943919354
 8.39018064403226,3.15224093497743 8.76536686473018,3.33706077539491
 9.11114046603921,3.58578643762691 9.4142135623731,3.8888595339608
 9.66293922460509,4.23463313526982 9.84775906502257,4.60981935596775
 9.96157056080646,5 10,5.39018064403226 9.96157056080646,5.76536686473018
 9.84775906502257,6.11114046603921 9.66293922460509,6.4142135623731
 9.41421356237309,6.66293922460509 9.1111404660392,6.84775906502258
 8.76536686473017,6.96157056080646 8.39018064403225,7 8,7 2,6.96157056080646
```

```
 1.60981935596774,6.84775906502257 1.23463313526982,6.66293922460509
 0.888859533960796,6.41421356237309 0.585786437626905,6.1111404660392
 0.33706077539491,5.76536686473018 0.152240934977427,5.39018064403226
 0.0384294391935391,5 0,1 0,0.609819355967744 0.0384294391935391,0.234633135269821
 0.152240934977427,-0.111140466039204 0.337060775394909,-0.414213562373095
 0.585786437626905,-0.662939224605091 0.888859533960796,-0.847759065022574
 1.23463313526982,-0.961570560806461 1.60981935596774,-1 2))
```

O comando SQL a seguir retorna o buffer da forma geométrica de ponto inserida, que se aproxima de um círculo. Como o comando não especifica o número de segmentos por quarto de círculo, a função usa o valor padrão de oito segmentos para aproximar o quarto de círculo.

```
SELECT ST_AsEwkt(ST_Buffer(ST_GeomFromText('POINT(3 4)'), 2));
```

```
               st_asewkt
POLYGON((1 4,1.03842943919354 4.39018064403226,1.15224093497743
 4.76536686473018,1.33706077539491 5.11114046603921,1.58578643762691
 5.4142135623731,1.8888595339608 5.66293922460509,2.23463313526982
 5.84775906502257,2.60981935596775 5.96157056080646,3 6,3.39018064403226
 5.96157056080646,3.76536686473019 5.84775906502257,4.11114046603921
 5.66293922460509,4.4142135623731 5.41421356237309,4.66293922460509
 5.1111404660392,4.84775906502258 4.76536686473017,4.96157056080646 4.39018064403225,5
 4,4.96157056080646 3.60981935596774,4.84775906502257 3.23463313526982,4.66293922460509
 2.8888595339608,4.41421356237309 2.58578643762691,4.1111404660392
 2.33706077539491,3.76536686473018 2.15224093497743,3.39018064403226 2.03842943919354,3
 2,2.60981935596774 2.03842943919354,2.23463313526982 2.15224093497743,1.8888595339608
 2.33706077539491,1.58578643762691 2.58578643762691,1.33706077539491
 2.8888595339608,1.15224093497743 3.23463313526982,1.03842943919354 3.60981935596774,1
 4))
```

O comando SQL a seguir retorna o buffer da forma geométrica de ponto inserida, que se aproxima de um círculo. Como o comando especifica 3 como o número de segmentos por quarto de círculo, a função usa três segmentos para aproximar o quarto de círculo.

```
SELECT ST_AsEwkt(ST_Buffer(ST_GeomFromText('POINT(3 4)'), 2, 3));
```

```
               st_asewkt
POLYGON((1 4,1.26794919243112 5,2 5.73205080756888,3 6,4
 5.73205080756888,4.73205080756888 5,5 4,4.73205080756888 3,4 2.26794919243112,3 2,2
 2.26794919243112,1.26794919243112 3,1 4))
```

ST_Centroid

ST_Centroid retorna um ponto que representa um centroide de uma geometria:

- Para geometrias `POINT`, retorna o ponto cujas coordenadas são a média das coordenadas dos pontos da geometria.
- Para geometrias `LINESTRING`, retorna o ponto cujas coordenadas são a média ponderada dos pontos médios dos segmentos da geometria, onde os pesos são os comprimentos dos segmentos da geometria.
- Para geometrias `POLYGON`, retorna o ponto cujas coordenadas são a média ponderada dos centroides de uma triangulação da geometria plana onde os pesos são as áreas dos triângulos na triangulação.
- Para coleções de geometrias, retorna a média ponderada dos centroides das geometrias da dimensão topológica máxima na coleção de geometrias.

Sintaxe

```
ST_Centroid(geom)
```

Argumentos

geom

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`.

Tipo de retorno

`GEOMETRY`

Nulo será retornado se geom for nulo.

Nulo será retornado se geom estiver vazio.

Exemplos

O SQL a seguir retorna um ponto central de uma linestring de entrada.

```
SELECT ST_AsEWKT(ST_Centroid(ST_GeomFromText('LINESTRING(110 40, 2 3, -10 80, -7 9, -22
 -33)', 4326)))
```

```
                    st_asewkt
-----------------------------------------------------
 SRID=4326;POINT(15.6965103455214 27.0206782881905)
```

ST_Collect

ST_Collect tem duas variantes. Um aceita duas geometrias, e outro aceita uma expressão agregada.

A primeira variante de ST_Collect cria uma geometria das geometrias de entrada. A ordem das geometrias de entrada é preservada. Essa variante funciona da seguinte maneira:

- Se ambas as geometrias de entrada forem pontos, então um `MULTIPOINT`com dois pontos é retornado.
- Se ambas as geometrias de entrada são linestrings, então um `MULTILINESTRING` com dois linestrings é retornado.
- Se ambas as geometrias de entrada forem polígonos, então um `MULTIPOLYGON` com dois polígonos é retornado.
- Caso contrário, um `GEOMETRYCOLLECTION` com duas geometrias de entrada é retornado.

A segunda variante de ST_Collect cria uma geometria a partir de geometrias em uma coluna de geometria. Não há uma determinada ordem de retorno das geometrias. Especifique a cláusula WITHIN GROUP (ORDER BY...) para especificar a ordem das geometrias retornadas. Essa variante funciona da seguinte maneira:

- Se todas as linhas não-NULL na expressão agregada de entrada forem pontos, um multiponto contendo todos os pontos na expressão agregada será retornado.
- Se todas as linhas não-NULL na expressão agregada forem linestrings, uma multilinestring contendo todas as linhas na expressão agregada será retornada.
- Se todas as linhas não-NULL na expressão agregada forem polígonos, o resultado será um multipolígono contendo todos os polígonos na expressão agregada será retornado.
- Caso contrário, uma `GEOMETRYCOLLECTION` contendo todas as geometrias na expressão agregada é retornada.

O ST_Collect retorna a geometria da mesma dimensão que as geometrias de entrada. Todas as geometrias de entrada devem ser da mesma dimensão.

Sintaxe

```
ST_Collect(geom1, geom2)
```

```
ST_Collect(aggregate_expression)  [WITHIN GROUP (ORDER BY sort_expression1 [ASC | DESC]
 [, sort_expression2 [ASC | DESC] ...])]
```

Argumentos

geom1

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

geom2

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

aggregate_expression

Uma coluna de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

[WITHIN GROUP (ORDER BY sort_expression1 [ASC | DESC] [, sort_expression2 [ASC | DESC] ...])]

Uma cláusula opcional que especifica a ordem de classificação dos valores agregados. A cláusula ORDER BY contém uma lista de expressões de classificação. Expressões de classificação são expressões semelhantes a expressões de classificação válidas em uma lista de seleção de consulta, como um nome de coluna. Você pode especificar por ordem crescente (ASC) ou decrescente (DESC). O padrão é ASC.

Tipo de retorno

GEOMETRY de subtipo MULTIPOINT, MULTILINESTRING, MULTIPOLYGON, ou GEOMETRYCOLLECTION.

O valor do identificador do sistema de referência espacial (SRID - spatial reference system identifier) da geometria retornada é o valor do SRID das geometrias de entrada.

Nulo será retornado se geom1 ou geom2 for nulo.

Se todas as linhas de aggregate_expression forem nulas, então null será retornado.

Null será retornado se geom1 ou geom2 for nulo. Da mesma forma, segeom2 for nulo, então uma cópia d ogeom1 é retornado.

Um erro será retornado se geom1 e geom2 tiverem valores de SRIDs diferentes.

Se duas geometrias em aggregate_expression tiverem valores de SRID diferentes, será retornado um erro.

Se a geometria retornada for maior que o tamanho máximo de um`GEOMETRY`, um erro será retornado.

Se geom1 e geom2 forem de dimensões diferentes, será retornado um erro.

Se duas geometrias em aggregate_expression forem de dimensões diferentes, será retornado um erro.

Exemplos

O SQL a seguir retorna uma coleção de geometrias que contém as duas geometrias de entrada.

```
SELECT ST_AsText(ST_Collect(ST_GeomFromText('LINESTRING(0 0,1 1)'),
 ST_GeomFromText('POLYGON((10 10,20 10,10 20,10 10))')));
```

```
st_astext
-----------
 GEOMETRYCOLLECTION(LINESTRING(0 0,1 1),POLYGON((10 10,20 10,10 20,10 10)))
```

O SQL a seguir coleta todas as geometrias de uma tabela em uma coleção de geometria.

```
WITH tbl(g) AS (SELECT ST_GeomFromText('POINT(1 2)', 4326) UNION ALL
SELECT ST_GeomFromText('LINESTRING(0 0,10 0)', 4326) UNION ALL
SELECT ST_GeomFromText('MULTIPOINT(13 4,8 5,4 4)', 4326) UNION ALL
```

```
SELECT NULL::geometry UNION ALL
SELECT ST_GeomFromText('POLYGON((0 0,10 0,0 10,0 0))', 4326))
SELECT ST_AsEWKT(ST_Collect(g)) FROM tbl;
```

```
st_astext
-----------
 SRID=4326;GEOMETRYCOLLECTION(POINT(1 2),LINESTRING(0 0,10 0),MULTIPOINT((13 4),(8 5),
(4 4)),POLYGON((0 0,10 0,0 10,0 0)))
```

O SQL a seguir coleta todas as geometrias na tabela agrupadas pela coluna id e ordenadas por esse ID. Neste exemplo, as geometrias resultantes são agrupadas por ID da seguinte forma:

- id 1 — pontos em um multiponto.
- id 2 — linestrings em uma multilinestring.
- id 3 — subtipos mistos em uma coleção de geometrias.
- id 4 — polígonos em um multipolígono.
- id 5 — nulo e o resultado será null.

```
WITH tbl(id, g) AS (SELECT 1, ST_GeomFromText('POINT(1 2)', 4326) UNION ALL
SELECT 1, ST_GeomFromText('POINT(4 5)', 4326) UNION ALL
SELECT 2, ST_GeomFromText('LINESTRING(0 0,10 0)', 4326) UNION ALL
SELECT 2, ST_GeomFromText('LINESTRING(10 0,20 -5)', 4326) UNION ALL
SELECT 3, ST_GeomFromText('MULTIPOINT(13 4,8 5,4 4)', 4326) UNION ALL
SELECT 3, ST_GeomFromText('MULTILINESTRING((-1 -1,-2 -2),(-3 -3,-5 -5))', 4326) UNION
 ALL
SELECT 4, ST_GeomFromText('POLYGON((0 0,10 0,0 10,0 0))', 4326) UNION ALL
SELECT 4, ST_GeomFromText('POLYGON((20 20,20 30,30 20,20 20))', 4326) UNION ALL
SELECT 1, NULL::geometry UNION ALL SELECT 2, NULL::geometry UNION ALL
SELECT 5, NULL::geometry UNION ALL SELECT 5, NULL::geometry)
SELECT id, ST_AsEWKT(ST_Collect(g)) FROM tbl GROUP BY id ORDER BY id;
```

```
 id |                                                st_asewkt

----
+-----------------------------------------------------------------------------------------
  1 | SRID=4326;MULTIPOINT((1 2),(4 5))
  2 | SRID=4326;MULTILINESTRING((0 0,10 0),(10 0,20 -5))
```

```
  3 | SRID=4326;GEOMETRYCOLLECTION(MULTIPOINT((13 4),(8 5),(4 4)),MULTILINESTRING((-1
 -1,-2 -2),(-3 -3,-5 -5)))
  4 | SRID=4326;MULTIPOLYGON(((0 0,10 0,0 10,0 0)),((20 20,20 30,30 20,20 20)))
  5 |
```

O SQL a seguir coleta todas as geometrias de uma tabela em uma coleção de geometrias. Os resultados são ordenados em ordem decrescente por `id` e, em seguida, lexicograficamente com base em suas coordenadas x mínimas e máximas.

```
WITH tbl(id, g) AS (
SELECT 1, ST_GeomFromText('POINT(4 5)', 4326) UNION ALL
SELECT 1, ST_GeomFromText('POINT(1 2)', 4326) UNION ALL
SELECT 2, ST_GeomFromText('LINESTRING(10 0,20 -5)', 4326) UNION ALL
SELECT 2, ST_GeomFromText('LINESTRING(0 0,10 0)', 4326) UNION ALL
SELECT 3, ST_GeomFromText('MULTIPOINT(13 4,8 5,4 4)', 4326) UNION ALL
SELECT 3, ST_GeomFromText('MULTILINESTRING((-1 -1,-2 -2),(-3 -3,-5 -5))', 4326) UNION
 ALL
SELECT 4, ST_GeomFromText('POLYGON((20 20,20 30,30 20,20 20))', 4326) UNION ALL
SELECT 4, ST_GeomFromText('POLYGON((0 0,10 0,0 10,0 0))', 4326) UNION ALL
SELECT 1, NULL::geometry UNION ALL SELECT 2, NULL::geometry UNION ALL
SELECT 5, NULL::geometry UNION ALL SELECT 5, NULL::geometry)
SELECT ST_AsEWKT(ST_Collect(g) WITHIN GROUP (ORDER BY id DESC, ST_XMin(g), ST_XMax(g)))
 FROM tbl;
```

```
                          st_asewkt

-----------------------------------------------------------------------------------------
 SRID=4326;GEOMETRYCOLLECTION(POLYGON((0 0,10 0,0 10,0 0)),POLYGON((20 20,20 30,30
 20,20 20)),MULTILINESTRING((-1 -1,-2 -2),(-3 -3,-5 -5)),MULTIPOINT((13 4),(8 5),(4
 4)),LINESTRING(0 0,10 0),LINESTRING(10 0,20 -5),POINT(1 2),POINT(4 5)
```

ST_Contains

ST_Contains retornará true se a primeira geometria de entrada contiver a segunda geometria de entrada. A geometria A conterá a geometria B se todos os pontos em B forem um ponto em A e seus interiores tiverem interseção não vazia.

ST_Contains(A, B) é equivalente a ST_Within(B, A).

Sintaxe

```
ST_Contains(geom1, geom2)
```

Argumentos

geom1

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

geom2

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY. Esse valor é comparado com geom1 para determinar se ele está contido no geom1.

Tipo de retorno

BOOLEAN

Nulo será retornado se geom1 ou geom2 for nulo.

Um erro será retornado se geom1 e geom2 não tiverem o mesmo valor do identificador do sistema referência espacial (SRID -spatial reference system identifier).

Um erro será retornado se geom1 ou geom2 for uma coleção de geometrias.

Exemplos

O SQL a seguir verifica se o primeiro polígono contém o segundo polígono.

```
SELECT ST_Contains(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'),
 ST_GeomFromText('POLYGON((-1 3,2 1,0 -3,-1 3))'));
```

```
st_contains
-----------
 false
```

ST_ContainsProperly

ST_ContainsProperly retorna true se ambas as geometrias de entrada não estiverem vazias, e todos os pontos da projeção 2D da segunda geometria são pontos interiores da projeção 2D da primeira geometria.

Sintaxe

```
ST_ContainsProperly(geom1, geom2)
```

Argumentos

geom1

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY. O subtipo não pode ser GEOMETRYCOLLECTION.

geom2

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY. O subtipo não pode ser GEOMETRYCOLLECTION. Este valor é comparado com geom1 para determinar se todos os seus pontos são pontos interiores de geom1.

Tipo de retorno

BOOLEAN

Nulo será retornado se geom1 ou geom2 for nulo.

Um erro será retornado se geom1 e geom2 não tiverem o mesmo valor do identificador do sistema referência espacial (SRID -spatial reference system identifier).

Um erro será retornado se geom1 ou geom2 for uma coleção de geometrias.

Exemplos

O SQL a seguir retorna os valores de ST_Contains e ST_ContainsProperly onde a linestring intersecta o interior e o limite do polígono de entrada (mas não seu exterior). O polígono contém a linestring, mas não contém corretamente a linestring.

```
WITH tmp(g1, g2)
```

```
AS (SELECT ST_GeomFromText('POLYGON((0 0,10 0,10 10,0 10,0 0))'),
 ST_GeomFromText('LINESTRING(5 5,10 5,10 6,5 5)')) SELECT ST_Contains(g1, g2),
 ST_ContainsProperly(g1, g2)
FROM tmp;
```

```
 st_contains | st_containsproperly
-------------+---------------------
 t           | f
```

ST_ConvexHull

ST_ConvexHull retorna uma geometria que representa o casco convexo dos pontos não vazios contidos na geometria de entrada.

Para entrada vazia, a geometria resultante é a mesma geometria de entrada. Para todas as entradas não vazias, a função opera na projeção 2D da geometria de entrada. No entanto, a dimensão da geometria de saída depende da dimensão da geometria de entrada. Mais especificamente, quando a geometria de entrada é uma geometria 3DM ou 3D não-vazia, coordenadas `m` são descartadas. Ou seja, a dimensão da geometria retornada é 2D ou 3DZ, respectivamente. Se a entrada for uma geometria 2D ou 3DZ não vazia, a geometria resultante terá a mesma dimensão.

Sintaxe

```
ST_ConvexHull(geom)
```

Argumentos

geom

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`.

Tipo de retorno

`GEOMETRY`

O valor do identificador do sistema de referência espacial (SRID - spatial reference system identifier) da geometria retornada é o valor do SRID das geometrias de entrada.

Nulo será retornado se geom for nulo.

Os valores retornados são os seguintes.

Número de pontos no casco convexo	Subtipo de geometria
0	Uma cópia do geom é retornada.
1	Um subtipo `POINT` é retornado.
2	Um subtipo `LINESTRING` é retornado. Os dois pontos do linestring retornado são ordenados lexicograficamente.
3 ou mais	Um subtipo `POLYGON` sem anéis interiores é retornado. O polígono é orientado no sentido horário, e o primeiro ponto do anel externo é o ponto lexicograficamente mais pequeno do anel.

Exemplos

O seguinte SQL retorna a representação de texto conhecido estendido (EWKT) de uma linestring. Nesse caso, o casco convexo retornado é um polígono.

```
SELECT ST_AsEWKT(ST_ConvexHull(ST_GeomFromText('LINESTRING(0 0,1 0,0 1,1 1,0.5 0.5)')))
 as output;
```

```
output
-------------
POLYGON((0 0,0 1,1 1,1 0,0 0))
```

O seguinte SQL retorna a representação EWKT de uma linestring. Nesse caso, o casco convexo retornado é uma linestring.

```
SELECT ST_AsEWKT(ST_ConvexHull(ST_GeomFromText('LINESTRING(0 0,1 1,0.2 0.2,0.6 0.6,0.5
 0.5)'))) as output;
```

```
output
-------------
LINESTRING(0 0,1 1)
```

O seguinte SQL retorna a representação EWKT de um multiponto. Nesse caso, o casco convexo retornado é um ponto.

```
SELECT ST_AsEWKT(ST_ConvexHull(ST_GeomFromText('MULTIPOINT(0 0,0 0,0 0)'))) as output;
```

```
output
-------------
POINT(0 0)
```

ST_CoveredBy

ST_CoveredBy retorna true se a projeção 2D da primeira geometria de entrada for coberta pela projeção 2D da segunda geometria de entrada. A geometria A será coberta pela geometria B se ambas não estiverem vazias e todos os pontos em A forem um ponto em B.

ST_CoveredBy(A, B) é equivalente a ST_Covers(B, A).

Sintaxe

```
ST_CoveredBy(geom1, geom2)
```

Argumentos

geom1

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY. Esse valor é comparado com geom2 para determinar se ele é coberto por geom2.

geom2

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

BOOLEAN

Nulo será retornado se geom1 ou geom2 for nulo.

Um erro será retornado se geom1 e geom2 não tiverem o mesmo valor do identificador do sistema referência espacial (SRID -spatial reference system identifier).

Um erro será retornado se geom1 ou geom2 for uma coleção de geometrias.

Exemplos

O SQL a seguir verifica se o primeiro polígono é coberto pelo segundo polígono.

```
SELECT ST_CoveredBy(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'),
 ST_GeomFromText('POLYGON((-1 3,2 1,0 -3,-1 3))'));
```

```
st_coveredby
------------
 true
```

ST_Covers

ST_Covers retorna true se a projeção 2D da primeira geometria de entrada cobrir a projeção 2D da segunda geometria de entrada. A geometria A cobrirá a geometria B se ambas não estiverem vazias e todos os pontos em B forem um ponto em A.

ST_Covers(A, B) é equivalente a ST_CoveredBy(B, A).

Sintaxe

```
ST_Covers(geom1, geom2)
```

Argumentos

geom1

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

geom2

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY. Esse valor é comparado com geom1 para determinar se ele cobre geom1.

Tipo de retorno

BOOLEAN

Nulo será retornado se geom1 ou geom2 for nulo.

Um erro será retornado se geom1 e geom2 não tiverem o mesmo valor do identificador do sistema referência espacial (SRID -spatial reference system identifier).

Um erro será retornado se geom1 ou geom2 for uma coleção de geometrias.

Exemplos

O SQL a seguir verifica se o primeiro polígono cobre o segundo polígono.

```
SELECT ST_Covers(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'),
 ST_GeomFromText('POLYGON((-1 3,2 1,0 -3,-1 3))'));
```

```
st_covers
-----------
 false
```

ST_Crosses

ST_Crosses retorna true se as projeções 2D das duas geometrias de entrada cruzam entre si.

Sintaxe

```
ST_Crosses(geom1, geom2)
```

Argumentos

geom1

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

geom2

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

BOOLEAN

Um erro será retornado se geom1 ou geom2 for nulo.

Um erro será retornado se geom1 ou geom2 for uma coleção de geometrias.

Um erro será retornado se geom1 e geom2 não tiverem o mesmo valor do identificador do sistema referência espacial (SRID -spatial reference system identifier).

Exemplos

O seguinte SQL verifica se o primeiro polígono cruza o segundo multiponto. Neste exemplo, o multiponto intersecta tanto o interior como o exterior do polígono, razão pela qual ST_Crosses retorna true.

```
SELECT ST_Crosses (ST_GeomFromText('polygon((0 0,10 0,10 10,0 10,0 0))'),
 ST_GeomFromText('multipoint(5 5,0 0,-1 -1)'));
```

```
st_crosses
-------------
 true
```

O seguinte SQL verifica se o primeiro polígono cruza o segundo multiponto. Neste exemplo, o multiponto intersecta o exterior do polígono, mas não o seu interior, razão pela qual ST_Crosses retorna false.

```
SELECT ST_Crosses (ST_GeomFromText('polygon((0 0,10 0,10 10,0 10,0 0))'),
 ST_GeomFromText('multipoint(0 0,-1 -1)'));
```

```
st_crosses
-------------
 false
```

ST_Dimension

ST_Dimension retorna a dimensão inerente de uma geometria de entrada. A dimensão inerente é o valor da dimensão do subtipo que está definido na geometria.

Para entradas de geometria 3DM, 3DZ e 4D, ST_Dimension retorna o mesmo resultado que para entradas de geometria 2D.

Sintaxe

```
ST_Dimension(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

INTEGER que representa a dimensão inerente de geom.

Nulo será retornado se geom for nulo.

Os valores retornados são os seguintes.

Valor retornado	Subtipo de geometria
0	Retornado se geom for um subtipo de POINT ou de MULTIPOINT .
1	Retornado se geom for um subtipo de LINESTRING ou de MULTILINESTRING .
2	Retornado se geom for um subtipo de POLYGON ou de MULTIPOLYGON .
0	Retornado se geom for um subtipo de GEOMETRYCOLLECTION vazio.
Maior dimensão dos componentes da coleção.	Retornado se geom for um subtipo GEOMETRYCOLLECTION

Exemplos

O SQL a seguir converte uma representação de texto bem-conhecido (WKT) de uma LINESTRING de quatro pontos em um objeto GEOMETRY e retorna a dimensão da linestring.

```
SELECT ST_Dimension(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27
 29.31,77.29 29.07)'));
```

```
st_dimension
-------------
1
```

ST_Disjoint

ST_Disjoint retorna true se as projeções 2D das duas geometrias de entrada não tiverem pontos em comum.

Sintaxe

```
ST_Disjoint(geom1, geom2)
```

Argumentos

geom1

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

geom2

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

BOOLEAN

Nulo será retornado se geom1 ou geom2 for nulo.

Um erro será retornado se geom1 e geom2 não tiverem o mesmo valor do identificador do sistema referência espacial (SRID -spatial reference system identifier).

Um erro será retornado se geom1 ou geom2 for uma coleção de geometrias.

Exemplos

O SQL a seguir verifica se o primeiro polígono é separado do segundo polígono.

```
SELECT ST_Disjoint(ST_GeomFromText('POLYGON((0 0,10 0,10 10,0 10,0 0),(2 2,2 5,5 5,5
 2,2 2))'), ST_Point(4, 4));
```

```
st_disjoint
-----------
 true
```

ST_Distance

Para geometrias de entrada, ST_Distance retorna a distância euclidiana mínima entre as projeções 2D dos dois valores de geometria de entrada.

Para geometrias 3DM, 3DZ, 4D, ST_Distance retorna a distância euclidiana entre as projeções 2D de dois valores de geometria de entrada.

Para geografias de entrada, ST_Distance retorna a distância geodésica dos dois pontos 2D. A unidade de distância é metro. Para regiões geográficas diferentes de pontos e pontos vazios, é retornado um erro.

Sintaxe

```
ST_Distance(geo1, geo2)
```

Argumentos

geo1

Um valor de tipo de dados GEOMETRY ou GEOGRAPHY ou uma expressão que é avaliada como tipo GEOMETRY ou GEOGRAPHY. O tipo de dados de geo1 deve ser o mesmo que de geo2.

geo2

Um valor de tipo de dados GEOMETRY ou GEOGRAPHY ou uma expressão que é avaliada como tipo GEOMETRY ou GEOGRAPHY. O tipo de dados de geo2 deve ser o mesmo que de geo1.

Tipo de retorno

DOUBLE PRECISION nas mesmas unidades que as geometrias ou geografias de entrada.

Será retornado null, se geo1 ou geo2 forem nulos ou vazios.

Será retornado um erro se geo1 e geo2 não tiverem o mesmo valor do identificador do sistema de referência espacial (SRID).

Será retornado um erro se geo1 ou geo2 forem uma coleção de geometrias.

Exemplos

O SQL a seguir retorna a distância entre dois polígonos.

```
SELECT ST_Distance(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'),
 ST_GeomFromText('POLYGON((-1 -3,-2 -1,0 -3,-1 -3))'));
```

```
  st_distance
-----------
1.4142135623731
```

O SQL a seguir retorna a distância (em metros) entre o Portão de Brandemburgo e o prédio do Reichstag em Berlim usando o tipo de dado GEOGRAPHY.

```
SELECT ST_Distance(ST_GeogFromText('POINT(13.37761826722198 52.516411678282445)'),
 ST_GeogFromText('POINT(13.377950831464005 52.51705102546893)'));
```

```
   st_distance
------------------
 74.64129172609631
```

ST_DistanceSphere

ST_DistanceSphere retorna a distância entre duas geometrias de pontos encontradas em uma esfera.

Sintaxe

```
ST_DistanceSphere(geom1, geom2)
```

```
ST_DistanceSphere(geom1, geom2, radius)
```

Argumentos

geom1

Um valor de ponto em graus do tipo de dados GEOMETRY encontrados em uma esfera. A primeira coordenada do ponto é o valor da longitude. A segunda coordenada do ponto é o valor a latitude Para geometrias 3DZ, 3DM ou 4D, apenas as duas primeiras coordenadas são usadas.

geom2

Um valor de ponto em graus do tipo de dados GEOMETRY encontrados em uma esfera. A primeira coordenada do ponto é o valor da longitude. A segunda coordenada do ponto é o valor a latitude Para geometrias 3DZ, 3DM ou 4D, apenas as duas primeiras coordenadas são usadas.

radius

O raio de uma esfera de tipo de dados DOUBLE PRECISION. Se nenhum radius for fornecido, a esfera será padronizada como a Terra, e o raio será computado da representação do Sistema Geodésico Mundial (WGS) 84 do elipsoide.

Tipo de retorno

DOUBLE PRECISION nas mesmas unidades que o raio. Se nenhum raio for fornecido, a distância será em metros.

Nulo será retornado se geom1 ou geom2 for nulo ou vazio.

Se nenhum radius for fornecido, o resultado será em metros ao longo da superfície da Terra.

Um erro será retornado se radius for um número negativo.

Um erro será retornado se geom1 e geom2 não tiverem o mesmo valor do identificador do sistema referência espacial (SRID -spatial reference system identifier).

Um erro será retornado se geom1 ou geom2 não for um ponto.

Exemplos

O exemplo SQL a seguir calcula a distância em quilômetros entre dois pontos na Terra.

```
SELECT ROUND(ST_DistanceSphere(ST_Point(-122, 47), ST_Point(-122.1, 47.1))/ 1000, 0);
```

```
  round
-----------
 13
```

O seguinte SQL de exemplo calcula as distâncias em quilômetros entre três localizações de aeroportos na Alemanha: Berlin Tegel (TXL), Munich International (MUC) e Frankfurt International (FRA).

```
WITH airports_raw(code,lon,lat) AS (
(SELECT 'MUC', 11.786111, 48.353889) UNION
(SELECT 'FRA', 8.570556, 50.033333) UNION
(SELECT 'TXL', 13.287778, 52.559722)),
airports1(code,location) AS (SELECT code, ST_Point(lon, lat) FROM airports_raw),
airports2(code,location) AS (SELECT * from airports1)
SELECT (airports1.code || ' <-> ' || airports2.code) AS airports,
round(ST_DistanceSphere(airports1.location, airports2.location) / 1000, 0) AS
 distance_in_km
FROM airports1, airports2 WHERE airports1.code < airports2.code ORDER BY 1;
```

```
  airports   | distance_in_km
-------------+----------------
 FRA <-> MUC |            299
 FRA <-> TXL |            432
 MUC <-> TXL |            480
```

ST_DWithin

ST_DWithin retorna true se a distância euclidiana entre as projeções 2D dos dois valores de geometria de entrada não for maior que um valor limite.

Sintaxe

```
ST_DWithin(geom1, geom2, threshold)
```

Argumentos

geom1

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`.

geom2

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`.

threshold

Um valor de de tipo de dados `DOUBLE PRECISION`. Esse valor está nas unidades dos argumentos de entrada.

Tipo de retorno

`BOOLEAN`

Nulo será retornado se geom1 ou geom2 for nulo.

Um erro será retornado se threshold for negativo.

Um erro será retornado se geom1 e geom2 não tiverem o mesmo valor do identificador do sistema referência espacial (SRID -spatial reference system identifier).

Um erro será retornado se geom1 ou geom2 for uma coleção de geometrias.

Exemplos

O SQL a seguir verifica se a distância entre dois polígonos está dentro de cinco unidades.

```
SELECT ST_DWithin(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'),
 ST_GeomFromText('POLYGON((-1 3,2 1,0 -3,-1 3))'),5);
```

```
st_dwithin
-----------
 true
```

ST_EndPoint

ST_EndPoint retorna o último ponto de uma linestring de entrada. O valor do Spatial Reference System Identifier (SRID – Identificador do sistema de referência espacial) do resultado é o mesmo da geometria de entrada. A dimensão da geometria retornada é a mesma geometria de entrada.

Sintaxe

```
ST_EndPoint(geom)
```

Argumentos

geom

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`. O subtipo deve ser `LINESTRING`.

Tipo de retorno

`GEOMETRY`

Nulo será retornado se geom for nulo.

Nulo será retornado se geom estiver vazio.

Se geom não for uma `LINESTRING`, será retornado null.

Exemplos

O SQL a seguir retorna uma representação de Extended well-known text (EWKT – Texto bem-conhecido estendido) de uma `LINESTRING` de quatro pontos para um objeto `GEOMETRY` e retorna o ponto final da linestring.

```
SELECT ST_AsEWKT(ST_EndPoint(ST_GeomFromText('LINESTRING(0 0,10 0,10 10,5 5,0
 5)',4326)));
```

```
st_asewkt
-------------
 SRID=4326;POINT(0 5)
```

ST_Envelope

ST_Envelope retorna a caixa delimitadora mínima da geometria de entrada, da seguinte forma:

- Se a geometria de entrada for vazia, a geometria retornada será uma cópia da geometria de entrada.
- Se a caixa delimitadora mínima da geometria de entrada se degenerar para um ponto, a geometria retornada será um ponto.
- Se a caixa delimitadora mínima da geometria de entrada for unidimensional, será retornada uma linestring de dois pontos.
- Se nenhum dos anteriores for verdadeiro, a função retornará um polígono orientado no sentido horário, cujos vértices serão os cantos da caixa delimitadora mínima.

O Spatial Reference System Identifier (SRID – Identificador do sistema de referência espacial) da geometria retornada é o mesmo da geometria de entrada.

Para todas as entradas não vazias, a função opera na projeção 2D da geometria de entrada.

Sintaxe

```
ST_Envelope(geom)
```

Argumentos

geom

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`.

Tipo de retorno

GEOMETRY

Nulo será retornado se geom for nulo.

Exemplos

O SQL a seguir converte uma representação de Well-known text (WKT – Texto bem-conhecido) de uma LINESTRING de quatro pontos para um objeto GEOMETRY e retorna um polígono cujos vértices são a caixa delimitadora mínima.

```
SELECT ST_AsText(ST_Envelope(ST_GeomFromText('GEOMETRYCOLLECTION(POLYGON((0 0,10 0,0
 10,0 0)),LINESTRING(20 10,20 0,10 0))')));
```

```
    st_astext
-----------------------------------
  POLYGON((0 0,0 10,20 10,20 0,0 0))
```

ST_Equals

ST_Equals retorna true se as projeções 2D das geometrias de entrada forem geometricamente iguais. Geometrias são consideradas geometricamente iguais se tiverem conjuntos de pontos iguais e seus interiores tiverem uma interseção não vazia.

Sintaxe

```
ST_Equals(geom1, geom2)
```

Argumentos

geom1

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

geom2

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY. Esse valor é comparado com geom1 para determinar se ele é igual a geom1.

Tipo de retorno

BOOLEAN

Um erro será retornado se geom1 ou geom2 for nulo.

Um erro será retornado se geom1 e geom2 não tiverem o mesmo valor do identificador do sistema referência espacial (SRID -spatial reference system identifier).

Um erro será retornado se geom1 ou geom2 for uma coleção de geometrias.

Exemplos

O SQL a seguir verifica se os dois polígonos são geometricamente iguais

```
SELECT ST_Equals(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'),
 ST_GeomFromText('POLYGON((-1 3,2 1,0 -3,-1 3))'));
```

```
st_equals
-----------
 false
```

O SQL a seguir verifica se as duas linestrings são geometricamente iguais

```
SELECT ST_Equals(ST_GeomFromText('LINESTRING(1 0,10 0)'), ST_GeomFromText('LINESTRING(1
 0,5 0,10 0)'));
```

```
st_equals
-----------
 true
```

ST_ExteriorRing

ST_ExteriorRing retorna um linestring fechado que representa o anel externo de um polígono de entrada. A dimensão da geometria retornada é a mesma geometria de entrada.

Sintaxe

```
ST_ExteriorRing(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

GEOMETRY de subtipo LINESTRING.

O valor do identificador do sistema de referência espacial (SRID - spatial reference system identifier) da geometria retornada é o valor do SRID das geometrias de entrada.

Nulo será retornado se geom for nulo.

Nulo será retornado se geom não for um polígono.

Se geom estiver vazio, será retornado um polígono vazio.

Exemplos

O SQL a seguir retorna o anel externo de um polígono como uma linestring fechada.

```
SELECT ST_AsEWKT(ST_ExteriorRing(ST_GeomFromText('POLYGON((7 9,8 7,11 6,15 8,16 6,17
 7,17 10,18 12,17 14,15 15,11 15,10 13,9 12,7 9),(9 9,10 10,11 11,11 10,10 8,9 9),(12
 14,15 14,13 11,12 14))')));
```

```
st_asewkt
-----------
 LINESTRING(7 9,8 7,11 6,15 8,16 6,17 7,17 10,18 12,17 14,15 15,11 15,10 13,9 12,7 9)
```

ST_Force2D

ST_Force2D retorna uma geometria 2D da geometria de entrada. Para geometrias 2D, uma cópia da entrada é retornada. Para geometrias 3DZ, 3DM e 4D, ST_Force2D projeta a geometria para o plano

XY cartesiano. Pontos vazios na geometria de entrada permanecem pontos vazios na geometria de saída.

Sintaxe

```
ST_Force2D(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

GEOMETRY.

O valor do identificador do sistema de referência espacial (SRID - spatial reference system identifier) da geometria retornada é o valor do SRID das geometrias de entrada.

Nulo será retornado se geom for nulo.

Se geom estiver vazio, será retornado uma geometria vazia.

Exemplos

O SQL a seguir retorna uma geometria 2D de uma geometria 3DZ.

```
SELECT ST_AsEWKT(ST_Force2D(ST_GeomFromText('MULTIPOINT Z(0 1 2, EMPTY, 2 3 4, 5 6
 7)')));
```

```
st_asewkt
-----------
  MULTIPOINT((0 1),EMPTY,(2 3),(5 6))
```

ST_Force3D

ST_Force3D é um alias para ST_Force3DZ. Para obter mais informações, consulte ST_Force3DZ.

ST_Force3DM

ST_Force3DM retorna uma geometria 3DM da geometria de entrada. Para geometrias 2D, as coordenadas `m` dos pontos não vazios na geometria de saída são todas definidas como `0`. Para geometrias 3DM, uma cópia da geometria de entrada será retornado. Para geometrias 3DZ, a geometria é projetada para o plano XY cartesiano, e as coordenadas `m` dos pontos não vazios na geometria de saída são todas definidas como `0`. Para geometrias 4D, a geometria é projetada para o espaço XYM cartesiano. Pontos vazios na geometria de entrada permanecem pontos vazios na geometria de saída.

Sintaxe

```
ST_Force3DM(geom)
```

Argumentos

geom

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`.

Tipo de retorno

`GEOMETRY`.

O valor do identificador do sistema de referência espacial (SRID - spatial reference system identifier) da geometria retornada é o valor do SRID das geometrias de entrada.

Nulo será retornado se geom for nulo.

Se geom estiver vazio, será retornado uma geometria vazia.

Exemplos

O SQL a seguir retorna uma geometria 3DM de uma geometria 3DZ.

```
SELECT ST_AsEWKT(ST_Force3DM(ST_GeomFromText('MULTIPOINT Z(0 1 2, EMPTY, 2 3 4, 5 6
 7)')));
```

```
st_asewkt
-----------
  MULTIPOINT M ((0 1 0),EMPTY,(2 3 0),(5 6 0))
```

ST_Force3DZ

ST_Force3DZ retorna uma geometria 3DZ da geometria de entrada. Para geometrias 2D, as coordenadas z dos pontos não vazios na geometria de saída são todas definidas como `0`. Para geometrias 3DM, a geometria é projetada no plano XY cartesiano, e as coordenadas z dos pontos não vazios na geometria de saída são todas definidas como `0`. Para geometrias 3DZ, uma cópia da geometria de entrada será retornado. Para geometrias 4D, a geometria é projetada para o espaço XYZ-cartesiano. Pontos vazios na geometria de entrada permanecem pontos vazios na geometria de saída.

Sintaxe

```
ST_Force3DZ(geom)
```

Argumentos

geom

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`.

Tipo de retorno

`GEOMETRY`.

O valor do identificador do sistema de referência espacial (SRID - spatial reference system identifier) da geometria retornada é o valor do SRID das geometrias de entrada.

Nulo será retornado se geom for nulo.

Se geom estiver vazio, será retornado uma geometria vazia.

Exemplos

O SQL a seguir retorna uma geometria 3DZ de uma geometria 3DM.

```
SELECT ST_AsEWKT(ST_Force3DZ(ST_GeomFromText('MULTIPOINT M(0 1 2, EMPTY, 2 3 4, 5 6
 7)')));
```

```
st_asewkt
-----------
  MULTIPOINT Z ((0 1 0),EMPTY,(2 3 0),(5 6 0))
```

ST_Force4D

ST_Force4D retorna uma geometria 4D da geometria de entrada. Para geometrias 2D, as coordenadas `z` e `m` dos pontos não vazios na geometria de saída são todas definidas como `0`. Para geometrias 3DM, as coordenadas `z` dos pontos não vazios na geometria de saída são todas definidas como `0`. Para geometrias 3DZ, as coordenadas `m` dos pontos não vazios na geometria de saída são todas definidas como `0`. Para geometrias 4D, uma cópia da geometria de entrada será retornada. Pontos vazios na geometria de entrada permanecem pontos vazios na geometria de saída.

Sintaxe

```
ST_Force4D(geom)
```

Argumentos

geom

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`.

Tipo de retorno

`GEOMETRY`.

O valor do identificador do sistema de referência espacial (SRID - spatial reference system identifier) da geometria retornada é o valor do SRID das geometrias de entrada.

Nulo será retornado se geom for nulo.

Se geom estiver vazio, será retornado uma geometria vazia.

Exemplos

O SQL a seguir retorna uma geometria 4D de uma geometria 3DM.

```
SELECT ST_AsEWKT(ST_Force4D(ST_GeomFromText('MULTIPOINT M(0 1 2, EMPTY, 2 3 4, 5 6
 7)')));
```

```
st_asewkt
-----------
  MULTIPOINT ZM ((0 1 0 2),EMPTY,(2 3 0 4),(5 6 0 7))
```

ST_GeoHash

ST_GeoHash retorna a representação geohash do ponto inserido com a precisão especificada. O valor de precisão padrão é 20. Para obter mais informações sobre a definição de geohash, consulte Geohash na Wikipédia.

Sintaxe

```
ST_GeoHash(geom)
```

```
ST_GeoHash(geom, precision)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

precisão

Um valor de de tipo de dados INTEGER. O padrão é 20.

Tipo de retorno

GEOMETRY

A função retorna a representação geohash do ponto inserido.

Se o ponto inserido for vazio, a função retornará null.

Se a forma geométrica inserida não for um ponto, a função retornará um erro.

Exemplos

O comando SQL a seguir retorna a representação em geohash do ponto inserido.

```
SELECT ST_GeoHash(ST_GeomFromText('POINT(45 -45)'), 25) AS geohash;
```

```
         geohash
--------------------------
 m0000000000000000000gzz
```

O comando SQL a seguir retorna null porque o ponto inserido é vazio.

```
SELECT ST_GeoHash(ST_GeomFromText('POINT EMPTY'), 10) IS NULL AS result;
```

```
 result
---------
 true
```

ST_GeogFromText

ST_GeomFromText cria um objeto geography a partir da representação de texto reconhecido (WKT) ou texto reconhecido estendido (EWKT) de uma geometria de entrada.

Sintaxe

```
ST_GeogFromText(wkt_string)
```

Argumentos

wkt_string

Um valor de tipo de dados VARCHAR que é uma representação de WKT ow EWKT de uma geografia.

Tipo de retorno

GEOGRAPHY

Se o valor SRID estiver definido para o valor fornecido na entrada. Se o SRID não for fornecido, será definido como 4326.

Se ewkb_string for nulo, será retornado null.

Se wkt_string for inválido, um erro será retornado.

Exemplos

O SQL a seguir cria um polígono a partir de um objeto geography com um valor de SRID.

```
SELECT ST_AsEWKT(ST_GeogFromText('SRID=4324;POLYGON((0 0,0 1,1 1,10 10,1 0,0 0))'));
```

```
  st_asewkt
-----------------------------------------------
 SRID=4324;POLYGON((0 0,0 1,1 1,10 10,1 0,0 0))
```

O SQL a seguir cria um polígono a partir de um objeto geography. O valor de SRID é definido como 4326.

```
SELECT ST_AsEWKT(ST_GeogFromText('POLYGON((0 0,0 1,1 1,10 10,1 0,0 0))'));
```

```
 st_asewkt
-----------------------------------------------
 SRID=4326;POLYGON((0 0,0 1,1 1,10 10,1 0,0 0))
```

St_GeogFromWKB

ST_GeomFromEWKB cria um objeto geography da representação binária reconhecida (WKB) hexadecimal de uma geografia de entrada.

Sintaxe

```
ST_GeogFromWKB(wkb_string)
```

Argumentos

wkb_string

Um valor de tipo de dados VARCHAR que é uma representação de WKB hexadecimal de uma geografia.

Tipo de retorno

GEOGRAPHY

Se o valor SRID for fornecido, será definido como o valor fornecido. Se o SRID não for fornecido, será definido como 4326.

Será retornado null, se wkb_string for nulo.

Se wkb_string for inválido, um erro será retornado.

Exemplos

O SQL a seguir cria uma geografia com base em um valor de WKB hexadecimal.

```
SELECT
 ST_AsEWKT(ST_GeogFromWKB('010300000001000000050000000000000000000000000000000000000000000000000
```

```
 st_asewkt
-------------------------------------------
 SRID=4326;POLYGON((0 0,0 1,1 1,1 0,0 0))
```

ST_GeometryN

ST_GeometryN retorna uma geometria apontada pelo índice de entrada da geometria de entrada, da seguinte forma:

- Se a entrada for um ponto, uma linestring ou um polígono, será retornada uma geometria no estado em que se encontra se o índice for igual a um (1), e null se o índice for diferente de um (1).
- Se a entrada for multiponto, multilinestring, multipolígono ou coleção de geometrias, será retornado um ponto, uma linestring, um polígono ou uma coleção de geometrias, conforme apontado por um índice de entrada.

O índice é baseado em um. O Spatial Reference System Identifier (SRID – Identificador do sistema de referência espacial) do resultado é o mesmo da geometria de entrada. A dimensão da geometria retornada é a mesma geometria de entrada.

Sintaxe

```
ST_GeometryN(geom, index)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Índice do

Um valor do tipo de dados INTEGER que representa a posição de um índice baseado em um.

Tipo de retorno

GEOMETRY

Se geom ou index for null, será retornado null.

Se index estiver fora do intervalo, será retornado um erro.

Exemplos

O SQL a seguir retorna as geometrias em uma coleção de geometrias.

```
WITH tmp1(idx) AS (SELECT 1 UNION SELECT 2),
tmp2(g) AS (SELECT ST_GeomFromText('GEOMETRYCOLLECTION(POLYGON((0 0,10 0,0 10,0
 0)),LINESTRING(20 10,20 0,10 0))'))
SELECT idx, ST_AsEWKT(ST_GeometryN(g, idx)) FROM tmp1, tmp2 ORDER BY idx;
```

```
 idx |          st_asewkt
-----+-----------------------------
   1 | POLYGON((0 0,10 0,0 10,0 0))
   2 | LINESTRING(20 10,20 0,10 0)
```

ST_GeometryType

ST_GeometryType retorna o subtipo de uma geometria de entrada como uma string.

Para entradas de geometria 3DM, 3DZ e 4D, ST_GeometryType retorna o mesmo resultado que para entradas de geometria 2D.

Sintaxe

```
ST_GeometryType(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

VARCHAR que representa o subtipo de geom.

Nulo será retornado se geom for nulo.

Os valores retornados são os seguintes.

Valor da string retornada	Subtipo de geometria
ST_Point	Retornado se geom for um subtipo POINT
ST_LineString	Retornado se geom for um subtipo LINESTRING
ST_Polygon	Retornado se geom for um subtipo POLYGON
ST_MultiPoint	Retornado se geom for um subtipo MULTIPOINT
ST_MultiLineString	Retornado se geom for um subtipo MULTILINESTRING

Valor da string retornada	Subtipo de geometria
`ST_MultiPolygon`	Retornado se geom for um subtipo `MULTIPOLYGON`
`ST_GeometryCollection`	Retornado se geom for um subtipo `GEOMETRYCOLLECTION`

Exemplos

O SQL a seguir retorna o subtipo da geometria de linestring de entrada.

```
SELECT ST_GeometryType(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27
 29.31,77.29 29.07)'));
```

```
st_geometrytype
-------------
 ST_LineString
```

ST_GeomFromEWKB

ST_GeomFromEWKB cria um objeto geometry da representação de binário bem-conhecido estendido (EWKB - extended well-known binary) de uma geometria de entrada.

ST_GeomFromEWKB aceita geometrias 3DZ, 3DM e 4D escritas em formato hexadecimal WKB e EWKB.

Sintaxe

```
ST_GeomFromEWKB(ewkb_string)
```

Argumentos

ewkb_string

Um valor do tipo de dados `VARCHAR` ou `VARBYTE` que seja uma representação EWKB hexadecimal de uma geometria.

Tipo de retorno

GEOMETRY

Nulo será retornado se ewkb_string for nulo.

Se ewkb_string for inválido, um erro será retornado.

Exemplos

O SQL a seguir cria um polígono de um valor de EWKB e retorna a representação de EWKT de um polígono.

```
SELECT
 ST_AsEWKT(ST_GeomFromEWKB('0103000020E6100000010000000500000000000000000000000000000000000000000
```

```
 st_asewkt
-------------------------------
 SRID=4326;POLYGON((0 0,0 1,1 1,1 0,0 0))
```

ST_GeomFromEWKT

ST_GeomFromEWKB constrói um objeto de geometria a partir da representação de texto conhecido estendido (EWKT) de uma geometria de entrada.

ST_GeomFromEWKT aceita 3DZ, 3DM e 4D onde o tipo de geometria é prefixado com Z, M ou ZM, respectivamente.

Sintaxe

```
ST_GeomFromEWKT(ewkt_string)
```

Argumentos

ewkt_string

Um valor de tipo de dados VARCHAR ou uma expressão que avalia para um tipo VARCHAR, que é uma representação EWKT de uma geometria.

Você pode usar a palavra-chave WKT EMPTY para designar um ponto vazio, um multiponto com um ponto vazio ou uma coleção de geometria com um ponto vazio. O exemplo a seguir cria um ponto vazio.

```
ST_GeomFromEWKT('SRID=4326;POINT EMPTY');
```

Tipo de retorno

GEOMETRY

Se ewkb_string for nulo, então null é retornado.

Se ewkt_string for inválido, um erro será retornado.

Exemplos

O SQL a seguir cria uma multilinestring a partir de um valor EWKT e retorna uma geometria. Ele também retorna o resultado ST_AsEWKT da geometria.

```
SELECT ST_GeomFromEWKT('SRID=4326;MULTILINESTRING((1 0,1 0),(2 0,3 0),(4 0,5 0,6 0))')
 as geom, ST_AsEWKT(geom);
```

```
                                                    geom

                                        |                          st_asewkt

-----------------------------------------------------------------------------------------------
+-----------------------------------------------------------
 0105000020E6100000030000000102000000020000000000000000F03F0000000000000000000000000000F03F00
 | SRID=4326;MULTILINESTRING((1 0,1 0),(2 0,3 0),(4 0,5 0,6 0))
```

ST_GeomFromGeoHash

ST_GeomFromGeoHash constrói um objeto de geometria com base na representação geohash de uma geometria de entrada. ST_GeomFromGeoHash retorna uma geometria bidimensional (2D) com o identificador de referência espacial (SRID) de zero (0). Para obter mais informações sobre o formato geohash, consulte Geohash na Wikipedia.

Sintaxe

```
ST_GeomFromGeoHash(geohash_string)
```

```
ST_GeomFromGeoHash(geohash_string, precision)
```

Argumentos

geohash_string

Um valor de tipo de dado VARCHAR ou uma expressão que avalia para um tipo VARCHAR, que é uma representação geohash de uma geometria.

precisão

Um valor de tipo de dado INTEGER que representa a precisão de geohash. O valor é o número de caracteres do geohash a ser usado como precisão. Se o valor não for especificado, for menor que zero ou for maior que o comprimento de geohash_string, o comprimento de geohash_string será usado.

Tipo de retorno

GEOMETRY

Se geohash_string for nulo, será retornado null.

Se geohash_string for inválido, será retornado um erro.

Exemplos

O SQL a seguir retorna um polígono de alta precisão.

```
SELECT ST_AsText(ST_GeomFromGeoHash('9qqj7nmxncgyy4d0dbxqz0'));
```

```
 st_asewkt
-----------------------
 POLYGON((-115.172816 36.114646,-115.172816 36.114646,-115.172816 36.114646,-115.172816
 36.114646,-115.172816 36.114646))
```

O SQL a seguir retorna um ponto de alta precisão.

```
SELECT ST_AsText(ST_GeomFromGeoHash('9qqj7nmxncgyy4d0dbxqz00'));
```

```
 st_asewkt
-----------------------
 POINT(-115.172816 36.114646)
```

O SQL a seguir retorna um polígono de baixa precisão.

```
SELECT ST_AsText(ST_GeomFromGeoHash('9qq'));
```

```
 st_asewkt
-----------------------
 POLYGON((-115.3125 35.15625,-115.3125 36.5625,-113.90625 36.5625,-113.90625
 35.15625,-115.3125 35.15625))
```

O SQL a seguir retorna um polígono de precisão 3.

```
SELECT ST_AsText(ST_GeomFromGeoHash('9qqj7nmxncgyy4d0dbxqz0', 3));
```

```
 st_asewkt
-----------------------
 POLYGON((-115.3125 35.15625,-115.3125 36.5625,-113.90625 36.5625,-113.90625
 35.15625,-115.3125 35.15625))
```

ST_GeomFromGeoJSON

ST_GeomFromGeoJSON constrói um objeto de geometria com base na representação GeoJSON de uma geometria de entrada. Para obter mais informações sobre o formato GeoJSON, consulte GeoJSON na Wikipédia.

Se houver pelo menos um ponto com três ou mais coordenadas, a geometria resultante será 3DZ, onde o componente Z é zero para os pontos que têm apenas duas coordenadas. Se

todos os pontos no GeoJSON de entrada contiverem duas coordenadas ou estiverem vazios, ST_GeomFromGeoJSON retornará uma geometria 2D. A geometria retornada sempre tem o identificador do sistema de referência espacial (SRID) definido como 4326.

Sintaxe

```
ST_GeomFromGeoJSON(geojson_string)
```

Argumentos

geojson_string

Um valor de tipo de dados VARCHAR ou SUPER ou uma expressão que determina o valor de um tipo VARCHAR, que é uma representação GeoJSON de uma geometria.

Tipo de retorno

GEOMETRY

Se geojson_string for nulo, será retornado null.

Se geojson_string for inválido, será retornado um erro.

Exemplos

O SQL a seguir retorna uma geometria 2D representada no GeoJSON de entrada.

```
SELECT ST_AsEWKT(ST_GeomFromGeoJSON('{"type":"Point","coordinates":[1,2]}'));
```

```
 st_asewkt
----------------------
 SRID=4326;POINT(1 2)
```

O SQL a seguir retorna uma geometria 3DZ representada no GeoJSON de entrada.

```
SELECT ST_AsEWKT(ST_GeomFromGeoJSON('{"type":"LineString","coordinates":[[1,2,3],
[4,5,6],[7,8,9]]}'));
```

```
 st_asewkt
-----------------------------------------
 SRID=4326;LINESTRING Z (1 2 3,4 5 6,7 8 9)
```

O SQL a seguir retorna a geometria 3DZ quando apenas um ponto tem três coordenadas, enquanto todos os outros pontos têm duas coordenadas no GeoJSON de entrada.

```
SELECT ST_AsEWKT(ST_GeomFromGeoJSON('{"type":"Polygon","coordinates":[[[0, 0],[0, 1,
 8],[1, 0],[0, 0]]]}'));
```

```
 st_asewkt
-----------------------------------------------
 SRID=4326;POLYGON Z ((0 0 0,0 1 8,1 0 0,0 0 0))
```

ST_GeomFromGeoSquare

ST_GeomFromGeoSquare retorna uma geometria que cobre a área representada por um valor de quadrado geográfico de entrada. A geometria retornada é sempre bidimensional. Para calcular um valor de quadrado geográfico, consulte ST_GeoSquare.

Sintaxe

```
ST_GeomFromGeoSquare(geosquare)
```

```
ST_GeomFromGeoSquare(geosquare, max_depth)
```

Argumentos

geosquare

Um valor do tipo de dados `BIGINT` ou uma expressão avaliada como um tipo `BIGINT` que é um valor de quadrado geográfico e descreve a sequência de subdivisões feitas no domínio inicial para alcançar o quadrado desejado. Esse valor é calculado por ST_GeoSquare.

max_depth

Um valor do tipo de dados `INTEGER` que representa o número máximo de subdivisões de domínio feitas no domínio inicial. O valor deve ser igual ou maior que 1.

Tipo de retorno

GEOMETRY

Se o geosquare não for válido, a função retornará um erro.

Se a entrada max_depth não estiver dentro do intervalo, a função retornará um erro.

Exemplos

O SQL a seguir retorna uma geometria de um valor de quadrado geográfico.

```
SELECT ST_AsText(ST_GeomFromGeoSquare(797852));
```

```
 st_astext
-----------------------------------------------------------------------------------------------
 POLYGON((13.359375 52.3828125,13.359375 52.734375,13.7109375 52.734375,13.7109375
 52.3828125,13.359375 52.3828125))
```

O SQL a seguir retorna uma geometria de um valor de quadrado geográfico e uma profundidade máxima de 3.

```
SELECT ST_AsText(ST_GeomFromGeoSquare(797852, 3));
```

```
 st_astext
-------------------------------------
 POLYGON((0 45,0 90,45 90,45 45,0 45))
```

O SQL a seguir primeiro calcula o valor de quadrado geográfico para Seattle especificando a coordenada x como longitude e a coordenada y como latitude (-122.3, 47.6). Depois, ele retorna o polígono para o quadrado geográfico. Embora a saída seja uma geometria bidimensional, ela pode ser usada para calcular dados espaciais em termos de longitude e latitude.

```
SELECT ST_AsText(ST_GeomFromGeoSquare(ST_GeoSquare(ST_Point(-122.3, 47.6))));
```

```
 st_astext
-----------------------------------------------------------------------------------------------
```

```
POLYGON((-122.335167014971 47.6080129947513,-122.335167014971
 47.6080130785704,-122.335166931152 47.6080130785704,-122.335166931152
 47.6080129947513,-122.335167014971 47.6080129947513))
```

ST_GeomFromText

ST_GeomFromText cria um objeto geometry da representação de texto bem-conhecido (WKT - well-known text) de uma geometria de entrada.

ST_GeomFromText aceita 3DZ, 3DM e 4D onde o tipo de geometria é prefixado com Z, M ou ZM, respectivamente.

Sintaxe

```
ST_GeomFromText(wkt_string)
```

```
ST_GeomFromText(wkt_string, srid)
```

Argumentos

wkt_string

Um valor de tipo de dados VARCHAR que é uma representação de WKT de uma geometria.

Você pode usar a palavra-chave WKT EMPTY para designar um ponto vazio, um multiponto com um ponto vazio ou uma coleção de geometria com um ponto vazio. O exemplo a seguir cria um multiponto com um ponto vazio e um ponto não vazio.

```
ST_GeomFromEWKT('MULTIPOINT(1 0,EMPTY)');
```

srid

Um valor de tipo de dados INTEGER que é um identificador de referência espacial (SRID - spatial reference identifier), Se o valor de um SRID for fornecido, a geometria retornada terá esse valor de SRID. Caso contrário, o valor do SRID da geometria retornada será definido como zero (0).

Tipo de retorno

GEOMETRY

Nulo será retornado se wkt_string ou srid for nulo.

Se srid for negativo, será retornado nulo.

Se wkt_string for inválido, um erro será retornado.

Se srid for inválido, um erro será retornado.

Exemplos

O SQL a seguir cria um objeto geometria da representação de WKT e do valor de SRID.

```
SELECT ST_GeomFromText('POLYGON((0 0,0 1,1 1,1 0,0 0))',4326);
```

```
st_geomfromtext
--------------------------------
0103000020E6100000010000000500000000000000000000000000000000000000000000000000F03F000000000000F03F000000000000F03F000000000000F03F0000000000000000000000000000000000
```

ST_GeomFromWKB

ST_GeomFromEWKB cria um objeto geometry da representação de binário bem-conhecido (WKB - well-known binary) hexadecimal de uma geometria de entrada.

ST_GeomFromWKB aceita geometrias 3DZ, 3DM e 4D escritas em formato hexadecimal WKB.

Sintaxe

```
ST_GeomFromWKB(wkb_string)
```

```
ST_GeomFromWKB(wkb_string, srid)
```

Argumentos

wkb_string

Um valor do tipo de dados VARCHAR ou VARBYTE que seja uma representação WKB hexadecimal de uma geometria.

srid

Um valor de tipo de dados `INTEGER` que é um identificador de referência espacial (SRID - spatial reference identifier), Se o valor de um SRID for fornecido, a geometria retornada terá esse valor de SRID. Caso contrário, o valor do SRID da geometria retornado será definido como 0.

Tipo de retorno

`GEOMETRY`

Nulo será retornado se wkb_string ou srid for nulo.

Se srid for negativo, será retornado nulo.

Se wkb_string for inválido, um erro será retornado.

Se srid for inválido, um erro será retornado.

Exemplos

O SQL a seguir cria um polígono de um valor de WKB e retorna a representação de WKT de um polígono.

```
SELECT
 ST_AsText(ST_GeomFromWKB('010300000001000000050000000000000000000000000000000000000000000000
```

```
 st_astext
-------------------------------
 POLYGON((0 0,0 1,1 1,1 0,0 0))
```

ST_GeoSquare

ST_GeoSquare subdivide recursivamente o domínio ([-180, 180], [-90, 90]) em regiões quadradas iguais chamadas quadrados geográficos até uma profundidade especificada. A subdivisão é baseada na localização de um ponto fornecido. Um dos quadrados geográficos contendo o ponto é subdividido em cada etapa até atingir a profundidade máxima. A seleção desse quadrado geográfico é estável, ou seja, o resultado da função depende somente dos argumentos de entrada. A função retorna um valor exclusivo que identifica o quadrado geográfico final no qual o ponto está localizado.

O ST_GeoSquare aceita um PONTO em que a coordenada x representa a longitude e a coordenada y representa a latitude. A longitude e a latitude são limitadas a [-180, 180] e [-90, 90], respectivamente. A saída de ST_GeoSquare pode ser usada como entrada para a função ST_GeomFromGeoSquare.

Há 360° ao redor do arco da circunferência equatorial da Terra que são divididos em dois hemisférios (leste e oeste), cada um com 180° de linhas longitudinais (meridianos) a partir do meridiano 0°. Por convenção, as longitudes orientais são coordenadas “+” (positivas) quando projetadas em um eixo x em um plano cartesiano e as longitudes ocidentais são coordenadas “-” (negativas) quando projetadas em um eixo x em um plano cartesiano. Há 90° de linhas latitudinais ao norte e ao sul da circunferência equatorial de 0° da Terra, cada uma paralela à circunferência equatorial de 0° da Terra. Por convenção, as linhas latitudinais do norte cruzam o eixo y “+” (positivo) quando projetadas em um plano cartesiano, e as linhas latitudinais do sul cruzam o eixo y “-” (negativo) quando projetadas em um plano cartesiano. A grade esférica formada pela interseção de linhas longitudinais e linhas latitudinais é convertida em uma grade projetada em um plano cartesiano com coordenadas x positivas e negativas padrão e coordenadas y positivas e negativas no plano cartesiano.

O objetivo do ST_GeoSquare é marcar pontos próximos com valores de código iguais. Os pontos localizados no mesmo quadrado geográfico recebem o mesmo valor de código. Um quadrado geográfico é usado para codificar coordenadas geográficas (latitude e longitude) em um número inteiro. Uma região maior é dividida em grades para delinear uma área em um mapa com diferentes resoluções. Um quadrado geográfico pode ser usado para indexação espacial, agrupamento espacial, pesquisas de proximidade, pesquisa de localização e criação de identificadores de locais exclusivos. A função ST_GeoHash segue um processo similar de divisão de uma região em grades, mas tem uma codificação diferente.

Sintaxe

```
ST_GeoSquare(geom)
```

```
ST_GeoSquare(geom, max_depth)
```

Argumentos

geom

Um valor POINT de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um subtipo POINT. A coordenada x (longitude) do ponto deve estar dentro do intervalo de `-180` a `180`. A coordenada y (latitude) do ponto deve estar dentro do intervalo de `-90` a`90`.

max_depth

Um valor de de tipo de dados `INTEGER`. O número máximo de vezes que o domínio contendo o ponto é subdividido recursivamente. O valor deve ser um número inteiro de 1 a 32. O padrão é 32. O número final real das subdivisões é menor ou igual ao max_depth especificado.

Tipo de retorno

`BIGINT`

A função retorna um valor exclusivo que identifica o quadrado geográfico final no qual o ponto de entrada está localizado.

Se a entrada geom não for um ponto, a função retornará um erro.

Se o ponto de entrada estiver vazio, o valor de retorno não será uma entrada válida para a função ST_GeomFromGeoSquare. Use a função ST_IsEmpty para evitar chamadas para ST_GeoSquare com um ponto vazio.

Se o ponto de entrada não estiver no intervalo, a função retornará um erro.

Se a entrada max_depth estiver fora do intervalo, a função retornará um erro.

Exemplos

O SQL a seguir retorna um quadrado geográfico de um ponto de entrada.

```
SELECT ST_GeoSquare(ST_Point(13.5, 52.5));
```

```
  st_geosquare
-----------------------
 -4410772491521635895
```

O SQL a seguir retorna um quadrado geográfico de um ponto de entrada com uma profundidade máxima de 10.

```
SELECT ST_GeoSquare(ST_Point(13.5, 52.5), 10);
```

```
 st_geosquare
--------------
 797852
```

ST_InteriorRingN

ST_InteriorRingN retorna um linestring fechado correspondente ao anel interior de um polígono de entrada na posição do índice. A dimensão da geometria retornada é a mesma geometria de entrada.

Sintaxe

```
ST_InteriorRingN(geom, index)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Índice do

Um valor do tipo de dados INTEGER que representa a posição de um anel de um índice baseado em um.

Tipo de retorno

GEOMETRY de subtipo LINESTRING.

O valor do identificador do sistema de referência espacial (SRID - spatial reference system identifier) da geometria retornada é o valor do SRID das geometrias de entrada.

Se geom ou index for null, será retornado null.

Se index estiver fora do intervalo, será retornado null.

Nulo será retornado se geom não for um polígono.

Se geom for um polígono vazio, então null será retornado.

Exemplos

O SQL a seguir retorna o segundo anel do polígono como uma linestring fechada.

```
SELECT ST_AsEWKT(ST_InteriorRingN(ST_GeomFromText('POLYGON((7 9,8 7,11 6,15 8,16 6,17
 7,17 10,18 12,17 14,15 15,11 15,10 13,9 12,7 9),(9 9,10 10,11 11,11 10,10 8,9 9),(12
 14,15 14,13 11,12 14))'),2));
```

```
st_asewkt
-----------
 LINESTRING(12 14,15 14,13 11,12 14)
```

ST_Intersects

ST_Intersects retorna true se as projeções 2D das duas geometrias de entrada tiverem pelo menos um ponto em comum.

Sintaxe

```
ST_Intersects(geom1, geom2)
```

Argumentos

geom1

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

geom2

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

BOOLEAN

Nulo será retornado se geom1 ou geom2 for nulo.

Um erro será retornado se geom1 e geom2 não tiverem o mesmo valor do identificador do sistema referência espacial (SRID -spatial reference system identifier).

Um erro será retornado se geom1 ou geom2 for uma coleção de geometrias.

Exemplos

O SQL a seguir verifica se o primeiro polígono intercepta o segundo polígono.

```
SELECT ST_Intersects(ST_GeomFromText('POLYGON((0 0,10 0,10 10,0 10,0 0),(2 2,2 5,5 5,5
 2,2 2))'), ST_GeomFromText('MULTIPOINT((4 4),(6 6))'));
```

```
st_intersects
-------------
 true
```

ST_Intersection

ST_Intersection retorna uma geometria que representa a interseção do conjunto de pontos de duas geometrias. Ou seja, retorna a parte das duas geometrias de entrada que é compartilhada entre elas.

Sintaxe

```
ST_Intersection(geom1, geom2)
```

Argumentos

geom1

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

geom2

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

GEOMETRY

Se geom1 e geom2 não compartilhar nenhum espaço (não contíguos), será retornada uma geometria vazia.

Se geom1 e geom2 estiverem vazios, será retornada uma geometria vazia.

Um erro será retornado se geom1 e geom2 não tiverem o mesmo valor do identificador do sistema referência espacial (SRID -spatial reference system identifier).

Um erro será retornado se geom1 ou geom2 for uma coleção de geometrias.

Se geom1 ou geom2 não forem uma geometria bidimensional (2D), será retornado um erro.

Exemplos

O SQL a seguir retorna a geometria não vazia que representa a interseção de duas geometrias de entrada.

```
SELECT ST_AsEWKT(ST_Intersection(ST_GeomFromText('polygon((0 0,100 100,0 200,0 0))'),
 ST_GeomFromText('polygon((0 0,10 0,0 10,0 0))')));
```

```
        st_asewkt
-------------------------
 POLYGON((0 0,0 10,5 5,0 0))
```

O SQL a seguir retorna uma geometria vazia quando passam geometrias de entrada não contíguas (sem interseção).

```
SELECT ST_AsEWKT(ST_Intersection(ST_GeomFromText('linestring(0 100,0 0)'),
 ST_GeomFromText('polygon((1 0,10 0,1 10,1 0))')));
```

```
    st_asewkt
------------------
 LINESTRING EMPTY
```

ST_IsPolygonCCW

ST_IsPolygonCCW retorna true se a projeção 2D do polígono de entrada ou multipolígono estiver no sentido anti-horário. Se a geometria de entrada for um ponto, linestring, multiponto ou multilinestring, então true é retornado. Para coleções de geometrias, ST_IsPolygonCCW retornará true se todas as geometrias presentes na coleção forem anti-horário.

Sintaxe

```
ST_IsPolygonCCW(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

BOOLEAN

Nulo será retornado se geom for nulo.

Exemplos

O SQL a seguir verifica se o polígono está no sentido anti-horário.

```
SELECT ST_IsPolygonCCW(ST_GeomFromText('POLYGON((7 9,8 7,11 6,15 8,16 6,17 7,17 10,18
 12,17 14,15 15,11 15,10 13,9 12,7 9),(9 9,10 10,11 11,11 10,10 8,9 9),(12 14,15 14,13
 11,12 14))'));
```

```
 st_ispolygonccw
----------
 true
```

ST_IsPolygonCW

ST_IsPolygonCW retorna true se a projeção 2D do polígono de entrada ou multipolígono estiver no sentido horário. Se a geometria de entrada for um ponto, linestring, multiponto ou multilinestring,

então true é retornado. Para coleções de geometrias, ST_IsPolygonCW retornará true se todas as geometrias presentes na coleção estiverem no sentido horário.

Sintaxe

```
ST_IsPolygonCW(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

BOOLEAN

Nulo será retornado se geom for nulo.

Exemplos

O SQL a seguir verifica se o polígono está no sentido horário.

```
SELECT ST_IsPolygonCW(ST_GeomFromText('POLYGON((7 9,8 7,11 6,15 8,16 6,17 7,17 10,18
 12,17 14,15 15,11 15,10 13,9 12,7 9),(9 9,10 10,11 11,11 10,10 8,9 9),(12 14,15 14,13
 11,12 14))'));
```

```
 st_ispolygonccw
----------
 true
```

ST_IsClosed

ST_IsClosed retorna true se a projeção 2D da geometria de entrada for fechada. As regras a seguir definem uma geometria fechada:

- A geometria de entrada é um ponto ou vários pontos.
- A geometria de entrada é uma linestring e os pontos de início e de término da linestring coincidem.
- A geometria de entrada é uma multilinestring não vazia e todas as suas linestrings são fechadas.

- A geometria de entrada é um polígono não vazio, todos os anéis do polígono não estão vazios, e os pontos de início e de término coincidem.
- A geometria de entrada é um multipolígono não vazio e todos os seus polígonos são fechados.
- A geometria de entrada é uma coleção de geometrias não vazias e todos os seus componentes são fechados.

Sintaxe

```
ST_IsClosed(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

BOOLEAN

Se geom é um ponto vazio, então false é retornado.

Nulo será retornado se geom for nulo.

Exemplos

O SQL a seguir verifica se o polígono é fechado.

```
SELECT ST_IsClosed(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'));
```

```
st_isclosed
-----------
 true
```

ST_IsCollection

ST_IsCollection retornará true se a geometria de entrada for um dos seguintes subtipos: GEOMETRYCOLLECTION, MULTIPOINT, MULTILINESTRING ou MULTIPOLYGON.

Sintaxe

```
ST_IsCollection(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

BOOLEAN

Nulo será retornado se geom for nulo.

Exemplos

O SQL a seguir verifica se o polígono é uma coleção.

```
SELECT ST_IsCollection(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'));
```

```
st_iscollection
-----------
 false
```

ST_IsEmpty

ST_IsEmpty retornará true se a geometria de entrada estiver vazia. Uma geometria não estará vazia se contiver pelo menos um ponto não vazio.

ST_IsEmpty retornará true se a geometria de entrada tiver pelo menos um ponto não vazio.

Sintaxe

```
ST_IsEmpty(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

BOOLEAN

Nulo será retornado se geom for nulo.

Exemplos

O SQL a seguir verifica se o polígono especificado está vazio.

```
SELECT ST_IsEmpty(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'));
```

```
st_isempty
-----------
 false
```

ST_IsRing

ST_IsRing retornará true se a linestring de entrada for um anel. A linestring é um anel se for fechada e simples.

Sintaxe

```
ST_IsRing(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY. A geometria deve ser uma LINESTRING.

Tipo de retorno

BOOLEAN

Se geom não for uma LINESTRING, será retornado um erro.

Exemplos

O SQL a seguir verifica se a linestring especificada é um anel.

```
SELECT ST_IsRing(ST_GeomFromText('linestring(0 0, 1 1, 1 2, 0 0)'));
```

```
st_isring
-----------
 true
```

ST_IsSimple

ST_IsSimple retornará true se a projeção 2D da geometria de entrada for simples. Para obter mais informações sobre a definição de uma geometria simples, consulte Simplicidade geométrica.

Sintaxe

```
ST_IsSimple(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

BOOLEAN

Nulo será retornado se geom for nulo.

Exemplos

O SQL a seguir verifica se a linestring especificada é simples. Neste exemplo, não é simples porque tem auto-intersecção.

```
SELECT ST_IsSimple(ST_GeomFromText('LINESTRING(0 0,10 0,5 5,5 -5)'));
```

```
 st_issimple
-----------
 false
```

ST_IsValid

ST_IsValid retornará true se a projeção 2D da geometria de entrada for válida. Para obter mais informações sobre a definição de uma geometria válida, consulte Validade geométrica.

Sintaxe

```
ST_IsValid(geom)
```

Argumentos

geom

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`.

Tipo de retorno

`BOOLEAN`

Nulo será retornado se geom for nulo.

Exemplos

O SQL a seguir verifica se o polígono especificado é válido. Neste exemplo, o polígono é inválido porque o interior do polígono não está simplesmente conectado.

```
SELECT ST_IsValid(ST_GeomFromText('POLYGON((0 0,10 0,10 10,0 10,0 0),(5 0,10 5,5 10,0
 5,5 0))'));
```

```
 st_isvalid
-----------
 false
```

ST_Length

Para uma geometria linear, ST_Length retorna o comprimento cartesiano de uma projeção 2D. As unidades de comprimento são as mesmas unidades em que as coordenadas da geometria de entrada são expressas. A função retorna zero (0) para pontos, multipontos e geometrias de área. Quando a entrada for uma coleção de geometrias, a função retornará a soma dos comprimentos das geometrias na coleção.

Para uma geografia, o ST_Length retorna o comprimento geodésico da projeção 2D de uma geografia linear de entrada computada no esferoide determinado pelo SRID. A unidade de comprimento é o metro. A função retorna zero (0) para pontos, multipontos e geografias planas. Quando a entrada for uma coleção de geometrias, a função retornará a soma dos comprimentos das geografias na coleção.

Sintaxe

```
ST_Length(geo)
```

Argumentos

geo

Um valor de tipo de dados `GEOMETRY` ou `GEOGRAPHY` ou uma expressão que é avaliada como tipo `GEOMETRY` ou `GEOGRAPHY`.

Tipo de retorno

`DOUBLE PRECISION`

Será retornado null, se geo for nulo.

Se o valor de SRID não for encontrado, será retornado um erro.

Exemplos

O SQL a seguir retorna o comprimento cartesiano de uma multilinestring.

```
SELECT ST_Length(ST_GeomFromText('MULTILINESTRING((0 0,10 0,0 10),(10 0,20 0,20
 10))'));
```

```
st_length
--------------------------------
  44.142135623731
```

O SQL a seguir retorna o comprimento de uma linestring em uma geografia.

```
SELECT ST_Length(ST_GeogFromText('SRID=4326;LINESTRING(5 0,6 0,4 0)'));
```

```
 st_length
------------------
 333958.472379804
```

O SQL a seguir retorna o comprimento de um ponto em uma geografia.

```
SELECT ST_Length(ST_GeogFromText('SRID=4326;POINT(4 5)'));
```

```
 st_length
-----------
 0
```

ST_LengthSphere

ST_LengthSphere retorna o comprimento de uma geometria linear em metros. Para geometrias de ponto, multiponto e área, ST_LengthSphere retorna 0. Para coleções de geometria, ST_LengthSphere retorna o comprimento total das geometrias lineares na coleção em metros.

ST_LengthSphere interpreta as coordenadas de cada ponto da geometria de entrada como longitude e latitude em graus. Para geometrias 3DZ, 3DM ou 4D, apenas as duas primeiras coordenadas são usadas.

Sintaxe

```
ST_LengthSphere(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

DOUBLE PRECISIONComprimento em metros. O cálculo do comprimento é baseado no modelo esférico da Terra cujo raio é o raio médio da Terra do modelo elipsoidal do Sistema Geodésico Mundial (WGS) 84 da Terra.

Nulo será retornado se geom for nulo.

Exemplos

O exemplo SQL a seguir calcula o comprimento de uma linestring em metros.

```
SELECT ST_LengthSphere(ST_GeomFromText('LINESTRING(10 10,45 45)'));
```

```
 st_lengthsphere
------------------
 5127736.08292556
```

ST_Length2D

ST_Length2D é um alias para ST_Length. Para obter mais informações, consulte ST_Length.

ST_LineFromMultiPoint

ST_LineFromMultiPoint retorna uma linestring de uma geometria multiponto de entrada. A ordem dos pontos é preservada. O Spatial Reference System Identifier (SRID – Identificador do sistema de referência espacial) da geometria retornada é o mesmo da geometria de entrada. A dimensão da geometria retornada é a mesma geometria de entrada.

Sintaxe

```
ST_LineFromMultiPoint(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY. O subtipo deve ser MULTIPOINT.

Tipo de retorno

GEOMETRY

Nulo será retornado se geom for nulo.

Se geom estiver vazio, uma LINESTRING vazia será retornada.

Se geom contém pontos vazios, então esses pontos vazios são ignorados.

Se geom não for um MULTIPOINT, será retornado error.

Exemplos

O SQL a seguir cria uma linestring de um multiponto.

```
SELECT ST_AsEWKT(ST_LineFromMultiPoint(ST_GeomFromText('MULTIPOINT(0 0,10 0,10 10,5 5,0
 5)',4326)));
```

```
 st_asewkt
-------------------------------------------
 SRID=4326;LINESTRING(0 0,10 0,10 10,5 5,0 5)
```

ST_LineInterpolatePoint

ST_LineInterpolatePoint retorna um ponto ao longo de uma linha a uma distância fracionada do início da linha.

Para determinar a igualdade de pontos, ST_LineInterpolatePoint opera na projeção 2D da geometria de entrada. Se a geometria de entrada estiver vazia, será retornado uma cópia na mesma dimensão da entrada. Para geometrias 3DZ, 3DM e 4D, a coordenada z ou m é a média das coordenadas z ou m do segmento onde se encontra o ponto.

Sintaxe

```
ST_LineInterpolatePoint(geom, fraction)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY. O subtipo é LINESTRING.

fraction

Um valor de tipo de dados DOUBLE PRECISION que representa a posição de um ponto ao longo da linestring para a linha. O valor é uma fração no intervalo 0-1, inclusivo.

Tipo de retorno

GEOMETRY de subtipo POINT.

Se geom ou fraction for nulo, null será retornado.

Se geom estiver vazio, o ponto vazio será retornado.

O valor do identificador do sistema de referência espacial (SRID - spatial reference system identifier) da geometria retornada é o valor do SRID das geometrias de entrada.

Se fraction estiver fora do intervalo, será retornado um erro.

Se geom não for uma linestring, será retornado um erro.

Exemplos

O SQL a seguir retorna um ponto a meio caminho de uma linestring.

```
SELECT ST_AsEWKT(ST_LineInterpolatePoint(ST_GeomFromText('LINESTRING(0 0, 5 5, 7 7, 10
 10)'), 0.50));
```

```
st_asewkt
-----------
```

```
 POINT(5 5)
```

O SQL a seguir retorna um ponto 90% do caminho ao longo de uma linestring.

```
SELECT ST_AsEWKT(ST_LineInterpolatePoint(ST_GeomFromText('LINESTRING(0 0, 5 5, 7 7, 10
 10)'), 0.90));
```

```
st_asewkt
-----------
 POINT(9 9)
```

ST_M

ST_X retorna a coordenada `m` de um ponto de entrada.

Sintaxe

```
ST_M(point)
```

Argumentos

point

Um valor de `POINT` de tipo de dados `GEOMETRY`.

Tipo de retorno

O valor de `DOUBLE PRECISION` da coordenada `m`.

Nulo será retornado se point for nulo.

Se point for um ponto 2D ou 3DZ, null será retornado.

Se point for um ponto vazio, null será retornado.

Se point não for um `POINT`, um erro será retornado.

Exemplos

O SQL a seguir retorna a coordenada `m` de um ponto em uma geometria 3DM.

```
SELECT ST_M(ST_GeomFromEWKT('POINT M (1 2 3)'));
```

```
st_m
-----------
 3
```

O SQL a seguir retorna a coordenada `m` de um ponto em uma geometria 4D.

```
SELECT ST_M(ST_GeomFromEWKT('POINT ZM (1 2 3 4)'));
```

```
st_m
-----------
 4
```

ST_MakeEnvelope

ST_MakeEnvelope retorna uma geometria da seguinte forma:

- Se as coordenadas de entrada especificarem um ponto, a geometria retornada será um ponto.
- Se as coordenadas de entrada especificarem uma linha, então a geometria retornada é uma linestring.
- Caso contrário, a geometria retornada é um polígono, onde as coordenadas de entrada especificam os cantos inferior esquerdo e superior direito de uma caixa.

Se fornecido, o valor do identificador do sistema de referência espacial (SRID - spatial reference system identifier) da geometria retornada é configurado para o valor SRID de entrada.

Sintaxe

```
ST_MakeEnvelope(xmin, ymin, xmax, ymax)
```

```
ST_MakeEnvelope(xmin, ymin, xmax, ymax, srid)
```

Argumentos

xmin

Um valor de de tipo de dados DOUBLE PRECISION. Este valor é a primeira coordenada do canto inferior esquerdo de uma caixa.

ymin

Um valor de de tipo de dados DOUBLE PRECISION. Este valor é a segunda coordenada do canto inferior esquerdo de uma caixa.

xmax

Um valor de de tipo de dados DOUBLE PRECISION. Este valor é a primeira coordenada do canto superior direito de uma caixa.

ymax

Um valor de de tipo de dados DOUBLE PRECISION. Este valor é a segunda coordenada do canto superior direito de uma caixa.

srid

Um valor de tipo de dados INTEGER que é um identificador do sistema de referência espacial (SRID - spatial reference system identifier). Se o valor do SRID não for fornecido, ele será definido como zero.

Tipo de retorno

GEOMETRY de subtipo POINT, LINESTRING, ou POLYGON.

O SRID da geometria retornado é definido como srid ou zero sesrid não está definido.

Se xmin, ymin, xmax, ymax ou srid forem nulos, será exibido null.

Um erro será retornado se srid for negativo.

Exemplos

O SQL a seguir retorna um polígono representando um envelope definido pelos quatro valores de coordenadas de entrada.

```
SELECT ST_AsEWKT(ST_MakeEnvelope(2,4,5,7));
```

```
 st_astext
---------------
 POLYGON((2 4,2 7,5 7,5 4,2 4))
```

O SQL a seguir retorna um polígono que representa um envelope definido pelos quatro valores de coordenadas de entrada e um valor SRID.

```
SELECT ST_AsEWKT(ST_MakeEnvelope(2,4,5,7,4326));
```

```
 st_astext
---------------------------------
 SRID=4326;POLYGON((2 4,2 7,5 7,5 4,2 4))
```

ST_MakeLine

ST_MakeLine cria uma linestring das geometrias de entrada.

A dimensão da geometria retornada é a mesma das geometrias de entrada. Ambas as geometrias de entrada devem ter a mesma dimensão.

Sintaxe

```
ST_MakeLine(geom1, geom2)
```

Argumentos

geom1

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`. O subtipo deve ser `POINT`, `LINESTRING` ou `MULTIPOINT`.

geom2

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`. O subtipo deve ser `POINT`, `LINESTRING` ou `MULTIPOINT`.

Tipo de retorno

`GEOMETRY` de subtipo `LINESTRING`.

Nulo será retornado se geom1 ou geom2 for nulo.

Se geom1 e geom2 são o ponto vazio ou contém pontos vazios, então esses pontos vazios são ignorados.

Se geom1 e geom2 estão vazios, então a `LINESTRING`vazia é retornado.

O valor do identificador do sistema de referência espacial (SRID - spatial reference system identifier) da geometria retornada é o valor do SRID das geometrias de entrada.

Um erro será retornado se geom1 e geom2 tiverem valores de SRIDs diferentes.

Se geom1 ou geom2 não forem `POINT`, `LINESTRING` ou `MULTIPOINT`, será exibido um erro.

Se geom1 e geom2 tiverem dimensões diferentes, um erro será retornado.

Exemplos

O SQL a seguir cria uma linestring de duas linestrings de entrada.

```
SELECT ST_MakeLine(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27
 29.31,77.29 29.07)'), ST_GeomFromText('LINESTRING(88.29 39.07,88.42 39.26,88.27
 39.31,88.29 39.07)'));
```

```
st_makeline
-----------
 01020000000800000C3F5285C8F52534052B81E85EB113D407B14AE47E15A5340C3F5285C8F423D40E17A14AE4751
```

ST_MakePoint

ST_MakePoint retorna uma geometria de pontos cujos valores de coordenadas são os valores de entrada.

Sintaxe

```
ST_MakePoint(x, y)
```

```
ST_MakePoint(x, y, z)
```

```
ST_MakePoint(x, y, z, m)
```

Argumentos

x

Um valor de tipo de dados DOUBLE PRECISION que representa a primeira coordenada.

y

Um valor de tipo de dados DOUBLE PRECISION que representa a segunda coordenada.

z

Um valor de tipo de dados DOUBLE PRECISION que representa a terceira coordenada.

m

Um valor de tipo de dados DOUBLE PRECISION que representa a quarta coordenada.

Tipo de retorno

GEOMETRY de subtipo POINT.

O valor do identificador do sistema de referência espacial (SRID - spatial reference system identifier) da geometria retornada está definido como 0.

Se x, y, z ou m forem nulos, será exibido null.

Exemplos

O SQL a seguir retorna um tipo GEOMETRY de subtipo POINT com as coordenadas fornecidas.

```
SELECT ST_AsText(ST_MakePoint(1,3));
```

```
st_astext
-----------
 POINT(1 3)
```

O SQL a seguir retorna um tipo GEOMETRY de subtipo POINT com as coordenadas fornecidas.

```
SELECT ST_AsEWKT(ST_MakePoint(1, 2, 3));
```

```
st_asewkt
----------------
 POINT Z (1 2 3)
```

O SQL a seguir retorna um tipo GEOMETRY de subtipo POINT com as coordenadas fornecidas.

```
SELECT ST_AsEWKT(ST_MakePoint(1, 2, 3, 4));
```

```
st_asewkt
-------------------
 POINT ZM (1 2 3 4)
```

ST_MakePolygon

ST_MakePolygon tem duas variantes que retornam um polígono. Um leva uma única geometria, e outro leva duas geometrias.

- A entrada da primeira variante é uma linestring que define o anel externo do polígono de saída.
- A entrada da segunda variante é uma linestring e uma multilinestring. Ambas estão vazias ou fechadas.

  O limite do anel exterior do polígono de saída é a linestring de entrada, e os limites dos anéis interiores do polígono são as linestrings na entrada multilinestring. Se a linestring estiver vazia, um polígono vazio será retornado. Linestrings vazias na multilinestring são ignoradas. O valor do identificador do sistema de referência espacial (SRID - spatial reference system identifier) da geometria resultante é o SRID comum das duas geometrias de entrada.

A dimensão da geometria retornada é a mesma das geometrias de entrada. O anel exterior e os anéis interiores devem ter a mesma dimensão.

Sintaxe

```
ST_MakePolygon(geom1)
```

```
ST_MakePolygon(geom1, geom2)
```

Argumentos

geom1

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY. O subtipo deve ser LINESTRING. O valor de linestring deve ser fechado ou vazio.

geom2

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY. O subtipo deve ser MULTILINESTRING.

Tipo de retorno

GEOMETRY de subtipo POLYGON.

O identificador do sistema de referência espacial (SRID - spatial reference system identifier) da geometria retornada é igual ao SRID das entradas.

Se geom1 ou geom2 for nulo, null será retornado.

Se geom1 não for uma linestring, será retornado um erro.

Se geom2 não for uma multilinestring, será retornado um erro.

Se geom1 não for fechado, será retornado um erro.

Se geom1 for um único ponto ou não estiver fechado, será retornado um erro.

Se geom2 contém pelo menos uma linestring com um único ponto ou não for fechada, será retornado um erro.

Um erro será retornado se geom1 e geom2 tiverem valores de SRIDs diferentes.

Se geom1 e geom2 tiverem dimensões diferentes, um erro será retornado.

Exemplos

O SQL a seguir retorna um polígono de uma linestring de entrada.

```
SELECT ST_AsText(ST_MakePolygon(ST_GeomFromText('LINESTRING(77.29 29.07,77.42
 29.26,77.27 29.31,77.29 29.07)')));
```

```
 st_astext
---------------
POLYGON((77.29 29.07,77.42 29.26,77.27 29.31,77.29 29.07))
```

O SQL a seguir cria um polígono a partir de uma linestring fechada e uma multilinestring fechada. O linestring é usado para o anel externo do polígono. As linestrings nas multilinestrings são usadas para os anéis interiores do polígono.

```
SELECT ST_AsEWKT(ST_MakePolygon(ST_GeomFromText('LINESTRING(0 0,10 0,10 10,0 10,0 0)'),
 ST_GeomFromText('MULTILINESTRING((1 1,1 2,2 1,1 1),(3 3,3 4,4 3,3 3))')));
```

```
 st_astext
---------------------------------
POLYGON((0 0,10 0,10 10,0 10,0 0),(1 1,1 2,2 1,1 1),(3 3,3 4,4 3,3 3))
```

ST_MemSize

ST_MemSize retorna a quantidade de espaço de memória (em bytes) usada pela geometria de entrada. Esse tamanho depende da representação interna do Amazon Redshift da geometria e, portanto, pode mudar se a representação interna mudar. Você pode usar esse tamanho como uma indicação do tamanho relativo dos objetos de geometria no Amazon Redshift.

Sintaxe

```
ST_MemSize(geom)
```

Argumentos

geom

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`.

Tipo de retorno

`INTEGER` que representa a dimensão inerente de geom.

Nulo será retornado se geom for nulo.

Exemplos

O SQL a seguir retorna o tamanho da memória de uma coleção de geometrias.

```
SELECT ST_MemSize(ST_GeomFromText('GEOMETRYCOLLECTION(POLYGON((0 0,10 0,0 10,0
 0)),LINESTRING(20 10,20 0,10 0))')::varchar + ' bytes';
```

```
 ?column?
-----------
 172 bytes
```

ST_MMax

ST_MMax retorna a coordenada `m` máxima de uma geometria de entrada.

Sintaxe

```
ST_MMax(geom)
```

Argumentos

geom

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`.

Tipo de retorno

O valor de `DOUBLE PRECISION` da coordenada `m` máxima.

Nulo será retornado se geom estiver vazio.

Nulo será retornado se geom for nulo.

Se geom for uma geometria 2D ou 3DZ, null será retornado.

Exemplos

O SQL a seguir retorna a maior coordenada `m` de uma linestring em uma geometria 3DM.

```
SELECT ST_MMax(ST_GeomFromEWKT('LINESTRING M (0 1 2, 3 4 5, 6 7 8)'));
```

```
st_mmax
-----------
  8
```

O SQL a seguir retorna a maior coordenada `m` de uma linestring em uma geometria 4DM.

```
SELECT ST_MMax(ST_GeomFromEWKT('LINESTRING ZM (0 1 2 3, 4 5 6 7, 8 9 10 11)'));
```

```
st_mmax
-----------
  11
```

ST_MMin

ST_YMin retorna a coordenada mínima `m` de uma geometria de entrada.

Sintaxe

```
ST_MMin(geom)
```

Argumentos

geom

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`.

Tipo de retorno

O valor de `DOUBLE PRECISION` da coordenada mínima `m` .

Nulo será retornado se geom estiver vazio.

Nulo será retornado se geom for nulo.

Se geom for uma geometria 2D ou 3DZ, null será retornado.

Exemplos

O SQL a seguir retorna a coordenada menor m de uma linestring em uma geometria 3DM.

```
SELECT ST_MMin(ST_GeomFromEWKT('LINESTRING M (0 1 2, 3 4 5, 6 7 8)'));
```

```
st_mmin
-----------
  2
```

O SQL a seguir retorna a coordenada menor m de uma linestring em uma geometria 4DM.

```
SELECT ST_MMin(ST_GeomFromEWKT('LINESTRING ZM (0 1 2 3, 4 5 6 7, 8 9 10 11)'));
```

```
st_mmin
-----------
  3
```

ST_Multi

ST_Multi converte uma geometria para o multitipo correspondente. Se a geometria de entrada já é um multitipo ou uma coleção de geometria, uma cópia dela é retornada. Se a geometria de entrada for ponto, uma linestring ou um polígono, será retornado um multiponto, um multipolígono ou um multipolígono, respectivamente, que contenha a geometria de entrada.

Sintaxe

```
ST_Multi(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

GEOMETRY com subtipo MULTIPOINT, MULTILINESTRING, MULTIPOLYGON ou GEOMETRYCOLLECTION.

O Spatial Reference System Identifier (SRID – Identificador do sistema de referência espacial) da geometria retornada é o mesmo da geometria de entrada.

Nulo será retornado se geom for nulo.

Exemplos

O SQL a seguir retorna um multiponto de um multiponto de entrada.

```
SELECT ST_AsEWKT(ST_Multi(ST_GeomFromText('MULTIPOINT((1 2),(3 4))', 4326)));
```

```
    st_asewkt
-----------------------------------
  SRID=4326;MULTIPOINT((1 2),(3 4))
```

O SQL a seguir retorna um multiponto de um ponto de entrada.

```
SELECT ST_AsEWKT(ST_Multi(ST_GeomFromText('POINT(1 2)', 4326)));
```

```
    st_asewkt
-----------------------------------
  SRID=4326;MULTIPOINT((1 2))
```

O SQL a seguir retorna uma coleção de geometria de uma coleção de geometria de entrada.

```
SELECT ST_AsEWKT(ST_Multi(ST_GeomFromText('GEOMETRYCOLLECTION(POINT(1 2),MULTIPOINT((1
 2),(3 4)))', 4326)));
```

```
    st_asewkt
-----------------------------------
  SRID=4326;GEOMETRYCOLLECTION(POINT(1 2),MULTIPOINT((1 2),(3 4)))
```

ST_NDims

ST_NDims retorna a dimensão de coordenadas de uma geometria. ST_NDims não considera a dimensão topológica de uma geometria. Em vez disso, ele retorna um valor constante dependendo da dimensão da geometria.

Sintaxe

```
ST_NDims(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

INTEGER que representa a dimensão inerente de geom.

Nulo será retornado se geom for nulo.

Os valores retornados são os seguintes.

Valor retornado	Dimensão da geometria de entrada
2	2D
3	3DZ ou 3DM
4	4D

Exemplos

O SQL a seguir retorna o número de dimensões de uma linestring 2D.

```
SELECT ST_NDims(ST_GeomFromText('LINESTRING(0 0,1 1,2 2,0 0)'));
```

```
st_ndims
-------------
 2
```

O SQL a seguir retorna o número de dimensões de uma linestring 3DZ.

```
SELECT ST_NDims(ST_GeomFromText('LINESTRING Z(0 0 3,1 1 3,2 2 3,0 0 3)'));
```

```
st_ndims
-------------
 3
```

O SQL a seguir retorna o número de dimensões de uma linestring 3DM.

```
SELECT ST_NDims(ST_GeomFromText('LINESTRING M(0 0 4,1 1 4,2 2 4,0 0 4)'));
```

```
st_ndims
-------------
 3
```

O SQL a seguir retorna o número de dimensões de uma linestring 4D.

```
SELECT ST_NDims(ST_GeomFromText('LINESTRING ZM(0 0 3 4,1 1 3 4,2 2 3 4,0 0 3 4)'));
```

```
st_ndims
-------------
 4
```

ST_NPoints

ST_NPoints retorna o número de pontos não vazios em uma geometria ou geografia de entrada.

Sintaxe

```
ST_NPoints(geo)
```

Argumentos

geo

Um valor de tipo de dados GEOMETRY ou GEOGRAPHY ou uma expressão que é avaliada como tipo GEOMETRY ou GEOGRAPHY.

Tipo de retorno

INTEGER

Se geo for um ponto vazio, será retornado 0.

Será retornado null, se geo for nulo.

Exemplos

O SQL a seguir retorna o número de pontos em uma linestring.

```
SELECT ST_NPoints(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27 29.31,77.29
 29.07)'));
```

```
st_npoints
-------------
 4
```

O SQL a seguir retorna o número de pontos em uma linestring em uma geografia.

```
SELECT ST_NPoints(ST_GeogFromText('LINESTRING(110 40, 2 3, -10 80, -7 9)'));
```

```
st_npoints
-------------
 4
```

ST_NRings

ST_NRings retorna o número de anéis em uma geometria de entrada.

Sintaxe

```
ST_NRings(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

INTEGER

Nulo será retornado se geom for nulo.

Os valores retornados são os seguintes.

Valor retornado	Subtipo de geometria
0	Retornado se geom for um subtipo de POINT, LINESTRING , MULTIPOINT ou MULTILINESTRING .
O número de anéis.	Retornado se geom for um subtipo de POLYGON ou de MULTIPOLYGON .
O número de anéis em todos os componentes.	Retornado se geom for um subtipo GEOMETRYCOLLECTION

Exemplos

O SQL a seguir retorna o número de pontos anéis um multipolígono.

```
SELECT ST_NRings(ST_GeomFromText('MULTIPOLYGON(((0 0,10 0,0 10,0 0)),((0 0,-10 0,0
 -10,0 0)))'));
```

```
 st_nrings
-------------
 2
```

ST_NumGeometries

ST_NumGeometries retorna o número de geometrias em uma geometria de entrada.

Sintaxe

```
ST_NumGeometries(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

INTEGER que representa o número de geometrias em geom.

Nulo será retornado se geom for nulo.

Se geom for uma geometria vazia única, então 0 será retornado.

Se geom for uma geometria não vazia única, então 1 será retornado.

Se geom for um subtipo de GEOMETRYCOLLECTION ou MULTI, então o número de geometrias será retornado.

Exemplos

O SQL a seguir retorna o número de geometrias em uma multilinestring de entrada.

```
SELECT ST_NumGeometries(ST_GeomFromText('MULTILINESTRING((0 0,1 0,0 5),(3 4,13 26))'));
```

```
st_numgeometries
-------------
```

```
 2
```

ST_NumInteriorRings

ST_NumInteriorRings retorna o número de anéis em uma geometria e polígono de entrada.

Sintaxe

```
ST_NumInteriorRings(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

INTEGER

Nulo será retornado se geom for nulo.

Nulo será retornado se geom não for um polígono.

Exemplos

O SQL a seguir retorna o número de anéis interiores no polígono entrada.

```
SELECT ST_NumInteriorRings(ST_GeomFromText('POLYGON((0 0,100 0,100 100,0 100,0 0),(1
 1,1 5,5 1,1 1),(7 7,7 8,8 7,7 7))'));
```

```
 st_numinteriorrings
-------------
 2
```

ST_NumPoints

ST_NumPoints retorna o número de pontos em uma geometria de entrada.

Sintaxe

```
ST_NumPoints(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

INTEGER

Nulo será retornado se geom for nulo.

Nulo será retornado se geom não for do subtipo LINESTRING.

Exemplos

O SQL a seguir retorna o número de pontos na linestring de entrada.

```
SELECT ST_NumPoints(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27
 29.31,77.29 29.07)'));
```

```
st_numpoints
-------------
4
```

O SQL a seguir retornará nulo porque a entrada geom não é do subtipo LINESTRING.

```
SELECT ST_NumPoints(ST_GeomFromText('MULTIPOINT(1 2,3 4)'));
```

```
st_numpoints
-------------
```

ST_Perimeter

Para uma geometria plana de entrada, ST_Perimeter retorna o perímetro cartesiano (comprimento do limite) da projeção 2D. As unidades de perímetro são as mesmas unidades em que as coordenadas da geometria de entrada são expressas. A função retorna zero (0) para pontos, multipontos e geometrias lineares. Quando a entrada for uma coleção de geometrias, a função retornará a soma dos perímetros das geometrias na coleção.

Para uma geografia de entrada, o ST_Perimeter retorna a área geodésica (comprimento do limite) da projeção 2D de uma geografia plana de entrada computada no esferoide determinado pelo SRID. A unidade de perímetro é o metro. A função retorna zero (0) para pontos, multipontos e geografias lineares. Quando a entrada for uma coleção de geometrias, a função retornará a soma dos perímetros das geografias da coleção.

Sintaxe

```
ST_Perimeter(geo)
```

Argumentos

geo

Um valor de tipo de dados `GEOMETRY` ou `GEOGRAPHY` ou uma expressão que é avaliada como tipo `GEOMETRY` ou `GEOGRAPHY`.

Tipo de retorno

`DOUBLE PRECISION`

Será retornado null, se geo for nulo.

Se o valor de SRID não for encontrado, será retornado um erro.

Exemplos

O SQL a seguir retorna o perímetro cartesiano de um multipolígono.

```
SELECT ST_Perimeter(ST_GeomFromText('MULTIPOLYGON(((0 0,10 0,0 10,0 0)),((10 0,20 0,20
 10,10 0)))'));
```

```
 st_perimeter
------------------------------
    68.2842712474619
```

O SQL a seguir retorna o perímetro cartesiano de um multipolígono.

```
SELECT ST_Perimeter(ST_GeomFromText('MULTIPOLYGON(((0 0,10 0,0 10,0 0)),((10 0,20 0,20
 10,10 0)))'));
```

```
 st_perimeter
------------------------------
    68.2842712474619
```

O SQL a seguir retorna o perímetro de um polígono em uma geografia.

```
SELECT ST_Perimeter(ST_GeogFromText('SRID=4326;POLYGON((0 0,1 0,0 1,0 0))'));
```

```
 st_perimeter
------------------
 378790.428393693
```

O SQL a seguir retorna o perímetro de uma linestring em uma geografia.

```
SELECT ST_Perimeter(ST_GeogFromText('SRID=4326;LINESTRING(5 0,10 0)'));
```

```
 st_perimeter
--------------
 0
```

ST_Perimeter2D

ST_Perimeter2D é um alias para ST_Perimeter. Para obter mais informações, consulte ST_Perimeter.

ST_Point

ST_Point retorna uma geometria de pontos dos valores de coordenadas de entrada.

Sintaxe

```
ST_Point(x, y)
```

Argumentos

x

Um valor de tipo de dados DOUBLE PRECISION que representa uma primeira coordenada.

y

Um valor de tipo de dados DOUBLE PRECISION que representa uma segunda coordenada.

Tipo de retorno

GEOMETRY de subtipo POINT.

O valor do identificador do sistema de referência espacial (SRID - spatial reference system identifier) da geometria retornada está definido como 0.

Nulo será retornado se x ou y for nulo.

Exemplos

O SQL a seguir cria uma geometria de pontos das coordenadas de entrada.

```
SELECT ST_AsText(ST_Point(5.0, 7.0));
```

```
st_astext
-------------
POINT(5 7)
```

ST_PointN

ST_PointN retorna um ponto em uma linestring, conforme especificado por um valor de índice. Os valores de índice negativos são contados de maneira regressiva, começando pelo final da linestring, de modo que -1 seja o último ponto.

A dimensão da geometria retornada é a mesma geometria de entrada.

Sintaxe

```
ST_PointN(geom, index)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY. O subtipo deve ser LINESTRING.

Índice do

Um valor do tipo de dados INTEGER que representa o índice de um ponto em uma linestring.

Tipo de retorno

GEOMETRY de subtipo POINT.

O valor do identificador do sistema de referência espacial (SRID - spatial reference system identifier) da geometria retornada está definido como 0.

Se geom ou index for null, será retornado null.

Se index estiver fora do intervalo, será retornado null.

Nulo será retornado se geom estiver vazio.

Se geom não for uma LINESTRING, será retornado null.

Exemplos

O SQL a seguir retorna uma representação de Extended well-known text (EWKT – Texto bem-conhecido estendido) de uma LINESTRING de seis pontos para um objeto GEOMETRY e retorna o ponto do índice 5 da linestring.

```
SELECT ST_AsEWKT(ST_PointN(ST_GeomFromText('LINESTRING(0 0,10 0,10 10,5 5,0 5,0
 0)',4326), 5));
```

```
st_asewkt
-------------
```

```
 SRID=4326;POINT(0 5)
```

ST_Points

ST_Points retorna uma geometria multiponto contendo todos os pontos não vazios na geometria de entrada. ST_Points não remove pontos duplicados na entrada, incluindo os pontos inicial e final das geometrias de anel.

Sintaxe

```
ST_Points(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

GEOMETRY de subtipo MULTIPOINT.

O valor do identificador do sistema de referência espacial (SRID) da geometria retornada é o mesmo que geom.

Nulo será retornado se geom for nulo.

Se geom está vazio, então o multiponto vazio será retornado.

Exemplos

Os exemplos SQL a seguir constroem uma geometria multiponto a partir da geometria de entrada. O resultado é uma geometria multiponto contendo os pontos não vazios na geometria de entrada.

```
SELECT ST_AsEWKT(ST_Points(ST_SetSRID(ST_GeomFromText('LINESTRING(1 0,2 0,3 0)'),
 4326)));
```

```
st_asewkt
-------------
```

```
SRID=4326;MULTIPOINT((1 0),(2 0),(3 0))
```

```
SELECT ST_AsEWKT(ST_Points(ST_SetSRID(ST_GeomFromText('MULTIPOLYGON(((0 0,1 0,0 1,0
 0)))'), 4326)));
```

```
st_asewkt
-------------
SRID=4326;MULTIPOINT((0 0),(1 0),(0 1),(0 0))
```

ST_Polygon

ST_Polygon retorna uma geometria de polígono cujo anel externo é a linestring de entrada com o valor que era a entrada do identificador do sistema de referência (SRID - spatial reference system identifier)

A dimensão da geometria retornada é a mesma geometria de entrada.

Sintaxe

```
ST_Polygon(linestring, srid)
```

Argumentos

linestring

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`. O subtipo deve ser `LINESTRING` que representa uma linestring. O valor de linestring deve ser fechado.

srid

Um valor de tipo de dados `INTEGER` que representa uma segunda um SRID.

Tipo de retorno

`GEOMETRY` de subtipo `POLYGON`.

Caso contrário, o valor do SRID da geometria retornado será definido como srid.

Nulo será retornado se linestring ou srid for nulo.

Um erro será retornado se linestring não for uma linestring.

Um erro será retornado se linestring não for fechada.

Um erro será retornado se srid for negativo.

Exemplos

O SQL a seguir cria um polígono com um valor de SRID.

```
SELECT ST_AsEWKT(ST_Polygon(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27
 29.31,77.29 29.07)'),4356));
```

```
st_asewkt
-------------
 SRID=4356;POLYGON((77.29 29.07,77.42 29.26,77.27 29.31,77.29 29.07))
```

ST_RemovePoint

ST_RemovePoint retorna uma geometria de linestring com o ponto da geometria de entrada em uma posição de índice removido.

O índice é baseado em zero. O Spatial Reference System Identifier (SRID – Identificador do sistema de referência espacial) do resultado é o mesmo da geometria de entrada. A dimensão da geometria retornada é a mesma geometria de entrada.

Sintaxe

```
ST_RemovePoint(geom, index)
```

Argumentos

geom

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`. O subtipo deve ser `LINESTRING`.

Índice do

Um valor do tipo de dados `INTEGER` que representa a posição de um índice baseado em zero.

Tipo de retorno

GEOMETRY

Se geom ou index for null, será retornado null.

Se geom não for do subtipo LINESTRING, será retornado um erro.

Se index estiver fora do intervalo, será retornado um erro. Os valores válidos para a posição do índice estão entre 0 e ST_NumPoints(geom) menos 1.

Exemplos

O SQL a seguir remove o último ponto de uma linestring.

```
WITH tmp(g) AS (SELECT ST_GeomFromText('LINESTRING(0 0,10 0,10 10,5 5,0 5)',4326))
SELECT ST_AsEWKT(ST_RemovePoint(g, ST_NumPoints(g) - 1)) FROM tmp;
```

```
   st_asewkt
----------------------------------------
 SRID=4326;LINESTRING(0 0,10 0,10 10,5 5)
```

ST_Reverse

ST_Reverse inverte a ordem dos vértices para geometrias lineares e areais. Para geometrias de ponto ou multiponto, uma cópia da geometria original é retornada. Para coleções de geometrias, ST_Reverse inverte a ordem dos vértices para cada uma das geometrias presentes na coleção.

A dimensão da geometria retornada é a mesma geometria de entrada.

Sintaxe

```
ST_Reverse(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

GEOMETRY

O Spatial Reference System Identifier (SRID – Identificador do sistema de referência espacial) da geometria retornada é o mesmo da geometria de entrada.

Nulo será retornado se geom for nulo.

Exemplos

O SQL a seguir inverte a ordem dos pontos em uma linestring.

```
SELECT ST_AsEWKT(ST_Reverse(ST_GeomFromText('LINESTRING(1 0,2 0,3 0,4 0)', 4326)));
```

```
    st_asewkt
-----------------------------------
  SRID=4326;LINESTRING(4 0,3 0,2 0,1 0)
```

ST_SetPoint

ST_SetPoint retorna uma linestring com coordenadas atualizadas em relação à posição do linestring de entrada conforme especificado pelo índice. As novas coordenadas são as coordenadas do ponto de entrada.

A dimensão da geometria retornada é a mesma do valor de geom1. Se geom1 e geom2 têm dimensões diferentes, geom2 é projetado para a dimensão de geom1.

Sintaxe

```
ST_SetPoint(geom1, index, geom2)
```

Argumentos

geom1

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY. O subtipo deve ser LINESTRING.

Índice do

Um valor de tipo de dados `INTEGER` que representa a posição de um índice. Um `0` refere-se ao primeiro ponto da linestring a partir da esquerda, `1` refere-se ao segundo ponto, e assim por diante. O índice pode ser um valor negativo. Um `-1` refere-se ao primeiro ponto da linestring a partir da direita, `-2` refere-se ao segundo ponto, e assim por diante.

geom2

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`. O subtipo deve ser `POINT`.

Tipo de retorno

`GEOMETRY`

Se geom2 for um ponto vazio, então geom1 será retornado.

Se geom1, geom2 ou index for null, será retornado null.

Se geom1 não for uma linestring, será retornado um erro.

Se índice não está dentro de um intervalo de índice válido, então será retornado um erro.

Se geom2 não for um ponto, então será retornado um erro.

Um erro será retornado se geom1 e geom2 não tiverem o mesmo valor do identificador do sistema referência espacial (SRID -spatial reference system identifier).

Exemplos

O SQL a seguir retorna uma nova linestring na qual o segundo ponto da linestring de entrada é definido como o ponto especificado.

```
SELECT ST_AsText(ST_SetPoint(ST_GeomFromText('LINESTRING(1 2, 3 2, 5 2, 1 2)'), 2,
 ST_GeomFromText('POINT(7 9)')));
```

```
st_astext
-------------
```

```
 LINESTRING(1 2,3 2,7 9,1 2)
```

O exemplo de SQL a seguir retorna uma nova linestring na qual o terceiro ponto a partir da direita da linestring, cujo índice é negativo, é definido como o ponto especificado.

```
SELECT ST_AsText(ST_SetPoint(ST_GeomFromText('LINESTRING(1 2, 3 2, 5 2, 1 2)'), -3,
 ST_GeomFromText('POINT(7 9)')));
```

```
st_astext
-------------
 LINESTRING(1 2,7 9,5 2,1 2)
```

ST_SetSRID

ST_SetSRID retorna uma geometria que é a mesma geometria de entrada, mas que está atualizada com a entrada do valor do identificador do sistema de referência espacial (SRID - spatial reference system identifier).

Sintaxe

```
ST_SetSRID(geom, srid)
```

Argumentos

geom

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`.

srid

Um valor de tipo de dados `INTEGER` que representa uma segunda um SRID.

Tipo de retorno

`GEOMETRY`

Caso contrário, o valor do SRID da geometria retornado será definido como srid.

Nulo será retornado se geom ou srid for nulo.

Um erro será retornado se srid for negativo.

Exemplos

O SQL a seguir define o valor do SRID de uma linestring.

```
SELECT ST_AsEWKT(ST_SetSRID(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27
 29.31,77.29 29.07)'),50));
```

```
st_asewkt
-------------
 SRID=50;LINESTRING(77.29 29.07,77.42 29.26,77.27 29.31,77.29 29.07)
```

ST_Simplify

ST_Simplify retorna uma cópia simplificada da geometria de entrada usando o algoritmo Ramer-Douglas-Peucker com a tolerância dada. A topologia da geometria de entrada pode não ser preservada. Para obter mais informações sobre o algoritmo, consulte Algoritmo Ramer—Douglas—Peucker na Wikipédia.

Quando ST_Simplify calcula distâncias para simplificar uma geometria, ST_Simplify opera na projeção 2D da geometria de entrada.

Sintaxe

```
ST_Simplify(geom, tolerance)
```

Argumentos

geom

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`.

tolerance

Um valor de tipo de dados `DOUBLE PRECISION` que representa o nível de tolerância do algoritmo Ramer-Douglas-Peucker. Se tolerance for um número negativo, então zero será usado.

Tipo de retorno

`GEOMETRY`.

O valor do identificador do sistema de referência espacial (SRID - spatial reference system identifier) da geometria retornada é o valor do SRID das geometrias de entrada.

A dimensão da geometria retornada é a mesma geometria de entrada.

Nulo será retornado se geom for nulo.

Exemplos

O SQL a seguir simplifica a linestring de entrada usando uma tolerância de distância euclidiana de 1 com o algoritmo Ramer-Douglas-Peucker. As unidades da distância são iguais às das coordenadas da geometria.

```
SELECT ST_AsEWKT(ST_Simplify(ST_GeomFromText('LINESTRING(0 0,1 2,1 1,2 2,2 1)'), 1));
```

```
 st_asewkt
-----------
LINESTRING(0 0,1 2,2 1)
```

ST_SRID

ST_SRID retorna o identificador do sistema de referência espacial (SRID - spatial reference system identifier) de uma geometria de entrada. Para obter mais informações sobre um SRID, consulte Consultar dados espaciais no Amazon Redshift.

Sintaxe

```
ST_SRID(geom)
```

Argumentos

geom

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`.

Tipo de retorno

INTEGER que representa o valor do SRID de geom.

Nulo será retornado se geom for nulo.

Exemplos

O SQL a seguir retorna o valor de um SRID de uma string de linha definida como SRID 4326.

```
SELECT ST_SRID(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27 29.31,77.29
 29.07)',4326));
```

```
st_srid
-------------
 4326
```

O SQL a seguir retorna um valor de SRID de uma string de linha que não está definida quando construída. Isso resulta em 0 para o valor de SRID.

```
SELECT ST_SRID(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27 29.31,77.29
 29.07)'));
```

```
st_srid
-------------
 0
```

ST_StartPoint

ST_StartPoint retorna o primeiro ponto de uma linestring de entrada. O valor do Spatial Reference System Identifier (SRID – Identificador do sistema de referência espacial) do resultado é o mesmo da geometria de entrada. A dimensão da geometria retornada é a mesma geometria de entrada.

Sintaxe

```
ST_StartPoint(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY. O subtipo deve ser LINESTRING.

Tipo de retorno

GEOMETRY

Nulo será retornado se geom for nulo.

Nulo será retornado se geom estiver vazio.

Se geom não for uma LINESTRING, será retornado null.

Exemplos

O SQL a seguir retorna uma representação de Extended well-known text (EWKT – Texto bem-conhecido estendido) de uma LINESTRING de quatro pontos para um objeto GEOMETRY e retorna o ponto inicial da linestring.

```
SELECT ST_AsEWKT(ST_StartPoint(ST_GeomFromText('LINESTRING(0 0,10 0,10 10,5 5,0
 5)',4326)));
```

```
st_asewkt
-------------
 SRID=4326;POINT(0 0)
```

ST_Touches

ST_Touches retorna true se as projeções 2D das duas geometrias de entrada se tocarem. As duas geometrias se tocam se não estiverem vazias, se houver interseção entre elas e nenhum ponto interior em comum.

Sintaxe

```
ST_Touches(geom1, geom2)
```

Argumentos

geom1

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

geom2

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

BOOLEAN

Nulo será retornado se geom1 ou geom2 for nulo.

Um erro será retornado se geom1 e geom2 não tiverem o mesmo valor do identificador do sistema referência espacial (SRID -spatial reference system identifier).

Um erro será retornado se geom1 ou geom2 for uma coleção de geometrias.

Exemplos

O SQL a seguir verifica se um polígono toca uma linestring.

```
SELECT ST_Touches(ST_GeomFromText('POLYGON((0 0,10 0,0 10,0 0))'),
 ST_GeomFromText('LINESTRING(20 10,20 0,10 0)'));
```

```
 st_touches
-------------
 t
```

ST_Transform

ST_Transform retorna uma nova geometria com coordenadas que são transformadas em um sistema de referência espacial definido pelo identificador de sistema de referência espacial (SRID) de entrada.

Sintaxe

```
ST_Transform(geom, srid)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

srid

Um valor de tipo de dados INTEGER que representa uma SRID.

Tipo de retorno

GEOMETRY.

Caso contrário, o valor do SRID da geometria retornado será definido como srid.

Nulo será retornado se geom ou srid for nulo.

Se o valor de SRID associado à geom de entrada não existe, será retornado um erro.

Se srid não existir, será retornado um erro.

Exemplos

O SQL a seguir transforma o SRID de uma coleção de geometria vazia.

```
SELECT ST_AsEWKT(ST_Transform(ST_GeomFromText('GEOMETRYCOLLECTION EMPTY', 3857),
 4326));
```

```
            st_asewkt
-----------------------------------
 SRID=4326;GEOMETRYCOLLECTION EMPTY
```

O SQL a seguir transforma o SRID de uma linestring.

```
SELECT ST_AsEWKT(ST_Transform(ST_GeomFromText('LINESTRING(110 40, 2 3, -10 80, -7 9,
 -22 -33)', 4326), 26918));
```

```
     st_asewkt
-----------------------------------------------------------------------------------------------
 SRID=26918;LINESTRING(73106.6977300955 15556182.9688576,14347201.5059964
 1545178.32934967,1515090.41262989 9522193.25115316,10491250.83295
 2575457.28410878,5672303.72135968 -5233682.61176205)
```

O SQL a seguir transforma o SRID de um polígono.

```
SELECT ST_AsEWKT(ST_Transform(ST_GeomFromText('POLYGON Z ((-10 10 -7, -65 10 -6, -10 64
 -5, -10 10 -7), (-11 11 5, -11 12 6, -12 11 7, -11 11 5))', 6989), 6317));
```

```
                                      st_asewkt
-----------------------------------------------------------------------------------------------
 SRID=6317;POLYGON Z ((6186430.2771091 -1090834.57212608
 1100247.33216237,2654831.67853801 -5693304.90741276 1100247.50581055,2760987.41750022
 -486836.575101877 5709710.44137268,6186430.2771091 -1090834.57212608
 1100247.33216237),(6146675.25029258 -1194792.63532103 1209007.1115113,6125027.87562215
 -1190584.81194058 1317403.77865723,6124888.99555252 -1301885.3455052
 1209007.49312929,6146675.25029258 -1194792.63532103 1209007.1115113))
```

ST_Union

ST_Union retorna uma geometria que representa a união de duas geometrias. Ou seja, as geometrias de entrada são mescladas para produzir uma geometria sem sobreposições.

Sintaxe

```
ST_Union(geom1, geom2)
```

Argumentos

geom1

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

geom2

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

GEOMETRY

O valor do identificador do sistema de referência espacial (SRID - spatial reference system identifier) da geometria retornada é o valor do SRID das geometrias de entrada.

Nulo será retornado se geom1 ou geom2 for nulo.

Se geom1 e geom2 estiverem vazios, será retornada uma geometria vazia.

Um erro será retornado se geom1 e geom2 não tiverem o mesmo valor do identificador do sistema referência espacial (SRID -spatial reference system identifier).

Se geom1 ou geom2 for um conjunto de geometrias, linestring ou multilinestring, será retornado um erro.

Se geom1 ou geom2 não forem uma geometria bidimensional (2D), será retornado um erro.

Exemplos

O SQL a seguir retorna a geometria não vazia que representa a união de duas geometrias de entrada.

```
SELECT ST_AsEWKT(ST_Union(ST_GeomFromText('POLYGON((0 0,100 100,0 200,0 0))'),
 ST_GeomFromText('POLYGON((0 0,10 0,0 10,0 0))')));
```

```
        st_asewkt
-------------------------
```

```
POLYGON((0 0,0 200,100 100,5 5,10 0,0 0))
```

ST_Within

ST_Within retorna true se a projeção 2D da primeira geometria de entrada estiver dentro da projeção 2D da segunda geometria de entrada.

Por exemplo, a geometria A estará dentro da geometria B se todos os pontos em A forem um ponto em B e seus interiores tiverem interseção não vazia.

ST_Within(A, B) é equivalente a ST_Contains(B, A).

Sintaxe

```
ST_Within(geom1, geom2)
```

Argumentos

geom1

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY. Esse valor é comparado com geom2 para determinar se ele está dentro de geom2.

geom2

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

BOOLEAN

Nulo será retornado se geom1 ou geom2 for nulo.

Um erro será retornado se geom1 e geom2 não tiverem o mesmo valor do identificador do sistema referência espacial (SRID -spatial reference system identifier).

Um erro será retornado se geom1 ou geom2 for uma coleção de geometrias.

Exemplos

O SQL a seguir verifica se o primeiro polígono está dentro do segundo polígono.

```
SELECT ST_Within(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'),
 ST_GeomFromText('POLYGON((-1 3,2 1,0 -3,-1 3))'));
```

```
st_within
-----------
 true
```

ST_X

ST_X retorna a primeira coordenada de um ponto de entrada.

Sintaxe

```
ST_X(point)
```

Argumentos

point

Um valor de POINT de tipo de dados GEOMETRY.

Tipo de retorno

DOUBLE PRECISIONO valor de da primeira coordenada.

Nulo será retornado se point for nulo.

Se point for um ponto vazio, null será retornado.

Se point não for um POINT, um erro será retornado.

Exemplos

O SQL a seguir retorna a primeira coordenada de um ponto.

```
SELECT ST_X(ST_Point(1,2));
```

```
st_x
```

```
-----------
 1.0
```

ST_XMax

ST_XMax retorna a primeira coordenada máxima de uma geometria de entrada.

Sintaxe

```
ST_XMax(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

DOUBLE PRECISIONO valor de da primeira coordenada máxima.

Nulo será retornado se geom estiver vazio.

Nulo será retornado se geom for nulo.

Exemplos

O SQL a seguir retorna a primeira coordenada maior de uma linestring.

```
SELECT ST_XMax(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27 29.31,77.29
 29.07)'));
```

```
st_xmax
-----------
 77.42
```

ST_XMin

ST_XMin retorna a primeira coordenada mínima de uma geometria de entrada.

Sintaxe

```
ST_XMin(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

DOUBLE PRECISIONO valor de da primeira coordenada mínima.

Nulo será retornado se geom estiver vazio.

Nulo será retornado se geom for nulo.

Exemplos

O SQL a seguir retorna a primeira coordenada menor de uma linestring.

```
SELECT ST_XMin(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27 29.31,77.29
 29.07)'));
```

```
st_xmin
-----------
 77.27
```

ST_Y

ST_Y retorna a segunda coordenada de um ponto de entrada.

Sintaxe

```
ST_Y(point)
```

Argumentos

point

Um valor de POINT de tipo de dados GEOMETRY.

Tipo de retorno

DOUBLE PRECISIONO valor de da segunda coordenada.

Nulo será retornado se point for nulo.

Se point for um ponto vazio, null será retornado.

Se point não for um POINT, um erro será retornado.

Exemplos

O SQL a seguir retorna a segunda coordenada de um ponto.

```
SELECT ST_Y(ST_Point(1,2));
```

```
st_y
-----------
 2.0
```

ST_YMax

ST_YMax retorna a segunda coordenada máxima de uma geometria de entrada.

Sintaxe

```
ST_YMax(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

DOUBLE PRECISIONO valor de da segunda coordenada máxima.

Nulo será retornado se geom estiver vazio.

Nulo será retornado se geom for nulo.

Exemplos

O SQL a seguir retorna a segunda coordenada maior de uma linestring.

```
SELECT ST_YMax(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27 29.31,77.29
 29.07)'));
```

```
st_ymax
-----------
 29.31
```

ST_YMin

ST_YMin retorna a segunda coordenada mínima de uma geometria de entrada.

Sintaxe

```
ST_YMin(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

DOUBLE PRECISIONO valor de da segunda coordenada mínima.

Nulo será retornado se geom estiver vazio.

Nulo será retornado se geom for nulo.

Exemplos

O SQL a seguir retorna a segunda coordenada menor de uma linestring.

```
SELECT ST_YMin(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27 29.31,77.29
 29.07)'));
```

```
st_ymin
-----------
 29.07
```

ST_Z

ST_Z retorna a coordenada z de um ponto de entrada.

Sintaxe

```
ST_Z(point)
```

Argumentos

point

Um valor de POINT de tipo de dados GEOMETRY.

Tipo de retorno

O valor de DOUBLE PRECISION da coordenada m.

Nulo será retornado se point for nulo.

Se point é um ponto 2D ou 3DM, então null será retornado.

Se point for um ponto vazio, null será retornado.

Se point não for um POINT, um erro será retornado.

Exemplos

O SQL a seguir retorna a coordenada z de um ponto em uma geometria 3DZ.

```
SELECT ST_Z(ST_GeomFromEWKT('POINT Z (1 2 3)'));
```

```
st_z
-----------
 3
```

O SQL a seguir retorna a coordenada z de um ponto em uma geometria 4D.

```
SELECT ST_Z(ST_GeomFromEWKT('POINT ZM (1 2 3 4)'));
```

```
st_z
-----------
 3
```

ST_ZMax

ST_ZMax retorna a coordenada máxima z de uma geometria de entrada.

Sintaxe

```
ST_ZMax(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

O valor de DOUBLE PRECISION da coordenada z máxima.

Nulo será retornado se geom estiver vazio.

Nulo será retornado se geom for nulo.

Se geom é uma geometria 2D ou 3DM, então null será retornado.

Exemplos

O SQL a seguir retorna a menor coordenada z de uma linestring em uma geometria 3DZ.

```
SELECT ST_ZMax(ST_GeomFromEWKT('LINESTRING Z (0 1 2, 3 4 5, 6 7 8)'));
```

```
st_zmax
-----------
  8
```

O SQL a seguir retorna a maior coordenada z de uma linestring em uma geometria 4D.

```
SELECT ST_ZMax(ST_GeomFromEWKT('LINESTRING ZM (0 1 2 3, 4 5 6 7, 8 9 10 11)'));
```

```
st_zmax
-----------
  10
```

ST_ZMin

ST_ZMin retorna a coordenada mínima z de uma geometria de entrada.

Sintaxe

```
ST_ZMin(geom)
```

Argumentos

geom

Um valor de tipo de dados `GEOMETRY` ou uma expressão que é avaliada como um tipo `GEOMETRY`.

Tipo de retorno

O valor de `DOUBLE PRECISION` da coordenada mínima `z` .

Nulo será retornado se geom estiver vazio.

Nulo será retornado se geom for nulo.

Se geom é uma geometria 2D ou 3DM, então null será retornado.

Exemplos

O SQL a seguir retorna a menor coordenada z de uma linestring em uma geometria 3DZ.

```
SELECT ST_ZMin(ST_GeomFromEWKT('LINESTRING Z (0 1 2, 3 4 5, 6 7 8)'));
```

```
st_zmin
-----------
  2
```

O SQL a seguir retorna a coordenada menor z de uma linestring em uma geometria 4DM.

```
SELECT ST_ZMin(ST_GeomFromEWKT('LINESTRING ZM (0 1 2 3, 4 5 6 7, 8 9 10 11)'));
```

```
st_zmin
-----------
  2
```

SupportsBBox

SupportsBBox retornará true se a geometria de entrada suportar a codificação com uma caixa delimitadora pré-computada. Para obter mais informações sobre o suporte para caixas delimitadoras, consulte Caixa delimitadora.

Sintaxe

```
SupportsBBox(geom)
```

Argumentos

geom

Um valor de tipo de dados GEOMETRY ou uma expressão que é avaliada como um tipo GEOMETRY.

Tipo de retorno

BOOLEAN

Nulo será retornado se geom for nulo.

Exemplos

O comando SQL a seguir retorna true porque a forma geométrica do ponto de entrada suporta ser codificada com uma caixa delimitadora.

```
SELECT SupportsBBox(AddBBox(ST_GeomFromText('POLYGON((0 0,1 0,0 1,0 0))')));
```

```
supportsbbox
--------------
t
```

O SQL a seguir retorna false porque a geometria do ponto de entrada não suporta ser codificado com uma caixa delimitadora.

```
SELECT SupportsBBox(DropBBox(ST_GeomFromText('POLYGON((0 0,1 0,0 1,0 0))')));
```

```
supportsbbox
--------------
f
```

Funções de string

Tópicos

- Operador || (Concatenação)
- Função ASCII
- Função BPCHARCMP
- Função BTRIM
- Função BTTEXT_PATTERN_CMP
- Função CHAR_LENGTH

- Função CHARACTER_LENGTH
- Função CHARINDEX
- Função CHR
- Função COLLATE
- Função CONCAT
- Função CRC32
- Função DIFFERENCE
- Função INITCAP
- Funções LEFT e RIGHT
- Função LEN
- Função LENGTH
- Função LOWER
- Funções LPAD e RPAD
- Função LTRIM
- Função OCTETINDEX
- Função OCTET_LENGTH
- Função POSITION
- Função QUOTE_IDENT
- Função QUOTE_LITERAL
- Função REGEXP_COUNT
- Função REGEXP_INSTR
- Função REGEXP_REPLACE
- Função REGEXP_SUBSTR
- Função REPEAT
- Função REPLACE
- Função REPLICATE
- Função REVERSE
- Função RTRIM
- Função SOUNDEX

- Função SPLIT_PART
- Função STRPOS
- Função STRTOL
- Função SUBSTRING
- Função TEXTLEN
- Função TRANSLATE
- Função TRIM
- Função UPPER

Funções de string processam e manipulam strings de caracteres ou expressões que avaliam para strings de caracteres. Quando o argumento string nessas funções é um valor literal, ele deve ser envolvido por aspas simples. Os tipos de dados compatíveis incluem CHAR e VARCHAR.

A próxima seção fornece os nomes de função, sintaxe e descrições para as funções compatíveis. Todos os deslocamentos em strings são baseados em um.

Funções somente nó de liderança defasadas

As seguintes funções de string estão defasadas, pois são executadas somente no nó líder. Para obter mais informações, consulte Função de apenas nó líder

- GET_BYTE
- SET_BIT
- SET_BYTE
- TO_ASCII

Operador || (Concatenação)

Concatena duas expressões em ambos os lados do símbolo `||` e retorna a expressão concatenada.

Similar a Função CONCAT.

> **Note**
>
> Se uma ou ambas as expressões forem nulas, o resultado da concatenação será `NULL`.

Sintaxe

```
expression1 || expression2
```

Argumentos

expression1

Uma string CHAR, uma string VARCHAR, uma expressão binária ou uma expressão avaliada como um desses tipos.

expression2

Uma string CHAR, uma string VARCHAR, uma expressão binária ou uma expressão avaliada como um desses tipos.

Tipo de retorno

O tipo retornado da string é o mesmo que o dos argumentos de entrada. Por exemplo, concatenar duas strings do tipo VARCHAR retorna uma string do tipo VARCHAR.

Exemplos

Os exemplos a seguir usam as tabelas USERS e VENUE do banco de dados de amostra TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para concatenar os campos FIRSTNAME e LASTNAME da tabela USERS no banco de dados de amostra, use o exemplo a seguir.

```
SELECT (firstname || ' ' || lastname) as fullname
FROM users
ORDER BY 1
LIMIT 10;

+-----------------+
|    fullname     |
+-----------------+
| Aaron Banks     |
| Aaron Booth     |
| Aaron Browning  |
| Aaron Burnett   |
```

```
| Aaron Casey     |
| Aaron Cash      |
| Aaron Castro    |
| Aaron Dickerson |
| Aaron Dixon     |
| Aaron Dotson    |
+-----------------+
```

Para concatenar colunas que possam conter nulos, use a expressão Funções NVL e COALESCE. O exemplo a seguir usa NVL para retornar um `0` sempre que NULL for encontrado.

```
SELECT (venuename || ' seats ' || NVL(venueseats, 0)) as seating
FROM venue
WHERE venuestate = 'NV' or venuestate = 'NC'
ORDER BY 1
LIMIT 10;

+-------------------------------------+
|               seating               |
+-------------------------------------+
| Ballys Hotel seats 0                |
| Bank of America Stadium seats 73298 |
| Bellagio Hotel seats 0              |
| Caesars Palace seats 0              |
| Harrahs Hotel seats 0               |
| Hilton Hotel seats 0                |
| Luxor Hotel seats 0                 |
| Mandalay Bay Hotel seats 0          |
| Mirage Hotel seats 0                |
| New York New York seats 0           |
+-------------------------------------+
```

Função ASCII

A função ASCII retorna o código ASCII, ou o ponto de código Unicode, do primeiro caractere na string que você especificar. A função retornará `0` se a string estiver vazia. Será retornado NULL se a string for nula.

Sintaxe

```
ASCII('string')
```

Argumento

string

Uma string CHAR ou uma string VARCHAR.

Tipo de retorno

INTEGER

Exemplos

Para retornar NULL, use o exemplo a seguir. A função NULLIF retornará NULL se os dois argumentos forem iguais, então o argumento de entrada da função ASCII será NULL. Para obter mais informações, consulte Função NULLIF.

```
SELECT ASCII(NULLIF('',''));

+-------+
| ascii |
+-------+
|  NULL |
+-------+
```

Para retornar o código ASCII 0, use o exemplo a seguir.

```
SELECT ASCII('');

+-------+
| ascii |
+-------+
|     0 |
+-------+
```

Para retornar o código ASCII 97 para a primeira letra da palavra amazon, use o exemplo a seguir.

```
SELECT ASCII('amazon');

+-------+
| ascii |
+-------+
```

```
|    97 |
+-------+
```

Para retornar o código ASCII 65 para a primeira letra da palavra Amazon, use o exemplo a seguir.

```
SELECT ASCII('Amazon');

+-------+
| ascii |
+-------+
|    65 |
+-------+
```

Função BPCHARCMP

Compara o valor de duas strings e retorna um número inteiro. Se as strings forem idênticas, a função retornará `0`. Se a primeira string for alfabeticamente posterior, a função retornará `1`. Se a segunda string for posterior, a função retornará `-1`.

Para caracteres multibyte, a comparação é baseada na codificação de byte.

Sinônimo de Função BTTEXT_PATTERN_CMP.

Sintaxe

```
BPCHARCMP(string1, string2)
```

Argumentos

string1

Uma string `CHAR` ou uma string `VARCHAR`.

string2

Uma string `CHAR` ou uma string `VARCHAR`.

Tipo de retorno

INTEGER

Exemplos

Os exemplos a seguir usam a tabela USERS do banco de dados de amostra TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para determinar se o primeiro nome de um usuário é alfabeticamente posterior ao sobrenome do usuário para as dez primeiras entradas na tabela USERS, use o exemplo a seguir. Para entradas em que a string de FIRSTNAME for alfabeticamente posterior à string de LASTNAME, a função retornará 1. Se LASTNAME for alfabeticamente posterior a FIRSTNAME, a função retornará `-1`.

```
SELECT userid, firstname, lastname, BPCHARCMP(firstname, lastname)
FROM users
ORDER BY 1, 2, 3, 4
LIMIT 10;

+--------+-----------+-----------+-----------+
| userid | firstname | lastname  | bpcharcmp |
+--------+-----------+-----------+-----------+
|      1 | Rafael    | Taylor    |        -1 |
|      2 | Vladimir  | Humphrey  |         1 |
|      3 | Lars      | Ratliff   |        -1 |
|      4 | Barry     | Roy       |        -1 |
|      5 | Reagan    | Hodge     |         1 |
|      6 | Victor    | Hernandez |         1 |
|      7 | Tamekah   | Juarez    |         1 |
|      8 | Colton    | Roy       |        -1 |
|      9 | Mufutau   | Watkins   |        -1 |
|     10 | Naida     | Calderon  |         1 |
+--------+-----------+-----------+-----------+
```

Para retornar todas as entradas na tabela USERS em que a função retorna `0`, use o exemplo a seguir. A função retorna `0` quando FIRSTNAME é idêntico a LASTNAME.

```
SELECT userid, firstname, lastname,
BPCHARCMP(firstname, lastname)
FROM users
WHERE BPCHARCMP(firstname, lastname)=0
ORDER BY 1, 2, 3, 4;

+--------+-----------+----------+-----------+
| userid | firstname | lastname | bpcharcmp |
+--------+-----------+----------+-----------+
```

```
|     62 | Chase     | Chase    |         0 |
|   4008 | Whitney   | Whitney  |         0 |
|  12516 | Graham    | Graham   |         0 |
|  13570 | Harper    | Harper   |         0 |
|  16712 | Cooper    | Cooper   |         0 |
|  18359 | Chase     | Chase    |         0 |
|  27530 | Bradley   | Bradley  |         0 |
|  31204 | Harding   | Harding  |         0 |
+--------+-----------+----------+-----------+
```

Função BTRIM

A função BTRIM apara uma string removendo os espaços em branco iniciais e finais ou removendo caracteres iniciais ou finais que correspondem a uma string opcional especificada.

Sintaxe

```
BTRIM(string [, trim_chars ] )
```

Argumentos

string

A string VARCHAR de entrada a ser cortada.

trim_chars

A string VARCHAR que contém os caracteres a serem correspondidos.

Tipo de retorno

A função BTRIM retorna uma string VARCHAR.

Exemplos

O seguinte exemplo apara espaços em branco inicias e finais da string `' abc '`:

```
select '     abc    ' as untrim, btrim('     abc    ') as trim;

untrim    | trim
----------+------
```

```
   abc    | abc
```

O exemplo a seguir remove a string `'xyz'` inicial e final da string `'xyzaxyzbxyzcxyz'`. As ocorrências inicial e final de `'xyz'` são removidas, mas as ocorrências internas da string não são removidas.

```
select 'xyzaxyzbxyzcxyz' as untrim,
btrim('xyzaxyzbxyzcxyz', 'xyz') as trim;

     untrim      |   trim
-----------------+-----------
 xyzaxyzbxyzcxyz | axyzbxyzc
```

O exemplo a seguir remove as partes iniciais e finais da string `'setuphistorycassettes'` que correspondem a qualquer um dos caracteres na lista trim_chars `'tes'`. Qualquer `t`, `e` ou `s` que ocorra antes que outro caractere que não esteja na lista trim_chars no início ou no final da string de entrada seja removido.

```
SELECT btrim('setuphistorycassettes', 'tes');

     btrim
-----------------
 uphistoryca
```

Função BTTEXT_PATTERN_CMP

Sinônimo da função BPCHARCMP.

Para mais detalhes, consulte Função BPCHARCMP.

Função CHAR_LENGTH

Sinônimo da função LEN.

Consulte Função LEN.

Função CHARACTER_LENGTH

Sinônimo da função LEN.

Consulte Função LEN.

Função CHARINDEX

Retorna a localização da substring especificada dentro de uma string.

Consulte Função POSITION e Função STRPOS para ver funções semelhantes.

Sintaxe

```
CHARINDEX( substring, string )
```

Argumentos

substring

A substring a procurar dentro da string.

string

A string ou coluna a ser procurada.

Tipo de retorno

INTEGER

A função CHARINDEX retorna um `INTEGER` correspondente à posição da substring (baseada em um, não baseada em zero). A posição é baseada no número de caracteres, e não bytes, de forma que caracteres multibyte são contados como caracteres simples. CHARINDEX retornará `0` se a substring não for localizada dentro da string.

Exemplos

Para retornar a posição da string `fish` na palavra `dog`, use o exemplo a seguir.

```
SELECT CHARINDEX('fish', 'dog');

+-----------+
| charindex |
+-----------+
```

```
|         0 |
+-----------+
```

Para retornar a posição da string `fish` na palavra `dogfish`, use o exemplo a seguir.

```
SELECT CHARINDEX('fish', 'dogfish');

+-----------+
| charindex |
+-----------+
|         4 |
+-----------+
```

O exemplo a seguir usa a tabela SALES do exemplo de banco de dados TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para retornar o número de transações de vendas distinta com uma comissão acima de 999,00 da tabela SALES, use o exemplo a seguir. Esse comando conta comissões maiores que 999,00 verificando se o decimal está acima de quatro casas a partir do início do valor da comissão.

```
SELECT DISTINCT CHARINDEX('.', commission), COUNT (CHARINDEX('.', commission))
FROM sales
WHERE CHARINDEX('.', commission) > 4
GROUP BY CHARINDEX('.', commission)
ORDER BY 1,2;

+-----------+-------+
| charindex | count |
+-----------+-------+
|         5 |   629 |
+-----------+-------+
```

Função CHR

A função CHR retorna o caractere que as corresponde ao valor de ponto do código ASCII especificado pelo parâmetro de entrada.

Sintaxe

```
CHR(number)
```

Argumento

Número da

O parâmetro de entrada é um `INTEGER` que representa um valor de ponto do código ASCII.

Tipo de retorno

CHAR

A função CHR retornará uma string `CHAR` se um caractere ASCII corresponder ao valor de entrada. Se o número de entrada não tiver nenhuma correspondência ASCII, a função retornará `NULL`.

Exemplos

Para retornar o caractere correspondente ao ponto 0 do código ASCII, use o exemplo a seguir. A função CHR retorna `NULL` para a entrada `0`.

```
SELECT CHR(0);

+-----+
| chr |
+-----+
|     |
+-----+
```

Para retornar o caractere que corresponde ao ponto 65 do código ASCII, use o exemplo a seguir.

```
SELECT CHR(65);

+-----+
| chr |
+-----+
| A   |
+-----+
```

Para retornar os nomes de eventos distintos que começam com um A maiúsculo (ponto de código ASCII 65), use o exemplo a seguir. O exemplo a seguir usa a tabela EVENT do banco de dados de amostra TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

```
SELECT DISTINCT eventname FROM event
WHERE SUBSTRING(eventname, 1, 1)=CHR(65) LIMIT 5;

+----------------------+
|       eventname      |
+----------------------+
| A Catered Affair     |
| As You Like It       |
| A Man For All Seasons |
| Alan Jackson         |
| Armando Manzanero    |
+----------------------+
```

Função COLLATE

A função COLATE substitui o agrupamento de uma coluna de string ou expressão.

Para obter informações sobre como criar tabelas usando o agrupamento de banco de dados, consulte CRIAR TABELA.

Para obter informações sobre como criar banco de dados usando o agrupamento de banco de dados, consulte CREATE DATABASE.

Sintaxe

```
COLLATE( string, 'case_sensitive' | 'cs' | 'case_insensitive' | 'ci');
```

Argumentos

string

Uma coluna de string ou expressão que você deseja substituir.

'case_sensitive' | 'cs' | 'case_insensitive' | 'ci'

Uma constante de string de um nome de agrupamento. O Amazon Redshift permite somente os seguintes valores para esse parâmetro:

- case_sensitive
- cs
- case_insensitive
- ci

case_sensitive e cs são intercambiáveis e geram os mesmos resultados. Da mesma forma, case_insensitive e ci são intercambiáveis e geram os mesmos resultados.

Tipo de retorno

A função COLLATE exibe VARCHAR, CHAR ou SUPER dependendo do primeiro tipo de expressão de entrada. Esta função altera apenas o agrupamento do primeiro argumento de entrada e não altera o seu valor de saída.

Exemplos

Para criar a tabela T e definir a col1 na tabela T como `case_sensitive`, use o exemplo a seguir.

```
CREATE TABLE T ( col1 Varchar(20) COLLATE case_sensitive );

INSERT INTO T VALUES ('john'),('JOHN');
```

Quando você executa a primeira consulta, o Amazon Redshift retorna apenas `john`. Depois que a função COLLATE é executada em col1, o agrupamento se torna `case_insensitive`. A segunda consulta retorna `john` e `JOHN`.

```
SELECT * FROM T WHERE col1 = 'john';

+------+
| col1 |
+------+
| john |
+------+

SELECT * FROM T WHERE COLLATE(col1, 'case_insensitive') = 'john';

+------+
| col1 |
+------+
| john |
| JOHN |
+------+
```

Para criar a tabela A e definir a col1 na tabela A como `case_insensitive`, use o exemplo a seguir.

```
CREATE TABLE A ( col1 Varchar(20) COLLATE case_insensitive );
```

```
INSERT INTO A VALUES ('john'),('JOHN');
```

Quando você executa a primeira consulta, o Amazon Redshift retorna `john` e `JOHN`. Depois que a função COLLATE é executada em col1, o agrupamento se torna `case_sensitive`. A segunda consulta retorna somente `john`.

```
SELECT * FROM A WHERE col1 = 'john';

+------+
| col1 |
+------+
| john |
| JOHN |
+------+

SELECT * FROM A WHERE COLLATE(col1, 'case_sensitive') = 'john';

+------+
| col1 |
+------+
| john |
+------+
```

Função CONCAT

A função CONCAT concatena duas expressões e retorna a expressão resultante. Para concatenar mais de duas strings, use funções CONCAT aninhadas. O operador de concatenação (`||`) entre duas expressões produz os mesmos resultados que a função CONCAT.

Sintaxe

```
CONCAT ( expression1, expression2 )
```

Argumentos

expression1, expression2

Os dois argumentos podem ser uma cadeia de caracteres de comprimento fixo, uma cadeia de caracteres de comprimento variável, uma expressão binária ou uma expressão que é avaliada para uma dessas entradas.

Tipo de retorno

CONCAT retorna uma expressão. O tipo de dados da expressão é o mesmo tipo dos argumentos de entrada.

Se as expressões de entrada forem de tipos diferentes, o Amazon Redshift tentará converter implicitamente o tipo de uma das expressões. Se os valores não puderem ser convertidos, será retornado um erro.

Observações de uso

- Tanto para a função CONCAT como para o operador de concatenação, se uma ou ambas as expressões forem nulas, o resultado da concatenação será null.

Exemplos

O seguinte exemplo concatena dois literais de caracteres:

```
SELECT CONCAT('December 25, ', '2008');

concat
-------------------
December 25, 2008
(1 row)
```

A seguinte consulta, usando o operador `||` em vez de CONCAT, produz o mesmo resultado:

```
SELECT 'December 25, '||'2008';

?column?
-------------------
December 25, 2008
(1 row)
```

O exemplo a seguir usa uma função CONCAT aninhada dentro de outra função CONCAT para concatenar três strings:

```
SELECT CONCAT('Thursday, ', CONCAT('December 25, ', '2008'));

concat
```

```
----------------------------
Thursday, December 25, 2008
(1 row)
```

Para concatenar colunas que possam conter NULLs, use Funções NVL e COALESCE, que retorna determinado valor quando encontra NULL. O seguinte exemplo usa NVL para retornar um 0 sempre que NULL for encontrado.

```
SELECT CONCAT(venuename, CONCAT(' seats ', NVL(venueseats, 0))) AS seating
FROM venue WHERE venuestate = 'NV' OR venuestate = 'NC'
ORDER BY 1
LIMIT 5;

seating
-----------------------------------
Ballys Hotel seats 0
Bank of America Stadium seats 73298
Bellagio Hotel seats 0
Caesars Palace seats 0
Harrahs Hotel seats 0
(5 rows)
```

A seguinte consulta concatena os valores CITY e STATE da tabela VENUE:

```
SELECT CONCAT(venuecity, venuestate)
FROM venue
WHERE venueseats > 75000
ORDER BY venueseats;

concat
-------------------
DenverCO
Kansas CityMO
East RutherfordNJ
LandoverMD
(4 rows)
```

A seguinte consulta usa funções CONCAT aninhadas. A consulta concatena os valores CITY e STATE da tabela VENUE, mas delimita a string resultante com uma vírgula e um espaço:

```
SELECT CONCAT(CONCAT(venuecity,', '),venuestate)
```

```
FROM venue
WHERE venueseats > 75000
ORDER BY venueseats;

concat
---------------------
Denver, CO
Kansas City, MO
East Rutherford, NJ
Landover, MD
(4 rows)
```

O exemplo a seguir concatena duas expressões binárias. Onde `abc` é um valor binário (com uma representação hexadecimal de `616263`) e `def` é um valor binário (com representação hexadecimal de `646566`). O resultado é exibido automaticamente como a representação hexadecimal do valor binário.

```
SELECT CONCAT('abc'::VARBYTE, 'def'::VARBYTE);

concat
-------------------
616263646566
```

Função CRC32

CRC32 é uma função usada para detecção de erros. A função usa um algoritmo CRC32 para detectar alterações entre os dados de origem e destino. A função CRC32 converte uma string de comprimento variável em uma string de 8 caracteres que é uma representação de texto de valor hexadecimal de uma sequência binária de 32 bits. Para detectar alterações entre os dados de origem e de destino, use a função CRC32 nos dados de origem e armazene a saída. Depois, use a função CRC32 nos dados de destino e compare essa saída com a dos dados de origem. As saídas serão as mesmas se os dados não tiverem sido modificados e serão diferentes se os dados forem modificados.

Sintaxe

```
CRC32(string)
```

Argumentos

string

Uma string CHAR, uma string VARCHAR ou uma expressão que é avaliada implicitamente como um tipo CHAR ou VARCHAR.

Tipo de retorno

A função CRC32 retorna uma string 8 caracteres que é uma representação de texto de valor hexadecimal de uma sequência binária de 32 bits. A função CRC32 do Amazon Redshift é baseada em no polinomial CRC-32C.

Exemplos

Para mostrar o valor de 8 bits da string `Amazon Redshift`.

```
SELECT CRC32('Amazon Redshift');

+----------+
|  crc32   |
+----------+
| f2726906 |
+----------+
```

Função DIFFERENCE

A função DIFFERENCE compara os códigos American Soundex de duas strings. A função retorna um `INTEGER` para indicar o número de caracteres correspondentes entre os códigos Soundex.

Um código Soundex é uma string com quatro caracteres. Um código Soundex representa como uma palavra soa em vez de como ela é escrita. Por exemplo, `Smith` e `Smyth` têm o mesmo código Soundex.

Sintaxe

```
DIFFERENCE(string1, string2)
```

Argumentos

string1

Uma string CHAR, uma string VARCHAR ou uma expressão que é avaliada implicitamente como um tipo CHAR ou VARCHAR.

string2

Uma string CHAR, uma string VARCHAR ou uma expressão que é avaliada implicitamente como um tipo CHAR ou VARCHAR.

Tipo de retorno

INTEGER

A função DIFFERENCE retorna um valor INTEGER de 0 a 4 que conta o número de caracteres correspondentes nos códigos American Soundex das duas strings. Um código Soundex tem quatro caracteres, então a função DIFFERENCE retornará 4 quando todos os quatro caracteres dos valores do código American Soundex das strings forem iguais. DIFFERENCE retornará 0 se uma das duas strings estiver vazia. A função retornará 1 se nenhuma string contiver caracteres válidos. A função DIFFERENCE converte apenas caracteres ASCII em letras minúsculas ou maiúsculas em inglês, incluindo a–z e A–Z. A DIFFERENCE ignora outros caracteres.

Exemplos

Para comparar os valores do Soundex das strings % e @, use o exemplo a seguir. A função retornará 1 porque nenhuma string contém caracteres válidos.

```
SELECT DIFFERENCE('%', '@');

+------------+
| difference |
+------------+
|          1 |
+------------+
```

Para comparar os valores do Soundex de Amazon e uma string vazia, use o exemplo a seguir. A função retornará 0 porque uma das duas strings está vazia.

```
SELECT DIFFERENCE('Amazon', '');
```

```
+------------+
| difference |
+------------+
|          0 |
+------------+
```

Para comparar os valores do Soundex das strings `Amazon` e `Ama`, use o exemplo a seguir. A função retornará 2 porque dois caracteres dos valores do Soundex das strings são iguais.

```
SELECT DIFFERENCE('Amazon', 'Ama');

+------------+
| difference |
+------------+
|          2 |
+------------+
```

Para comparar os valores do Soundex das strings `Amazon` e `+-*/%Amazon`, use o exemplo a seguir. A função retornará 4 porque todos os quatro caracteres dos valores do Soundex das strings são iguais. Observe que a função ignora os caracteres inválidos `+-*/%` na segunda string.

```
SELECT DIFFERENCE('Amazon', '+-*/%Amazon');

+------------+
| difference |
+------------+
|          4 |
+------------+
```

Para comparar os valores do Soundex das strings `AC/DC` e `Ay See Dee See`, use o exemplo a seguir. A função retornará 4 porque todos os quatro caracteres dos valores do Soundex das strings são iguais.

```
SELECT DIFFERENCE('AC/DC', 'Ay See Dee See');

+------------+
| difference |
+------------+
|          4 |
+------------+
```

Função INITCAP

Capitaliza a primeira letra de cada palavra de uma string especificada. INITCAP é compatível com caracteres UTF-8 multibyte, até o máximo de quatro bytes por caractere.

Sintaxe

```
INITCAP(string)
```

Argumento

string

Uma string CHAR, uma string VARCHAR ou uma expressão que é avaliada implicitamente como um tipo CHAR ou VARCHAR.

Tipo de retorno

VARCHAR

Observações de uso

A função INITCAP transforma em maiúscula a primeira letra de cada palavra em uma string e quaisquer letras subsequentes são transformadas (ou deixadas) em minúsculo. Portanto, é importante compreender quais caracteres (além de caracteres de espaço) funcionam como separadores de palavras. Um caractere separador de palavras é qualquer caractere não alfanumérico, incluindo marcas de pontuação, símbolos e caracteres de controle. Todos os seguintes caracteres são separadores de palavras:

```
! " # $ % & ' ( ) * + , - . / : ; < = > ? @ [ \ ] ^ _ ` { | } ~
```

Guias, caracteres de nova linha, alimentações de formulário e retornos de carro também são separadores de palavras.

Exemplos

Os exemplos a seguir usam dados da tabela CATEGORY e USERS do banco de dados de amostra TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para capitalizar as iniciais de cada palavra na coluna CATDESC, use o exemplo a seguir.

```
SELECT catid, catdesc, INITCAP(catdesc)
FROM category
ORDER BY 1, 2, 3;

+-------+--------------------------------------------
+-----------------------------------------+
| catid |                  catdesc                   |                  initcap
          |
+-------+--------------------------------------------
+-----------------------------------------+
|     1 | Major League Baseball                      | Major League Baseball
          |
|     2 | National Hockey League                     | National Hockey League
          |
|     3 | National Football League                   | National Football League
          |
|     4 | National Basketball Association            | National Basketball Association
          |
|     5 | Major League Soccer                        | Major League Soccer
          |
|     6 | Musical theatre                            | Musical Theatre
          |
|     7 | All non-musical theatre                    | All Non-Musical Theatre
          |
|     8 | All opera and light opera                  | All Opera And Light Opera
          |
|     9 | All rock and pop music concerts            | All Rock And Pop Music Concerts
          |
|    10 | All jazz singers and bands                 | All Jazz Singers And Bands
          |
|    11 | All symphony, concerto, and choir concerts | All Symphony, Concerto, And
 Choir Concerts |
+-------+--------------------------------------------
+-----------------------------------------+
```

Para mostrar que a função INITCAP não preserva caracteres maiúsculos quando eles não começam palavras, use o exemplo a seguir. Por exemplo, a string `MLB` torna-se `Mlb`.

```
SELECT INITCAP(catname)
FROM category
ORDER BY catname;

+-----------+
```

```
|  initcap  |
+-----------+
| Classical |
| Jazz      |
| Mlb       |
| Mls       |
| Musicals  |
| Nba       |
| Nfl       |
| Nhl       |
| Opera     |
| Plays     |
| Pop       |
+-----------+
```

Para mostrar que caracteres não alfanuméricos, exceto espaços, funcionam como separadores de palavras, use o exemplo a seguir. Várias letras em cada string serão maiúsculas.

```
SELECT email, INITCAP(email)
FROM users
ORDER BY userid DESC LIMIT 5;

+-----------------------------------+-----------------------------------+
|               email               |              initcap              |
+-----------------------------------+-----------------------------------+
| urna.Ut@egetdictumplacerat.edu    | Urna.Ut@Egetdictumplacerat.Edu    |
| nibh.enim@egestas.ca              | Nibh.Enim@Egestas.Ca              |
| in@Donecat.ca                     | In@Donecat.Ca                     |
| sodales@blanditviverraDonec.ca    | Sodales@Blanditviverradonec.Ca    |
| sociis.natoque.penatibus@vitae.org | Sociis.Natoque.Penatibus@Vitae.Org |
+-----------------------------------+-----------------------------------+
```

Funções LEFT e RIGHT

Essas funções retornam o número especificado de caracteres mais à esquerda ou mais à direita de uma string de caracteres.

O número é baseado no número de caracteres, e não bytes, de forma que caracteres multibyte são contados como caracteres simples.

Sintaxe

```
LEFT( string,  integer )
```

```
RIGHT( string,  integer )
```

Argumentos

string

Uma string CHAR, uma string VARCHAR ou qualquer expressão que seja avaliada como uma string CHAR ou VARCHAR.

integer

Um inteiro positivo.

Tipo de retorno

VARCHAR

Exemplos

O exemplo a seguir usa dados da tabela EVENT do banco de dados de amostra TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para retornar os cinco caracteres mais à esquerda e os cinco caracteres mais à direita dos nomes de eventos com IDs entre 1000 e 1005, use o exemplo a seguir.

```
SELECT eventid, eventname,
LEFT(eventname,5) AS left_5,
RIGHT(eventname,5) AS right_5
FROM event
WHERE eventid BETWEEN 1000 AND 1005
ORDER BY 1;

+---------+----------------+--------+---------+
| eventid |   eventname    | left_5 | right_5 |
+---------+----------------+--------+---------+
|    1000 | Gypsy          | Gypsy  | Gypsy   |
|    1001 | Chicago        | Chica  | icago   |
|    1002 | The King and I | The K  | and I   |
|    1003 | Pal Joey       | Pal J  | Joey    |
|    1004 | Grease         | Greas  | rease   |
|    1005 | Chicago        | Chica  | icago   |
+---------+----------------+--------+---------+
```

Função LEN

Retorna o tamanho da string especificada como número de caracteres.

Sintaxe

LEN é um sinônimo de Função LENGTH, Função CHAR_LENGTH, Função CHARACTER_LENGTH e Função TEXTLEN.

```
LEN(expression)
```

Argumento

expressão

Uma string CHAR, uma string VARCHAR, uma expressão VARBYTE ou uma expressão que é avaliada implicitamente como um tipo CHAR, VARCHAR ou VARBYTE.

Tipo de retorno

INTEGER

A função LEN retorna um inteiro indicando o número de caracteres em string de entrada.

Se a string de entrada for uma cadeia de caracteres, a função LEN retornará o número real de caracteres em strings multibyte, e não o número de bytes. Por exemplo, uma coluna VARCHAR(12) deve armazenar três caracteres chineses de quatro bytes. A função LEN retornará 3 para essa mesma string. Para obter o tamanho de uma string em bytes, use a função OCTET_LENGTH.

Observações de uso

Se a expressão for uma string CHAR, os espaços finais não serão contados.

Se a expressão for uma string VARCHAR, os espaços finais serão contados.

Exemplos

Para retornar o número de bytes e o número de caracteres na string `français`, use o exemplo a seguir.

```
SELECT OCTET_LENGTH('français'),
LEN('français');

+--------------+-----+
| octet_length | len |
+--------------+-----+
|            9 |   8 |
+--------------+-----+
```

Para retornar o número de bytes e o número de caracteres na string `français` sem usar a função OCTET_LENGTH, use o exemplo a seguir. Para obter mais informações, consulte a Função CAST.

```
SELECT LEN(CAST('français' AS VARBYTE)) as bytes, LEN('français');

+-------+-----+
| bytes | len |
+-------+-----+
|     9 |   8 |
+-------+-----+
```

Para retornar o número de caracteres nas strings `cat` sem espaços finais, `cat` com três espaços finais, `cat` com três espaços finais lançados como um CHAR de comprimento 6 e `cat` com três espaços finais lançados como um VARCHAR de comprimento 6, use o exemplo a seguir. Observe que a função não conta os espaços finais para strings CHAR, mas conta esses espaços para strings VARCHAR.

```
SELECT LEN('cat'), LEN('cat   '), LEN(CAST('cat   ' AS CHAR(6))) AS len_char,
 LEN(CAST('cat   ' AS VARCHAR(6))) AS len_varchar;

+-----+-----+----------+-------------+
| len | len | len_char | len_varchar |
+-----+-----+----------+-------------+
|   3 |   6 |        3 |           6 |
+-----+-----+----------+-------------+
```

O exemplo a seguir usa dados da tabela VENUE do banco de dados de amostra TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para retornar os dez nomes de locais mais longos na tabela VENUE, use o exemplo a seguir.

```
SELECT venuename, LEN(venuename)
```

```
FROM venue
ORDER BY 2 DESC, 1
LIMIT 10;

+----------------------------------------+-----+
|               venuename                | len |
+----------------------------------------+-----+
| Saratoga Springs Performing Arts Center |  39 |
| Lincoln Center for the Performing Arts  |  38 |
| Nassau Veterans Memorial Coliseum       |  33 |
| Jacksonville Municipal Stadium          |  30 |
| Rangers BallPark in Arlington           |  29 |
| University of Phoenix Stadium           |  29 |
| Circle in the Square Theatre            |  28 |
| Hubert H. Humphrey Metrodome            |  28 |
| Oriole Park at Camden Yards             |  27 |
| Dick's Sporting Goods Park              |  26 |
+----------------------------------------+-----+
```

Função LENGTH

Sinônimo da função LEN.

Consulte Função LEN.

Função LOWER

Converte uma string em letras minúsculas. LOWER é compatível com caracteres UTF-8 multibyte, até o máximo de quatro bytes por caractere.

Sintaxe

```
LOWER(string)
```

Argumento

string

Uma string VARCHAR ou uma expressão que avalia como o tipo VARCHAR.

Tipo de retorno

string

A função LOWER retorna uma string do mesmo tipo de dados que a string de entrada. Por exemplo, se a entrada for uma string CHAR, a função retornará uma string CHAR.

Exemplos

O exemplo a seguir usa dados da tabela CATEGORY do banco de dados de amostra TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para converter as strings VARCHAR na coluna CATNAME em letras minúsculas, use o exemplo a seguir.

```
SELECT catname, LOWER(catname) FROM category ORDER BY 1,2;

+-----------+-----------+
|  catname  |   lower   |
+-----------+-----------+
| Classical | classical |
| Jazz      | jazz      |
| MLB       | mlb       |
| MLS       | mls       |
| Musicals  | musicals  |
| NBA       | nba       |
| NFL       | nfl       |
| NHL       | nhl       |
| Opera     | opera     |
| Plays     | plays     |
| Pop       | pop       |
+-----------+-----------+
```

Funções LPAD e RPAD

Essas funções inserem caracteres no início ou final de uma string com base em um comprimento especificado.

Sintaxe

```
LPAD(string1, length, [ string2 ])
```

```
RPAD(string1, length, [ string2 ])
```

Argumentos

string1

Uma string CHAR, uma string VARCHAR ou uma expressão que é avaliada implicitamente como um tipo CHAR ou VARCHAR.

length

Um inteiro que define o comprimento dos resultados da função. O comprimento de uma string é baseado no número de caracteres, e não bytes, de forma que caracteres multibyte são contados como caracteres simples. Se string1 for mais longa que o comprimento especificado, ela será truncada (à direita). Se length for zero ou um número negativo, o resultado da função será uma string vazia.

string2

(Opcional) Um ou mais caracteres inseridos no início ou no fim da string1. Se esse argumento não é especificado, são usados espaços.

Tipo de retorno

VARCHAR

Exemplos

Os exemplos a seguir usam dados da tabela EVENT do banco de dados de amostra TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para truncar um conjunto específico de nomes de eventos para 20 caracteres e inserir espaços no início dos nomes mais curtos, use o exemplo a seguir.

```
SELECT LPAD(eventname, 20) FROM event
WHERE eventid BETWEEN 1 AND 5 ORDER BY 1;

+----------------------+
|         lpad         |
+----------------------+
|              Salome |
```

```
|        Il Trovatore |
|       Boris Godunov |
|     Gotterdammerung |
|La Cenerentola (Cind |
+---------------------+
```

Para truncar o mesmo conjunto de nomes de eventos para 20 caracteres, mas inserir `0123456789` no início dos nomes mais curtos, use o exemplo a seguir.

```
SELECT RPAD(eventname, 20,'0123456789') FROM event
WHERE eventid BETWEEN 1 AND 5 ORDER BY 1;

+----------------------+
|         rpad         |
+----------------------+
| Boris Godunov0123456 |
| Gotterdammerung01234 |
| Il Trovatore01234567 |
| La Cenerentola (Cind |
| Salome01234567890123 |
+----------------------+
```

Função LTRIM

Corta caracteres do início de uma string. Remove a string mais longa que contém somente caracteres que estão na lista de caracteres de corte. O corte é concluído quando nenhum caractere de corte aparece na string de entrada.

Sintaxe

```
LTRIM( string [, trim_chars] )
```

Argumentos

string

Uma coluna, expressão ou literal de string a ser cortado.

trim_chars

Uma coluna, expressão ou literal de string que representa os caracteres a serem cortados do começo da string. Se não for especificado, um espaço será usado como caractere de corte.

Tipo de retorno

A função LTRIM retorna uma string no mesmo tipo de dado que a string de entrada (CHAR ou VARCHAR).

Exemplos

O exemplo a seguir corta o ano da coluna `listime`. Os caracteres de corte no literal de string `'2008-'` indicam os caracteres a serem cortados da esquerda. Se você usar os caracteres de corte `'028-'`, obterá o mesmo resultado.

```
select listid, listtime, ltrim(listtime, '2008-')
from listing
order by 1, 2, 3
limit 10;

listid |      listtime       |     ltrim
-------+---------------------+---------------
     1 | 2008-01-24 06:43:29 | 1-24 06:43:29
     2 | 2008-03-05 12:25:29 | 3-05 12:25:29
     3 | 2008-11-01 07:35:33 | 11-01 07:35:33
     4 | 2008-05-24 01:18:37 | 5-24 01:18:37
     5 | 2008-05-17 02:29:11 | 5-17 02:29:11
     6 | 2008-08-15 02:08:13 | 15 02:08:13
     7 | 2008-11-15 09:38:15 | 11-15 09:38:15
     8 | 2008-11-09 05:07:30 | 11-09 05:07:30
     9 | 2008-09-09 08:03:36 | 9-09 08:03:36
    10 | 2008-06-17 09:44:54 | 6-17 09:44:54
```

LTRIM remove qualquer um dos caracteres em trim_chars quando eles aparecem no início da string. O seguinte exemplo apara os caracteres “C”, “D” e “G” quando eles aparecem no início de VENUENAME, que é uma coluna VARCHAR.

```
select venueid, venuename, ltrim(venuename, 'CDG')
from venue
where venuename like '%Park'
order by 2
limit 7;

venueid | venuename                  | btrim
--------+----------------------------+--------------------------
    121 | ATT Park                   | ATT Park
```

```
    109 | Citizens Bank Park         | itizens Bank Park
    102 | Comerica Park              | omerica Park
      9 | Dick's Sporting Goods Park | ick's Sporting Goods Park
     97 | Fenway Park                | Fenway Park
    112 | Great American Ball Park   | reat American Ball Park
    114 | Miller Park                | Miller Park
```

O exemplo a seguir usa o caractere de corte 2 que é recuperado da coluna `venueid`.

```
select ltrim('2008-01-24 06:43:29', venueid)
from venue where venueid=2;

ltrim
------------------
008-01-24 06:43:29
```

O exemplo a seguir não corta nenhum caractere porque 2 é encontrado antes do caractere de corte `'0'`.

```
select ltrim('2008-01-24 06:43:29', '0');

ltrim
-------------------
2008-01-24 06:43:29
```

O exemplo a seguir usa o caractere de corte de espaço padrão e corta os dois espaços do início da string.

```
select ltrim('  2008-01-24 06:43:29');

ltrim
-------------------
2008-01-24 06:43:29
```

Função OCTETINDEX

A função OCTETINDEX retorna a localização de uma substring dentro de uma string como um número de bytes.

Sintaxe

```
OCTETINDEX(substring, string)
```

Argumentos

substring

Uma string CHAR, uma string VARCHAR ou uma expressão que é avaliada implicitamente como um tipo CHAR ou VARCHAR.

string

Uma string CHAR, uma string VARCHAR ou uma expressão que é avaliada implicitamente como um tipo CHAR ou VARCHAR.

Tipo de retorno

INTEGER

A função OCTETINDEX exibe um valor INTEGER correspondente à posição da substring dentro da string como um número de bytes, em que o primeiro caractere na string é contado como 1. Se a string não contiver caracteres multibyte, o resultado será igual ao resultado da função CHARINDEX. Se a string não contiver a substring, a função retornará 0. Se a substring estiver vazia, a função retornará 1.

Exemplos

Para exibir a posição da substring q na string Amazon Redshift, use o exemplo a seguir. Esse exemplo retornará 0 porque a substring não está na string.

```
SELECT OCTETINDEX('q', 'Amazon Redshift');

+------------+
| octetindex |
+------------+
|          0 |
+------------+
```

Para exibir a posição de uma substring vazia na string Amazon Redshift, use o exemplo a seguir. Este exemplo retornará 1 porque a substring está vazia.

```
SELECT OCTETINDEX('', 'Amazon Redshift');

+------------+
| octetindex |
+------------+
|          1 |
+------------+
```

Para exibir a posição da substring `Redshift` na string `Amazon Redshift`, use o exemplo a seguir. Esse exemplo retorna 8 porque a substring começa no oitavo byte da string.

```
SELECT OCTETINDEX('Redshift', 'Amazon Redshift');

+------------+
| octetindex |
+------------+
|          8 |
+------------+
```

Para exibir a posição da substring `Redshift` na string `Amazon Redshift`, use o exemplo a seguir. O exemplo a seguir retorna 21 porque os primeiros seis caracteres da string são caracteres de byte duplo.

```
SELECT OCTETINDEX('Redshift', 'Άμαζον Amazon Redshift');

+------------+
| octetindex |
+------------+
|         21 |
+------------+
```

Função OCTET_LENGTH

Retorna o tamanho da string especificada como número de bytes.

Sintaxe

```
OCTET_LENGTH(expression)
```

Argumento

expressão

Uma string CHAR, uma string VARCHAR, uma expressão VARBYTE ou uma expressão que é avaliada implicitamente como um tipo CHAR, VARCHAR ou VARBYTE.

Tipo de retorno

INTEGER

A função OCTET_LENGTH retorna um número inteiro indicando o número de bytes na string de entrada.

Se a string de entrada for uma cadeia de caracteres, a função LEN retornará o número real de caracteres em strings multibyte, e não o número de bytes. Por exemplo, uma coluna VARCHAR(12) deve armazenar três caracteres chineses de quatro bytes. A função OCTET_LENGTH retornará 12 para essa string, e a função LEN retornará 3 para essa mesma string.

Observações de uso

Se a expressão for uma string CHAR, a função retornará o comprimento da string CHAR. Por exemplo, a saída de uma entrada CHAR(6) é CHAR(6).

Se a expressão for uma string VARCHAR, os espaços finais serão contados.

Exemplos

Para retornar o número de bytes quando a string `francais` com três espaços finais é lançada em um tipo CHAR e VARCHAR, use o exemplo a seguir. Para obter mais informações, consulte a Função CAST.

```
SELECT OCTET_LENGTH(CAST('francais   ' AS CHAR(15))) AS octet_length_char,
 OCTET_LENGTH(CAST('francais   ' AS VARCHAR(15))) AS octet_length_varchar;

+------------------+----------------------+
| octet_length_char | octet_length_varchar |
+------------------+----------------------+
|               15 |                   11 |
```

```
+------------------+---------------------+
```

Para retornar o número de bytes e o número de caracteres na string `français`, use o exemplo a seguir.

```
SELECT OCTET_LENGTH('français'), LEN('français');

+--------------+-----+
| octet_length | len |
+--------------+-----+
|            9 |   8 |
+--------------+-----+
```

Para retornar o número de bytes quando a string `français` é transmitida como um VARBYTE, use o exemplo a seguir.

```
SELECT OCTET_LENGTH(CAST('français' AS VARBYTE));

+--------------+
| octet_length |
+--------------+
|            9 |
+--------------+
```

Função POSITION

Retorna a localização da substring especificada dentro de uma string.

Consulte Função CHARINDEX e Função STRPOS para ver funções semelhantes.

Sintaxe

```
POSITION(substring IN string )
```

Argumentos

substring

A substring a procurar dentro da string.

string

A string ou coluna a ser procurada.

Tipo de retorno

A função POSITION retorna um `INTEGER` correspondente à posição da substring (baseada em 1, não baseada em zero). A posição é baseada no número de caracteres, e não bytes, de forma que caracteres multibyte são contados como caracteres simples. POSITION retornará `0` se a substring não for localizada dentro da string.

Exemplos

Para retornar a posição da string `fish` na palavra `dog`, use o exemplo a seguir.

```
SELECT POSITION('fish' IN 'dog');

+-----------+
|  position |
+-----------+
|         0 |
+-----------+
```

Para retornar a posição da string `fish` na palavra `dogfish`, use o exemplo a seguir.

```
SELECT POSITION('fish' IN 'dogfish');

+-----------+
|  position |
+-----------+
|         4 |
+-----------+
```

O exemplo a seguir usa a tabela SALES do exemplo de banco de dados TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para retornar o número de transações de vendas distinta com uma comissão acima de 999,00 da tabela SALES, use o exemplo a seguir. Esse comando conta comissões maiores que 999,00 verificando se o decimal está acima de quatro casas a partir do início do valor da comissão.

```
SELECT DISTINCT POSITION('.' IN commission), COUNT (POSITION('.' IN commission))
FROM sales
WHERE POSITION('.' IN commission) > 4
GROUP BY POSITION('.' IN commission)
ORDER BY 1,2;
```

```
+-----------+-------+
|  position | count |
+-----------+-------+
|         5 |   629 |
+-----------+-------+
```

Função QUOTE_IDENT

A função QUOTE_IDENT retorna a string especificada como uma string entre aspas duplas iniciais e uma aspa dupla final. A saída da função pode ser usada como um identificador em uma instrução SQL. A função duplica apropriadamente quaisquer aspas duplas incorporadas.

QUOTE_IDENT adiciona aspas duplas apenas quando necessário para criar um identificador válido, quando a string contém caracteres não identificadores ou seria dobrada em minúsculas. Para sempre retornar uma sequência entre aspas simples, use QUOTE_LITERAL.

Sintaxe

```
QUOTE_IDENT(string)
```

Argumento

string

Uma string CHAR ou VARCHAR.

Tipo de retorno

A função QUOTE_IDENT retorna o mesmo tipo de string da entrada string.

Exemplos

Para retornar a string "CAT" com aspas duplas, use o exemplo a seguir.

```
SELECT QUOTE_IDENT('"CAT"');

+-------------+
| quote_ident |
+-------------+
| """CAT"""   |
+-------------+
```

O exemplo a seguir usa dados da tabela CATEGORY do banco de dados de amostra TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para retornar a coluna CATNAME entre aspas use o exemplo a seguir.

```
SELECT catid, QUOTE_IDENT(catname)
FROM category
ORDER BY 1,2;

+-------+-------------+
| catid | quote_ident |
+-------+-------------+
|     1 | "MLB"       |
|     2 | "NHL"       |
|     3 | "NFL"       |
|     4 | "NBA"       |
|     5 | "MLS"       |
|     6 | "Musicals"  |
|     7 | "Plays"     |
|     8 | "Opera"     |
|     9 | "Pop"       |
|    10 | "Jazz"      |
|    11 | "Classical" |
+-------+-------------+
```

Função QUOTE_LITERAL

A função QUOTE_LITERAL retorna a string especificada como uma string entre aspas únicas para que ela possa ser usada como um literal de string em uma instrução SQL. Se o parâmetro de entrada for um número, QUOTE_LITERAL o tratará como uma string. Duplica apropriadamente quaisquer aspas simples e barras invertidas incorporadas.

Sintaxe

```
QUOTE_LITERAL(string)
```

Argumento

string

Uma string CHAR ou VARCHAR.

Tipo de retorno

A função QUOTE_LITERAL retorna uma string CHAR ou VARCHAR do mesmo tipo de dados que a entrada string.

Exemplos

Para retornar a string ''CAT'' com aspas SIMPLES, use o exemplo a seguir.

```
SELECT QUOTE_LITERAL('''CAT''');

+---------------+
| quote_literal |
+---------------+
| '''CAT'''     |
+---------------+
```

Os exemplos a seguir usam dados da tabela CATEGORY do banco de dados de amostra TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para retornar a coluna CATNAME entre aspas únicas, use o exemplo a seguir.

```
SELECT catid, QUOTE_LITERAL(catname)
FROM category
ORDER BY 1,2;

+-------+---------------+
| catid | quote_literal |
+-------+---------------+
|     1 | 'MLB'         |
|     2 | 'NHL'         |
|     3 | 'NFL'         |
|     4 | 'NBA'         |
|     5 | 'MLS'         |
|     6 | 'Musicals'    |
|     7 | 'Plays'       |
|     8 | 'Opera'       |
|     9 | 'Pop'         |
|    10 | 'Jazz'        |
|    11 | 'Classical'   |
+-------+---------------+
```

Para retornar a coluna CATID entre aspas únicas, use o exemplo a seguir.

```
SELECT QUOTE_LITERAL(catid), catname
FROM category
ORDER BY 1,2;

+---------------+-----------+
| quote_literal |  catname  |
+---------------+-----------+
| '1'           | MLB       |
| '10'          | Jazz      |
| '11'          | Classical |
| '2'           | NHL       |
| '3'           | NFL       |
| '4'           | NBA       |
| '5'           | MLS       |
| '6'           | Musicals  |
| '7'           | Plays     |
| '8'           | Opera     |
| '9'           | Pop       |
+---------------+-----------+
```

Função REGEXP_COUNT

Pesquisa uma string quanto a um padrão de expressão regular e retorna um inteiro que indica o número de vezes que o padrão especificado ocorre na string. Se nenhuma correspondência for encontrada, a função retornará `0`. Para ter mais informações sobre expressões regulares, consulte Operadores POSIX e Expressão regular na Wikipédia.

Sintaxe

```
REGEXP_COUNT( source_string, pattern [, position [, parameters ] ] )
```

Argumentos

source_string

Uma string CHAR ou VARCHAR.

Padrão

Um literal de string UTF-8 que representa um padrão de expressão regular. Para obter mais informações, consulte Operadores POSIX.

position

(Opcional) Um INTEGER positivo que indica a posição em source_string para começar a pesquisar. A posição é baseada no número de caracteres, e não bytes, de forma que caracteres multibyte são contados como caracteres simples. O padrão é 1. Se a posição for menor que 1, a pesquisa começará no primeiro caractere da source_string. Se a posição for maior que o número de caracteres na source_string, o resultado será 0.

parameters

(Opcional) Uma ou mais literais de string que indicam como a função corresponde ao padrão. Os valores possíveis são os seguintes:

- c – Executa a correspondência diferenciando maiúsculas e minúsculas. O padrão é usar a correspondência diferenciando maiúsculas e minúsculas.
- i – Executa a correspondência sem diferenciar maiúsculas de minúsculas.
- p — Interpreta o padrão com o dialeto de expressão regular compatível com Perl (PCRE - Perl Compatible Regular Expression). Para ter mais informações sobre PCRE, consulte Perl Compatible Regular Expressions na Wikipédia.

Tipo de retorno

INTEGER

Exemplos

Para contar o número de vezes em que uma sequência de três letras ocorre, use o exemplo a seguir.

```
SELECT REGEXP_COUNT('abcdefghijklmnopqrstuvwxyz', '[a-z]{3}');

+--------------+
| regexp_count |
+--------------+
|            8 |
+--------------+
```

Para contar as ocorrências da string FOX usando correspondência sem diferenciar letras maiúsculas de minúsculas, use o exemplo a seguir.

```
SELECT REGEXP_COUNT('the fox', 'FOX', 1, 'i');
```

```
+--------------+
| regexp_count |
+--------------+
|            1 |
+--------------+
```

Para usar um padrão escrito no dialeto PCRE para localizar palavras contendo pelo menos um número e uma letra minúscula, use o exemplo a seguir. O exemplo usa o operador ?=, que tem uma conotação específica look-ahead em PCRE. Este exemplo conta o número de ocorrências de tais palavras, com correspondência diferenciando maiúsculas de minúsculas.

```
SELECT REGEXP_COUNT('passwd7 plain A1234 a1234', '(?=[^ ]*[a-z])(?=[^ ]*[0-9])[^ ]+',
 1, 'p');

+--------------+
| regexp_count |
+--------------+
|            2 |
+--------------+
```

Para usar um padrão escrito no dialeto PCRE para localizar palavras contendo pelo menos um número e uma letra minúscula, use o exemplo a seguir. Ele usa o operador ?=, que tem uma conotação específica em PCRE. Este exemplo conta o número de ocorrências de tais palavras, mas difere do exemplo anterior na medida em que usa correspondência sem diferenciar maiúsculas dee minúsculas.

```
SELECT REGEXP_COUNT('passwd7 plain A1234 a1234', '(?=[^ ]*[a-z])(?=[^ ]*[0-9])[^ ]+',
 1, 'ip');

+--------------+
| regexp_count |
+--------------+
|            3 |
+--------------+
```

O exemplo a seguir usa dados da tabela USERS do banco de dados de amostra TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para contar o número de vezes que nome de domínio de nível superior é `org` ou `edu`, use o exemplo a seguir.

```
SELECT email, REGEXP_COUNT(email,'@[^.]*\.(org|edu)') FROM users
ORDER BY userid LIMIT 4;

+-----------------------------------------------+--------------+
|                     email                     | regexp_count |
+-----------------------------------------------+--------------+
| Etiam.laoreet.libero@sodalesMaurisblandit.edu |            1 |
| Suspendisse.tristique@nonnisiAenean.edu       |            1 |
| amet.faucibus.ut@condimentumegetvolutpat.ca   |            0 |
| sed@lacusUtnec.ca                             |            0 |
+-----------------------------------------------+--------------+
```

Função REGEXP_INSTR

Pesquisa um padrão de expressão regular em uma sequência e retorna um inteiro que indica a posição inicial ou final da subsequência correspondente. Se nenhuma correspondência for encontrada, a função retornará `0`. REGEXP_INSTR é semelhante à função POSITION, mas, com REGEXP_INSTR, é possível pesquisar uma string em busca de um padrão de expressão regular. Para ter mais informações sobre expressões regulares, consulte Operadores POSIX e Expressão regular na Wikipédia.

Sintaxe

```
REGEXP_INSTR( source_string, pattern [, position [, occurrence] [, option [, parameters
 ] ] ] ] )
```

Argumentos

source_string

Uma expressão de string, tal como um nome de coluna, a ser procurada.

Padrão

Um literal de string UTF-8 que representa um padrão de expressão regular. Para obter mais informações, consulte Operadores POSIX.

position

(Opcional) Um `INTEGER` positivo que indica a posição em source_string para começar a pesquisar. A posição é baseada no número de caracteres, e não bytes, de forma que caracteres multibyte são contados como caracteres simples. O padrão é 1. Se a posição for menor que 1, a

pesquisa começará no primeiro caractere da source_string. Se a posição for maior que o número de caracteres na source_string, o resultado será `0`.

occurrence

(Opcional) Um `INTEGER` positivo que indica qual ocorrência do padrão usar. REGEXP_INSTR ignora as primeiras correspondências de *`occurrence`*`-1`. O padrão é 1. Se a ocorrência for menor que 1 ou maior que o número de caracteres em source_string, a pesquisa será ignorada e o resultado será `0`.

option

(Opcional) Um valor que indica se retornar a posição do primeiro caractere da correspondência (`0`) ou a posição do primeiro caractere seguinte ao final da correspondência (1). Um valor diferente de zero é o mesmo que 1. O valor padrão é `0`.

parameters

(Opcional) Uma ou mais literais de string que indicam como a função corresponde ao padrão. Os valores possíveis são os seguintes:

- c – Executa a correspondência diferenciando maiúsculas e minúsculas. O padrão é usar a correspondência diferenciando maiúsculas e minúsculas.
- i – Executa a correspondência sem diferenciar maiúsculas de minúsculas.
- e – Extrai uma subsequência usando uma subexpressão.

  Se o padrão incluir uma subexpressão, REGEXP_INSTR corresponderá uma subsequência usando a primeira subexpressão em padrão. REGEXP_INSTR considera apenas a primeira subexpressão. As subexpressões adicionais são ignoradas. Se o padrão não tiver uma subexpressão, REGEXP_INSTR ignorará o parâmetro 'e'.
- p — Interpreta o padrão com o dialeto de expressão regular compatível com Perl (PCRE - Perl Compatible Regular Expression). Para ter mais informações sobre PCRE, consulte Perl Compatible Regular Expressions na Wikipédia.

Tipo de retorno

Inteiro

Exemplos

Os exemplos a seguir usam dados da tabela USERS do banco de dados de amostra TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para pesquisar o caractere @ que inicia o nome de um domínio e retorna a posição de início da primeira correspondência, use o exemplo a seguir.

```
SELECT email, REGEXP_INSTR(email, '@[^.]*')
FROM users
ORDER BY userid LIMIT 4;

+----------------------------------------------+--------------+
|                     email                    | regexp_instr |
+----------------------------------------------+--------------+
| Etiam.laoreet.libero@sodalesMaurisblandit.edu |           21 |
| Suspendisse.tristique@nonnisiAenean.edu       |           22 |
| amet.faucibus.ut@condimentumegetvolutpat.ca   |           17 |
| sed@lacusUtnec.ca                             |            4 |
+----------------------------------------------+--------------+
```

Para pesquisar as variantes da palavra `Center` e retornar a posição inicial da primeira correspondência, use o exemplo a seguir.

```
SELECT venuename, REGEXP_INSTR(venuename,'[cC]ent(er|re)$')
FROM venue
WHERE REGEXP_INSTR(venuename,'[cC]ent(er|re)$') > 0
ORDER BY venueid LIMIT 4;

+-----------------------+--------------+
|       venuename       | regexp_instr |
+-----------------------+--------------+
| The Home Depot Center |           16 |
| Izod Center           |            6 |
| Wachovia Center       |           10 |
| Air Canada Centre     |           12 |
+-----------------------+--------------+
```

Para encontrar a posição inicial da primeira ocorrência da string `FOX`, utilizando a lógica de correspondência que não diferencia letras maiúsculas de minúsculas, use o exemplo a seguir.

```
SELECT REGEXP_INSTR('the fox', 'FOX', 1, 1, 0, 'i');

+--------------+
| regexp_instr |
+--------------+
|            5 |
```

```
+--------------+
```

Para usar um padrão escrito no dialeto PCRE para localizar palavras contendo pelo menos um número e uma letra minúscula, use o exemplo a seguir. Ele usa o operador ?=, que tem uma conotação específica look-ahead em PCRE. Este exemplo encontra a posição inicial da segunda palavra.

```
SELECT REGEXP_INSTR('passwd7 plain A1234 a1234', '(?=[^ ]*[a-z])(?=[^ ]*[0-9])[^ ]+',
 1, 2, 0, 'p');

+--------------+
| regexp_instr |
+--------------+
|           21 |
+--------------+
```

Para usar um padrão escrito no dialeto PCRE para localizar palavras contendo pelo menos um número e uma letra minúscula, use o exemplo a seguir. Ele usa o operador ?=, que tem uma conotação específica look-ahead em PCRE. Este exemplo localiza a posição inicial da segunda palavra, mas difere do exemplo anterior na medida em que usa correspondência sem diferenciar maiúsculas de minúsculas.

```
SELECT REGEXP_INSTR('passwd7 plain A1234 a1234', '(?=[^ ]*[a-z])(?=[^ ]*[0-9])[^ ]+',
 1, 2, 0, 'ip');

+--------------+
| regexp_instr |
+--------------+
|           15 |
+--------------+
```

Função REGEXP_REPLACE

Pesquisa uma string quanto a um padrão de expressão regular e substitui cada ocorrência do padrão pela string especificada. REGEXP_REPLACE é semelhante a Função REPLACE, mas, com REGEXP_REPLACE, é possível pesquisar uma string em busca de um padrão de expressão regular. Para ter mais informações sobre expressões regulares, consulte Operadores POSIX e Expressão regular na Wikipédia.

REGEXP_REPLACE é semelhante a Função TRANSLATE e Função REPLACE, exceto que TRANSLATE faz várias substituições de caracteres individuais e REPLACE substitui uma string inteira por outra, enquanto, com REGEXP_REPLACE, é possível pesquisar uma string em busca de um padrão de expressão regular.

Sintaxe

```
REGEXP_REPLACE( source_string, pattern [, replace_string [ , position [, parameters
 ] ] ] )
```

Argumentos

source_string

Uma expressão de string CHAR ou VARCHAR, tal como um nome de coluna, a ser pesquisada.

Padrão

Um literal de string UTF-8 que representa um padrão de expressão regular. Para obter mais informações, consulte Operadores POSIX.

replace_string

(Opcional) Uma expressão de string CHAR ou VARCHAR, tal como um nome de coluna, que substituirá cada ocorrência do padrão. O padrão é uma string vazia ("").

position

(Opcional) Um inteiro positivo que indica a posição em source_string para começar a pesquisar. A posição é baseada no número de caracteres, e não bytes, de forma que caracteres multibyte são contados como caracteres simples. O padrão é 1. Se a posição for menor que 1, a pesquisa começará no primeiro caractere da source_string. Se position for maior que o número de caracteres na source_string, o resultado será source_string.

parameters

(Opcional) Uma ou mais literais de string que indicam como a função corresponde ao padrão. Os valores possíveis são os seguintes:

- c – Executa a correspondência diferenciando maiúsculas e minúsculas. O padrão é usar a correspondência diferenciando maiúsculas e minúsculas.
- i – Executa a correspondência sem diferenciar maiúsculas de minúsculas.

- p — Interpreta o padrão com o dialeto de expressão regular compatível com Perl (PCRE - Perl Compatible Regular Expression). Para ter mais informações sobre PCRE, consulte Perl Compatible Regular Expressions na Wikipédia.

Tipo de retorno

VARCHAR

Se o padrão ou replace_string for `NULL`, a função retornará `NULL`.

Exemplos

Para substituir todas as ocorrências da string `FOX` no valor `quick brown fox` usando a correspondência que não diferencia letras maiúsculas de minúsculas, use o exemplo a seguir.

```
SELECT REGEXP_REPLACE('the fox', 'FOX', 'quick brown fox', 1, 'i');

+----------------------+
|   regexp_replace     |
+----------------------+
| the quick brown fox |
+----------------------+
```

O exemplo a seguir usa um padrão escrito no dialeto PCRE para localizar palavras contendo pelo menos um número e uma letra minúscula. Ele usa o operador `?=`, que tem uma conotação específica look-ahead em PCRE. Para substituir cada ocorrência de tal palavra pelo valor `[hidden]`, use o exemplo a seguir.

```
SELECT REGEXP_REPLACE('passwd7 plain A1234 a1234', '(?=[^ ]*[a-z])(?=[^ ]*[0-9])[^ ]+',
 '[hidden]', 1, 'p');

+--------------------------------+
|        regexp_replace          |
+--------------------------------+
| [hidden] plain A1234 [hidden] |
+--------------------------------+
```

O exemplo a seguir usa um padrão escrito no dialeto PCRE para localizar palavras contendo pelo menos um número e uma letra minúscula. Ele usa o operador `?=`, que tem uma conotação específica look-ahead em PCRE. Para substituir cada ocorrência de tal palavra pelo valor `[hidden]`, mas que

difere do exemplo anterior porque que ele usa correspondência sem diferenciar letras maiúsculas de minúsculas, use o exemplo a seguir.

```
SELECT REGEXP_REPLACE('passwd7 plain A1234 a1234', '(?=[^ ]*[a-z])(?=[^ ]*[0-9])[^ ]+',
 '[hidden]', 1, 'ip');

+----------------------------------+
|          regexp_replace          |
+----------------------------------+
| [hidden] plain [hidden] [hidden] |
+----------------------------------+
```

Os exemplos a seguir usam dados da tabela USERS do banco de dados de amostra TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para excluir o `@` e o nome de domínio dos endereços de e-mail, use o exemplo a seguir.

```
SELECT email, REGEXP_REPLACE(email, '@.*\\.(org|gov|com|edu|ca)$')
FROM users
ORDER BY userid LIMIT 4;

+----------------------------------------------+----------------------+
|                     email                    |    regexp_replace    |
+----------------------------------------------+----------------------+
| Etiam.laoreet.libero@sodalesMaurisblandit.edu | Etiam.laoreet.libero  |
| Suspendisse.tristique@nonnisiAenean.edu       | Suspendisse.tristique |
| amet.faucibus.ut@condimentumegetvolutpat.ca   | amet.faucibus.ut      |
| sed@lacusUtnec.ca                             | sed                   |
+----------------------------------------------+----------------------+
```

Para substituir os nomes de domínio de endereços de e-mail por `internal.company.com`, use o exemplo a seguir.

```
SELECT email, REGEXP_REPLACE(email, '@.*\\.[[:alpha:]]{2,3}','@internal.company.com')
FROM users
ORDER BY userid LIMIT 4;

+----------------------------------------------
+-------------------------------------------+
|                     email                    |               regexp_replace
      |
```

```
+-----------------------------------------------
+-----------------------------------------+
| Etiam.laoreet.libero@sodalesMaurisblandit.edu |
 Etiam.laoreet.libero@internal.company.com  |
| Suspendisse.tristique@nonnisiAenean.edu       |
 Suspendisse.tristique@internal.company.com |
| amet.faucibus.ut@condimentumegetvolutpat.ca   | amet.faucibus.ut@internal.company.com
      |
| sed@lacusUtnec.ca                             | sed@internal.company.com
      |
+-----------------------------------------------
+-----------------------------------------+
```

Função REGEXP_SUBSTR

Retorna os caracteres de uma string ao procurar por um padrão de expressão regular. REGEXP_SUBSTR é semelhante à função Função SUBSTRING, mas, com REGEXP_SUBSTR, é possível pesquisar uma string em busca de um padrão de expressão regular. Se a função não conseguir corresponder a expressão regular com nenhum caractere na string, ela retornará uma string vazia. Para ter mais informações sobre expressões regulares, consulte Operadores POSIX e Expressão regular na Wikipédia.

Sintaxe

```
REGEXP_SUBSTR( source_string, pattern [, position [, occurrence [, parameters ] ] ] )
```

Argumentos

source_string

Uma expressão de string a ser pesquisada.

Padrão

Um literal de string UTF-8 que representa um padrão de expressão regular. Para obter mais informações, consulte Operadores POSIX.

position

Um inteiro positivo que indica a posição em source_string para começar a pesquisar. A posição é baseada no número de caracteres, e não bytes, de forma que caracteres multibyte são contados como caracteres simples. O padrão é 1. Se a posição for menor que 1, a pesquisa começará

no primeiro caractere da source_string. Se position for maior que o número de caracteres na source_string, o resultado será uma string vazia ("").

occurrence

Um inteiro positivo que indica qual ocorrência do padrão usar. REGEXP_SUBSTR ignora as primeiras correspondências de occurrence -1. O padrão é 1. Se a ocorrência for menor que 1 ou maior que o número de caracteres em source_string, a pesquisa será ignorada e o resultado será vazio.

parameters

Uma ou mais literais de sequências que indicam como a função corresponde o padrão. Os valores possíveis são os seguintes:

- c – Executa a correspondência diferenciando maiúsculas e minúsculas. O padrão é usar a correspondência diferenciando maiúsculas e minúsculas.
- i – Executa a correspondência sem diferenciar maiúsculas de minúsculas.
- e – Extrai uma subsequência usando uma subexpressão.

  Se o padrão incluir uma subexpressão, REGEXP_SUBSTR corresponderá uma subsequência usando a primeira subexpressão em padrão. Uma subexpressão é uma expressão dentro do padrão que está entre parênteses. Por exemplo, para o padrão `'This is a (\\w+)'` faz correspondência com a primeira expressão com a string `'This is a '` seguida por uma palavra. Em vez de retornar o padrão, REGEXP_SUBSTR com o parâmetro e retorna somente a string dentro da subexpressão.

  REGEXP_SUBSTR considera apenas a primeira subexpressão. As subexpressões adicionais são ignoradas. Se o padrão não tiver uma subexpressão, REGEXP_SUBSTR ignorará o parâmetro 'e'.
- p — Interpreta o padrão com o dialeto de expressão regular compatível com Perl (PCRE - Perl Compatible Regular Expression). Para ter mais informações sobre PCRE, consulte Perl Compatible Regular Expressions na Wikipédia.

Tipo de retorno

VARCHAR

Exemplos

O seguinte exemplo retorna a porção de um endereço de e-mail entre o caractere @ e a extensão de domínio. Os dados de `users` consultados são dos dados de amostra do Amazon Redshift. Para obter mais informações, consulte Banco de dados de exemplo.

```
SELECT email, regexp_substr(email,'@[^.]*')
FROM users
ORDER BY userid LIMIT 4;

                     email                     |      regexp_substr
-----------------------------------------------+--------------------------
 Suspendisse.tristique@nonnisiAenean.edu       | @nonnisiAenean
 amet.faucibus.ut@condimentumegetvolutpat.ca   | @condimentumegetvolutpat
 sed@lacusUtnec.ca                             | @lacusUtnec
 Cum@accumsan.com                              | @accumsan
```

O exemplo a seguir retorna a porção da entrada correspondente à primeira ocorrência da string FOX com a correspondência sem diferenciar letras maiúsculas de minúsculas.

```
SELECT regexp_substr('the fox', 'FOX', 1, 1, 'i');

 regexp_substr
---------------
 fox
```

O exemplo a seguir retorna a porção da entrada correspondente à segunda ocorrência da string FOX usando a correspondência que não diferencia letras maiúsculas de minúsculas. O resultado é vazio (não nulo, tamanho 0) porque não há uma segunda ocorrência.

```
SELECT regexp_substr('the fox', 'FOX', 1, 2, 'i');

 regexp_substr
---------------

```

O exemplo a seguir retorna a primeira parte da entrada que começa com letras minúsculas. Isso é funcionalmente idêntico à mesma instrução SELECT sem o parâmetro `c`.

```
SELECT regexp_substr('THE SECRET CODE IS THE LOWERCASE PART OF 1931abc0EZ.', '[a-z]+',
 1, 1, 'c');
```

```
 regexp_substr
---------------
 abc
```

O exemplo a seguir usa um padrão escrito no dialeto PCRE para localizar palavras contendo pelo menos um número e uma letra minúscula. Ele usa o operador ?=, que tem uma conotação específica look-ahead em PCRE. Este exemplo retorna a parte da entrada correspondente à segunda palavra.

```
SELECT regexp_substr('passwd7 plain A1234 a1234', '(?=[^ ]*[a-z])(?=[^ ]*[0-9])[^ ]+',
 1, 2, 'p');

 regexp_substr
---------------
 a1234
```

O exemplo a seguir usa um padrão escrito no dialeto PCRE para localizar palavras contendo pelo menos um número e uma letra minúscula. Ele usa o operador ?=, que tem uma conotação específica look-ahead em PCRE. Este exemplo retorna a parte da entrada correspondente à segunda palavra, mas difere do exemplo anterior na medida em que usa a correspondência sem diferenciar maiúsculas de minúsculas.

```
SELECT regexp_substr('passwd7 plain A1234 a1234', '(?=[^ ]*[a-z])(?=[^ ]*[0-9])[^ ]+',
 1, 2, 'ip');

 regexp_substr
---------------
 A1234
```

O exemplo a seguir usa uma subexpressão para encontrar a segunda string correspondente ao padrão `'this is a (\\w+)'` usando a correspondência que não diferencia letras maiúsculas de minúsculas. Ele retorna a subexpressão dentro dos parênteses.

```
SELECT regexp_substr(
               'This is a cat, this is a dog. This is a mouse.',
               'this is a (\\w+)', 1, 2, 'ie');

 regexp_substr
---------------
 dog
```

Função REPEAT

Repete uma string pelo número especificado de vezes. Se o parâmetro de entrada for numérico, REPEAT o tratará como uma string.

Sinônimo de Função REPLICATE.

Sintaxe

```
REPEAT(string, integer)
```

Argumentos

string

O primeiro parâmetro de entrada é a string a ser repetida.

integer

O segundo parâmetro é um INTEGER indicando o número de vezes a repetir a string.

Tipo de retorno

VARCHAR

Exemplos

O exemplo a seguir usa dados da tabela CATEGORY do banco de dados de amostra TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para repetir o valor da coluna CATID na tabela CATEGORY três vezes, use o exemplo a seguir.

```
SELECT catid, REPEAT(catid,3)
FROM category
ORDER BY 1,2;

+-------+--------+
| catid | repeat |
+-------+--------+
|     1 |    111 |
|     2 |    222 |
|     3 |    333 |
|     4 |    444 |
```

```
|     5 |    555 |
|     6 |    666 |
|     7 |    777 |
|     8 |    888 |
|     9 |    999 |
|    10 | 101010 |
|    11 | 111111 |
+-------+--------+
```

O exemplo a seguir demonstra a geração de strings de até 16 milhões de bytes:

```
SELECT
    LEN(REPEAT('X', 5000000)) AS five_million_bytes,
    LEN(REPEAT('Y', 16000000)) AS sixteen_million_bytes;

 five_million_bytes  | sixteen_million_bytes
---------+-----------
 5000000  | 16000000
```

Função REPLACE

Substitui todas as ocorrências de um conjunto de caracteres em uma string existente por outros caracteres especificados.

REPLACE é semelhante a Função TRANSLATE e Função REGEXP_REPLACE, exceto que TRANSLATE faz várias substituições de caracteres individuais e, com REGEXP_REPLACE, é possível pesquisar uma string em busca de um padrão de expressão regular, enquanto REPLACE substitui uma string inteira por outra.

Sintaxe

```
REPLACE(string, old_chars, new_chars)
```

Argumentos

string

String CHAR ou VARCHAR a ser pesquisada

old_chars

A string CHAR ou VARCHAR a ser substituída.

new_chars

Nova string CHAR ou VARCHAR que substitui old_string.

Tipo de retorno

VARCHAR

Se old_chars ou new_chars for NULL, o retorno será NULL.

Exemplos

O exemplo a seguir usa dados da tabela CATEGORY do banco de dados de amostra TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para converter a string Shows em Theatre no campo CATGROUP, use o exemplo a seguir.

```
SELECT catid, catgroup, REPLACE(catgroup, 'Shows', 'Theatre')
FROM category
ORDER BY 1,2,3;

+-------+----------+----------+
| catid | catgroup | replace  |
+-------+----------+----------+
|     1 | Sports   | Sports   |
|     2 | Sports   | Sports   |
|     3 | Sports   | Sports   |
|     4 | Sports   | Sports   |
|     5 | Sports   | Sports   |
|     6 | Shows    | Theatre  |
|     7 | Shows    | Theatre  |
|     8 | Shows    | Theatre  |
|     9 | Concerts | Concerts |
|    10 | Concerts | Concerts |
|    11 | Concerts | Concerts |
+-------+----------+----------+
```

Função REPLICATE

Sinônimo da função REPEAT.

Consulte Função REPEAT.

Função REVERSE

A função REVERSE opera em uma string e retorna os caracteres na ordem reversa. Por exemplo, `reverse('abcde')` exibe `edcba`. Essa função funciona em tipos de dados numéricos e de data, assim como nos tipos de dados de caracteres; no entanto, na maioria dos casos ela possui um valor prático para strings de caracteres.

Sintaxe

```
REVERSE( expression )
```

Argumento

expressão

Uma expressão com um tipo de dados de caractere, data, timestamp ou numérico que representa o destino da reversão de caracteres. Todas as expressões são convertidas implicitamente em strings `VARCHAR`. Espaços em branco finais em strings `CHAR` são ignorados.

Tipo de retorno

VARCHAR

Exemplos

Os exemplos a seguir usam dados da tabela USERS e SALES do banco de dados de amostra TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para selecionar cinco nomes de cidade distintos e seus nomes invertidos correspondentes da tabela USERS, use o exemplo a seguir.

```
SELECT DISTINCT city AS cityname, REVERSE(cityname)
FROM users
ORDER BY city LIMIT 5;

+----------+----------+
| cityname | reverse  |
+----------+----------+
| Aberdeen | needrebA |
| Abilene  | enelibA  |
```

```
| Ada      | adA      |
| Agat     | tagA     |
| Agawam   | mawagA   |
+----------+----------+
```

Para selecionar cinco IDs de vendas e seus IDs invertidos correspondentes convertidos como strings de caracteres, use o exemplo a seguir.

```
SELECT salesid, REVERSE(salesid)
FROM sales
ORDER BY salesid DESC LIMIT 5;

+---------+---------+
| salesid | reverse |
+---------+---------+
|  172456 |  654271 |
|  172455 |  554271 |
|  172454 |  454271 |
|  172453 |  354271 |
|  172452 |  254271 |
+---------+---------+
```

Função RTRIM

A função RTRIM apara um conjunto específico de caracteres do final de uma string. Remove a string mais longa que contém somente caracteres que estão na lista de caracteres de corte. O corte é concluído quando nenhum caractere de corte aparece na string de entrada.

Sintaxe

```
RTRIM( string, trim_chars )
```

Argumentos

string

Uma coluna, expressão ou literal de string a ser cortado.

trim_chars

Uma coluna, expressão ou literal de string que representa os caracteres a serem cortados do final da string. Se não for especificado, um espaço será usado como caractere de corte.

Tipo de retorno

Uma string no mesmo tipo de dados que o argumento da string.

Exemplo

O seguinte exemplo apara espaços em branco inicias e finais da string `' abc '`:

```
select '     abc    ' as untrim, rtrim('     abc    ') as trim;

untrim    | trim
----------+------
   abc    |    abc
```

O exemplo a seguir remove a string `'xyz'` final da string `'xyzaxyzbxyzcxyz'`. As ocorrências iniciais de `'xyz'` são removidas, mas as ocorrências internas da string não são removidas.

```
select 'xyzaxyzbxyzcxyz' as untrim,
rtrim('xyzaxyzbxyzcxyz', 'xyz') as trim;

     untrim      |   trim
-----------------+-----------
 xyzaxyzbxyzcxyz | xyzaxyzbxyzc
```

O exemplo a seguir remove as partes finais da string `'setuphistorycassettes'` que correspondem a qualquer um dos caracteres na lista trim_chars `'tes'`. Qualquer `t`, `e` ou `s` que ocorra antes que outro caractere que não esteja na lista trim_chars no final da string de entrada é removido.

```
SELECT rtrim('setuphistorycassettes', 'tes');

     rtrim
----------------
 setuphistoryca
```

O seguinte exemplo apara os caracteres "Park" do final de VENUENAME, onde presente:

```
select venueid, venuename, rtrim(venuename, 'Park')
from venue
order by 1, 2, 3
limit 10;
```

```
venueid |         venuename          |          rtrim
--------+----------------------------+-------------------------
      1 | Toyota Park                | Toyota
      2 | Columbus Crew Stadium      | Columbus Crew Stadium
      3 | RFK Stadium                | RFK Stadium
      4 | CommunityAmerica Ballpark  | CommunityAmerica Ballp
      5 | Gillette Stadium           | Gillette Stadium
      6 | New York Giants Stadium    | New York Giants Stadium
      7 | BMO Field                  | BMO Field
      8 | The Home Depot Center      | The Home Depot Cente
      9 | Dick's Sporting Goods Park | Dick's Sporting Goods
     10 | Pizza Hut Park             | Pizza Hut
```

Observe que RTRIM remove qualquer um dos caracteres `P`, `a`, `r` ou `k` que aparecem no final de um VENUENAME.

Função SOUNDEX

A função SOUNDEX retorna o valor American Soundex consistindo na primeira letra da string de entrada seguida de uma codificação de três dígitos dos sons que representam a pronúncia em inglês da string especificada. Por exemplo, `Smith` e `Smyth` têm o mesmo valor do Soundex.

Sintaxe

```
SOUNDEX(string)
```

Argumentos

string

Especificar uma string de caracteres `CHAR` ou `VARCHAR` que pretende converter em um valor de código American Soundex.

Tipo de retorno

VARCHAR(4)

Observações de uso

A função DIFFERENCE converte apenas caracteres ASCII em letras minúsculas ou maiúsculas em inglês, incluindo a–z e A–Z. SOUNDEX ignora outros caracteres. SOUNDEX retorna um único valor Soundex para uma string de várias palavras separadas por espaços.

```
SELECT SOUNDEX('AWS Amazon');

+---------+
| soundex |
+---------+
| A252    |
+---------+
```

SOUNDEX retorna uma string vazia se a string de entrada não contém letras inglesas.

```
SELECT SOUNDEX('+-*/%');

+---------+
| soundex |
+---------+
|         |
+---------+
```

Exemplos

Para retornar o valor do Soundex para `Amazon`, use o exemplo a seguir.

```
SELECT SOUNDEX('Amazon');

+---------+
| soundex |
+---------+
| A525    |
+---------+
```

Para retornar o valor do Soundex para `smith` e `smyth`, use o exemplo a seguir. Observe que os valores do Soundex são os mesmos.

```
SELECT SOUNDEX('smith'), SOUNDEX('smyth');

+-------+-------+
| smith | smyth |
+-------+-------+
| S530  | S530  |
+-------+-------+
```

Função SPLIT_PART

Divide uma string no delimitador especificado e retorna a parte na posição especificada.

Sintaxe

```
SPLIT_PART(string, delimiter, position)
```

Argumentos

string

Uma coluna, expressão ou literal de string a ser dividido. A string pode ser CHAR ou VARCHAR.

delimitador

A string delimitadora que indica seções da string de entrada.

Se o delimitador for um literal, coloque-o entre aspas simples.

position

Posição da porção da string a retornar (contando de 1). Deve ser um número inteiro maior que 0. Se position for maior que o número de porções de string, SPLIT_PART retornará uma string vazia. Se delimiter não for encontrado em string, o valor retornado conterá o conteúdo da parte especificada, que poderá ser toda a string ou um valor vazio.

Tipo de retorno

Uma string CHAR ou VARCHAR, o mesmo que o parâmetro da string.

Exemplos

O exemplo a seguir divide uma string literal em partes usando o delimitador $ e retorna a segunda parte.

```
select split_part('abc$def$ghi','$',2)

split_part
----------
def
```

O exemplo a seguir divide uma string literal em partes usando o delimitador $. Ele retorna uma string vazia porque a parte 4 não foi encontrada.

```
select split_part('abc$def$ghi','$',4)

split_part
----------

```

O exemplo a seguir divide uma string literal em partes usando o delimitador #. Ele retorna a string inteira, que é a primeira parte, porque o delimitador não foi encontrado.

```
select split_part('abc$def$ghi','#',1)

split_part
------------
abc$def$ghi
```

O exemplo a seguir divide o campo de timestamp LISTTIME em componentes de ano, mês e dia.

```
select listtime, split_part(listtime,'-',1) as year,
split_part(listtime,'-',2) as month,
split_part(split_part(listtime,'-',3),' ',1) as day
from listing limit 5;

      listtime       | year | month | day
---------------------+------+-------+------
 2008-03-05 12:25:29 | 2008 | 03    | 05
 2008-09-09 08:03:36 | 2008 | 09    | 09
 2008-09-26 05:43:12 | 2008 | 09    | 26
 2008-10-04 02:00:30 | 2008 | 10    | 04
 2008-01-06 08:33:11 | 2008 | 01    | 06
```

O seguinte exemplo seleciona o campo de timestamp LISTTIME e o divide no caractere `'-'` para obter o mês (a segunda parte da string LISTTIME) e, então, conta o número de entradas para cada mês:

```
select split_part(listtime,'-',2) as month, count(*)
from listing
group by split_part(listtime,'-',2)
order by 1, 2;
```

```
 month | count
-------+-------
    01 | 18543
    02 | 16620
    03 | 17594
    04 | 16822
    05 | 17618
    06 | 17158
    07 | 17626
    08 | 17881
    09 | 17378
    10 | 17756
    11 | 12912
    12 | 4589
```

Função STRPOS

Retorna a posição de uma substring em uma string especificada.

Consulte Função CHARINDEX e Função POSITION para ver funções semelhantes.

Sintaxe

```
STRPOS(string, substring )
```

Argumentos

string

O primeiro parâmetro de entrada é a string CHAR ou VARCHAR a ser pesquisada.

substring

O segundo parâmetro é a substring a procurar dentro da string.

Tipo de retorno

INTEGER

A função STRPOS retorna um INTEGER correspondente à posição da substring (baseada em 1, não baseada em zero). A posição é baseada no número de caracteres, e não bytes, de forma que caracteres multibyte são contados como caracteres simples.

Observações de uso

STRPOS retornará `0` se a substring não for localizada dentro da string.

```
SELECT STRPOS('dogfish', 'fist');

+--------+
| strpos |
+--------+
|      0 |
+--------+
```

Exemplos

Para mostrar a posição de `fish` em `dogfish`, use o exemplo a seguir.

```
SELECT STRPOS('dogfish', 'fish');

+--------+
| strpos |
+--------+
|      4 |
+--------+
```

O exemplo a seguir usa dados da tabela SALES do banco de dados de amostra TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para retornar o número de transações de vendas com uma COMMISSION acima de 999,00 da tabela SALES, use o exemplo a seguir.

```
SELECT DISTINCT STRPOS(commission, '.'),
COUNT (STRPOS(commission, '.'))
FROM sales
WHERE STRPOS(commission, '.') > 4
GROUP BY STRPOS(commission, '.')
ORDER BY 1, 2;

+--------+-------+
| strpos | count |
+--------+-------+
|      5 |   629 |
+--------+-------+
```

Função STRTOL

Converte a expressão de string de um número da base especificada ao valor inteiro equivalente. O valor convertido deve estar dentro do intervalo de 64 bits assinado.

Sintaxe

```
STRTOL(num_string, base)
```

Argumentos

num_string

Expressão de string de um número a ser convertido. Se num_string estiver vazia ('') ou começar com o caractere nulo ('\0'), o valor convertido será 0. Se num_string for uma coluna que contém um valor NULL, STRTOL retornará NULL. A string pode começar com qualquer quantidade de espaços em branco seguida, opcionalmente seguida por um sinal de mais "+" ou menos "-" para indicar positivo ou negativo. O padrão é '+'. Se base for 16, a string pode opcionalmente começar com "0x".

base

INTEGER entre 2 e 36.

Tipo de retorno

BIGINT

Se num_string for nulo, a função retornará NULL.

Exemplos

Para converter pares de valores de string e base para números inteiros, use os exemplos a seguir.

```
SELECT STRTOL('0xf',16);

+--------+
| strtol |
+--------+
|     15 |
+--------+
```

```
SELECT STRTOL('abcd1234',16);

+------------+
|   strtol   |
+------------+
| 2882343476 |
+------------+

SELECT STRTOL('1234567', 10);

+---------+
| strtol  |
+---------+
| 1234567 |
+---------+

SELECT STRTOL('1234567', 8);

+--------+
| strtol |
+--------+
| 342391 |
+--------+

SELECT STRTOL('110101', 2);

+--------+
| strtol |
+--------+
|     53 |
+--------+

SELECT STRTOL('\0', 2);

+--------+
| strtol |
+--------+
|      0 |
+--------+
```

Função SUBSTRING

Retorna o subconjunto de uma string com base na posição inicial especificada da string.

Se a entrada for uma cadeia de caracteres, a posição inicial e o número de caracteres extraídos são baseados nos caracteres, e não bytes, de forma que caracteres multibyte são contados como caracteres simples. Se a entrada for uma expressão binária, a posição inicial e a substring extraída são baseadas em bytes. Você não pode especificar um comprimento negativo, mas pode especificar uma posição inicial negativa.

Sintaxe

```
SUBSTRING(character_string FROM start_position [ FOR number_characters ] )
```

```
SUBSTRING(character_string, start_position, number_characters )
```

```
SUBSTRING(binary_expression, start_byte, number_bytes )
```

```
SUBSTRING(binary_expression, start_byte )
```

Argumentos

character_string

A string a ser pesquisada. Tipos de dados não caracteres são tratados como uma string.

start_position

A posição dentro da sequência para começar a extração, começando em 1. A start_position é baseada no número de caracteres, e não bytes, de forma que caracteres multibyte são contados como caracteres simples. Esse número pode ser negativo.

number_characters

O número de caracteres a extrair (o comprimento da substring). O number_characters é baseado no número de caracteres, e não bytes, de forma que caracteres multibyte são contados como caracteres simples. Esse número não pode ser negativo.

binary_expression

O binary_expression do tipo de dados VARBYTE a ser pesquisado.

start_byte

A posição dentro da expressão binária para começar a extração, começando por 1. Esse número pode ser negativo.

number_bytes

O número de bytes a serem extraídos, ou seja, o comprimento da substring. Esse número não pode ser negativo.

Tipo de retorno

VARCHAR ou VARBYTE, de acordo com a entrada.

Observações sobre o uso

Veja a seguir alguns exemplos de como você pode usar start_position e number_characters para extrair substrings de várias posições em uma string.

O seguinte exemplo retorna uma string de quatro caracteres começando com o sexto caractere.

```
select substring('caterpillar',6,4);
substring
-----------
pill
(1 row)
```

Se start_position + number_characters exceder o comprimento da string, SUBSTRING exibirá uma substring que começa na start_position e vai até o final da string. Por exemplo:

```
select substring('caterpillar',6,8);
substring
-----------
pillar
(1 row)
```

Se `start_position` for negativa ou 0, a função SUBSTRING retornará uma substring começando no primeiro caractere da string com um comprimento de `start_position` + `number_characters` -1. Por exemplo:

```
select substring('caterpillar',-2,6);
substring
-----------
cat
(1 row)
```

Se `start_position` + `number_characters` -1 for menor ou igual a zero, a SUBSTRING retornará uma string vazia. Por exemplo:

```
select substring('caterpillar',-5,4);
substring
-----------

(1 row)
```

Exemplos

O seguinte exemplo retorna o mês da string LISTTIME na tabela LISTING:

```
select listid, listtime,
substring(listtime, 6, 2) as month
from listing
order by 1, 2, 3
limit 10;

 listid |      listtime       | month
--------+---------------------+-------
      1 | 2008-01-24 06:43:29 | 01
      2 | 2008-03-05 12:25:29 | 03
      3 | 2008-11-01 07:35:33 | 11
      4 | 2008-05-24 01:18:37 | 05
      5 | 2008-05-17 02:29:11 | 05
      6 | 2008-08-15 02:08:13 | 08
      7 | 2008-11-15 09:38:15 | 11
      8 | 2008-11-09 05:07:30 | 11
      9 | 2008-09-09 08:03:36 | 09
     10 | 2008-06-17 09:44:54 | 06
(10 rows)
```

O seguinte exemplo é o mesmo que o exemplo acima, mas usa a opção FROM...FOR:

```
select listid, listtime,
substring(listtime from 6 for 2) as month
from listing
order by 1, 2, 3
limit 10;

 listid |      listtime       | month
```

```
--------+---------------------+-------
      1 | 2008-01-24 06:43:29 | 01
      2 | 2008-03-05 12:25:29 | 03
      3 | 2008-11-01 07:35:33 | 11
      4 | 2008-05-24 01:18:37 | 05
      5 | 2008-05-17 02:29:11 | 05
      6 | 2008-08-15 02:08:13 | 08
      7 | 2008-11-15 09:38:15 | 11
      8 | 2008-11-09 05:07:30 | 11
      9 | 2008-09-09 08:03:36 | 09
     10 | 2008-06-17 09:44:54 | 06
(10 rows)
```

Não é possível usar a SUBSTRING para extrair previsivelmente o prefixo de uma string que possa conter caracteres multibyte, pois é necessário especificar o comprimento de uma string multibyte com base no número de bytes, e não no número de caracteres. Para extrair o segmento inicial de uma sequência com base no comprimento em bytes, você pode CAST a string como VARCHAR (byte_length) para truncar a string, onde byte_length é o tamanho exigido. O seguinte exemplo extrai os primeiros cinco bytes da string `'Fourscore and seven'`.

```
select cast('Fourscore and seven' as varchar(5));

varchar
-------
Fours
```

O seguinte exemplo mostra uma posição inicial negativa de um valor binário `abc`. Como a posição inicial é -3, a substring é extraída do início do valor binário. O resultado é exibido automaticamente como a representação hexadecimal da substring binária.

```
select substring('abc'::varbyte, -3);

 substring
-----------
 616263
```

O seguinte exemplo mostra 1 para a posição inicial de um valor binário `abc`. Como não há tamanho especificado, a string é extraída da posição inicial até o final da string. O resultado é exibido automaticamente como a representação hexadecimal da substring binária.

```
select substring('abc'::varbyte, 1);
```

```
 substring
-----------
 616263
```

O exemplo a seguir mostra 3 para a posição inicial de um valor binário `abc`. Como não há tamanho especificado, a string é extraída da posição inicial até o final da string. O resultado é exibido automaticamente como a representação hexadecimal da substring binária.

```
select substring('abc'::varbyte, 3);

 substring
-----------
 63
```

O exemplo a seguir mostra 2 para a posição inicial de um valor binário `abc`. A string é extraída da posição inicial para a posição 10, mas o final da string está na posição 3. O resultado é exibido automaticamente como a representação hexadecimal da substring binária.

```
select substring('abc'::varbyte, 2, 10);

 substring
-----------
 6263
```

O exemplo a seguir mostra 2 para a posição inicial de um valor binário `abc`. A string é extraída da posição inicial por 1 byte. O resultado é exibido automaticamente como a representação hexadecimal da substring binária.

```
select substring('abc'::varbyte, 2, 1);

 substring
-----------
 62
```

O exemplo a seguir retorna o nome `Ana` que aparece após o último espaço na string de entrada `Silva, Ana`.

```
select reverse(substring(reverse('Silva, Ana'), 1, position(' ' IN reverse('Silva,
 Ana'))))
```

```
 reverse
-----------
 Ana
```

Função TEXTLEN

Sinônimo da função LEN.

Consulte Função LEN.

Função TRANSLATE

Para dada expressão, substitui todas as ocorrências dos caracteres especificados pelos substitutos especificados. Os caracteres existentes são mapeados aos caracteres de substituição pelas suas posições nos argumentos characters_to_replace e characters_to_substitute. Se mais caracteres estiverem especificados no argumento characters_to_replace que no argumento characters_to_substitute, os caracteres adicionais do argumento characters_to_replace serão omitidos do valor de retorno.

TRANSLATE é semelhante a Função REPLACE e Função REGEXP_REPLACE, exceto que REPLACE substitui uma string inteira por outra e, com REGEXP_REPLACE, é possível pesquisar uma string em busca de um padrão de expressão regular, enquanto TRANSLATE faz várias substituições de caracteres individuais.

Se qualquer um dos argumentos for nulo, o retorno será NULL.

Sintaxe

```
TRANSLATE( expression, characters_to_replace, characters_to_substitute )
```

Argumentos

expressão

A expressão a ser traduzida.

characters_to_replace

Uma string contendo os caracteres a serem substituídos.

characters_to_substitute

Uma string contendo os caracteres a substituir.

Tipo de retorno

VARCHAR

Exemplos

Para substituir vários caracteres em uma string, use o exemplo a seguir.

```
SELECT TRANSLATE('mint tea', 'inea', 'osin');

+-----------+
| translate |
+-----------+
| most tin  |
+-----------+
```

Os exemplos a seguir usam dados da tabela USERS do banco de dados de amostra TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para substituir o sinal (@) por um ponto final para todos os valores de uma coluna, use o exemplo a seguir.

```
SELECT email, TRANSLATE(email, '@', '.') as obfuscated_email
FROM users LIMIT 10;

+--------------------------------------+--------------------------------------+
|                email                 |           obfuscated_email           |
+--------------------------------------+--------------------------------------+
| Cum@accumsan.com                     | Cum.accumsan.com                     |
| lorem.ipsum@Vestibulumante.com       | lorem.ipsum.Vestibulumante.com       |
| non.justo.Proin@ametconsectetuer.edu | non.justo.Proin.ametconsectetuer.edu |
| non.ante.bibendum@porttitortellus.org | non.ante.bibendum.porttitortellus.org |
| eros@blanditatnisi.org               | eros.blanditatnisi.org               |
| augue@Donec.ca                       | augue.Donec.ca                       |
| cursus@pedeacurna.edu                | cursus.pedeacurna.edu                |
| at@Duis.com                          | at.Duis.com                          |
| quam@facilisisvitaeorci.ca           | quam.facilisisvitaeorci.ca           |
| mi.lorem@nunc.edu                    | mi.lorem.nunc.edu                    |
```

```
+--------------------------------------+--------------------------------------+
```

Para substituir espaços por sublinhados e remover pontos finais de todos os valores em uma coluna, use o exemplo a seguir.

```
SELECT city, TRANSLATE(city, ' .', '_')
FROM users
WHERE city LIKE 'Sain%' OR city LIKE 'St%'
GROUP BY city
ORDER BY city;

+----------------+---------------+
|      city      |   translate   |
+----------------+---------------+
| Saint Albans   | Saint_Albans  |
| Saint Cloud    | Saint_Cloud   |
| Saint Joseph   | Saint_Joseph  |
| Saint Louis    | Saint_Louis   |
| Saint Paul     | Saint_Paul    |
| St. George     | St_George     |
| St. Marys      | St_Marys      |
| St. Petersburg | St_Petersburg |
| Stafford       | Stafford      |
| Stamford       | Stamford      |
| Stanton        | Stanton       |
| Starkville     | Starkville    |
| Statesboro     | Statesboro    |
| Staunton       | Staunton      |
| Steubenville   | Steubenville  |
| Stevens Point  | Stevens_Point |
| Stillwater     | Stillwater    |
| Stockton       | Stockton      |
| Sturgis        | Sturgis       |
+----------------+---------------+
```

Função TRIM

Corta da string os espaços em branco ou caracteres especificados.

Sintaxe

```
TRIM( [ BOTH | LEADING | TRAILING ] [trim_chars FROM ] string )
```

Argumentos

BOTH | LEADING | TRAILING

(Opcional) Especifica de onde cortar os caracteres. Use BOTH para remover caracteres iniciais e finais, use LEADING para remover somente caracteres iniciais e use TRAILING para remover somente caracteres finais. Se esse parâmetro for omitido, os caracteres iniciais e finais serão cortados.

trim_chars

(Opcional) Os caracteres a serem aparados da string. Se este parâmetro for omitido, espaços em branco serão aparados.

string

A string a ser aparada.

Tipo de retorno

A função TRIM retorna uma string VARCHAR ou CHAR. Se você usar a função TRIM com um comando SQL, o Amazon Redshift converterá implicitamente os resultados em VARCHAR. Se você usar a função TRIM na lista SELECT para uma função SQL, o Amazon Redshift não converterá implicitamente os resultados e poderá ser necessário realizar uma conversão explícita para evitar um erro de incompatibilidade de tipo de dados. Consulte as funções Função CAST e Função CONVERT para obter informações sobre conversões explícitas.

Exemplos

Para remover espaços em branco inicias e finais da string `  dog  `, use o exemplo a seguir.

```
SELECT TRIM('    dog    ');

+-------+
| btrim |
+-------+
| dog   |
+-------+
```

Para remover espaços em branco inicias e finais da string `  dog  `, use o exemplo a seguir.

```
SELECT TRIM(BOTH FROM '    dog    ');
```

```
+-------+
| btrim |
+-------+
| dog   |
+-------+
```

Para remover as aspas duplas iniciais da string "dog", use o exemplo a seguir.

```
SELECT TRIM(LEADING '"' FROM'"dog"');

+-------+
| ltrim |
+-------+
| dog"  |
+-------+
```

Para remover as aspas duplas finais da string "dog", use o exemplo a seguir.

```
SELECT TRIM(TRAILING '"' FROM'"dog"');

+-------+
| rtrim |
+-------+
| "dog  |
+-------+
```

TRIM remove qualquer um dos caracteres em trim_chars quando eles aparecem no início ou no final da string. O exemplo a seguir apara os caracteres “C”, “D” e “G” quando eles aparecem no início de VENUENAME, que é uma coluna VARCHAR. Para obter mais informações, consulte Banco de dados de exemplo.

```
SELECT venueid, venuename, TRIM('CDG' FROM venuename)
FROM venue
WHERE venuename LIKE '%Park'
ORDER BY 2
LIMIT 7;

+---------+----------------------------+----------------------------+
| venueid |         venuename          |           btrim            |
+---------+----------------------------+----------------------------+
|     121 | AT&T Park                  | AT&T Park                  |
|     109 | Citizens Bank Park         | itizens Bank Park          |
```

```
|     102 | Comerica Park              | omerica Park              |
|       9 | Dick's Sporting Goods Park | ick's Sporting Goods Park |
|      97 | Fenway Park                | Fenway Park               |
|     112 | Great American Ball Park   | reat American Ball Park   |
|     114 | Miller Park                | Miller Park               |
+---------+----------------------------+---------------------------+
```

Função UPPER

Converte uma string em letras maiúsculas. UPPER é compatível com caracteres UTF-8 multibyte, até o máximo de quatro bytes por caractere.

Sintaxe

```
UPPER(string)
```

Argumentos

string

O parâmetro de entrada é uma string VARCHAR ou qualquer outro tipo de dados, como CHAR, que pode ser convertido implicitamente em VARCHAR.

Tipo de retorno

A função UPPER retorna uma string de caracteres que é o mesmo tipo de dados da string de entrada. Por exemplo, a função retornará uma string VARCHAR se a entrada for uma string VARCHAR.

Exemplos

O exemplo a seguir usa dados da tabela CATEGORY do banco de dados de amostra TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

Para converter o campo CATNAME em maiúsculas, use o exemplo a seguir.

```
SELECT catname, UPPER(catname)
FROM category
ORDER BY 1,2;

+-----------+-----------+
|  catname  |   upper   |
+-----------+-----------+
```

```
| Classical | CLASSICAL |
| Jazz      | JAZZ      |
| MLB       | MLB       |
| MLS       | MLS       |
| Musicals  | MUSICALS  |
| NBA       | NBA       |
| NFL       | NFL       |
| NHL       | NHL       |
| Opera     | OPERA     |
| Plays     | PLAYS     |
| Pop       | POP       |
+-----------+-----------+
```

Funções de informação de tipo SUPER

A seguir, você pode encontrar uma descrição para as funções de informações de tipo para SQL aceitas pelo Amazon Redshift para derivar as informações dinâmicas de entradas do tipo de dados SUPER.

Tópicos

- Função DECIMAL_PRECISION
- Função DECIMAL_SCALE
- Função IS_ARRAY
- Função IS_BIGINT
- Função IS_BOOLEAN
- Função IS_CHAR
- Função IS_DECIMAL
- Função IS_FLOAT
- Função IS_INTEGER
- Função IS_OBJECT
- Função IS_SCALAR
- Função IS_SMALLINT
- Função IS_VARCHAR
- Função JSON_SIZE
- Função JSON_TYPEOF
- SIZE

Função DECIMAL_PRECISION

Verifica a precisão do número total máximo de dígitos decimais a serem armazenados. Este número inclui os dígitos esquerdo e direito do ponto decimal. O alcance da precisão é de 1 a 38, com um padrão de 38.

Sintaxe

```
DECIMAL_PRECISION(super_expression)
```

Argumentos

super_expression

Uma expressão ou coluna SUPER.

Tipo de retorno

INTEGER

Exemplos

Para aplicar a função DECIMAL_PRECISION à tabela t, use o exemplo a seguir.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES (3.14159);

SELECT DECIMAL_PRECISION(s) FROM t;

+-------------------+
| decimal_precision |
+-------------------+
|                 6 |
+-------------------+
```

Função DECIMAL_SCALE

Verifica o número de dígitos decimais a serem armazenados à direita do ponto. O intervalo da escala é de 0 ao ponto de precisão, com um padrão de 0.

Sintaxe

```
DECIMAL_SCALE(super_expression)
```

Argumentos

super_expression

Uma expressão ou coluna SUPER.

Tipo de retorno

INTEGER

Exemplos

Para aplicar a função DECIMAL_SCALE à tabela t, use o exemplo a seguir.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES (3.14159);

SELECT DECIMAL_SCALE(s) FROM t;

+---------------+
| decimal_scale |
+---------------+
|             5 |
+---------------+
```

Função IS_ARRAY

Confere se há uma variável em uma matriz. A função retorna `true` quando a variável é uma matriz. A função também inclui arrays vazios. Caso contrário, a função retorna `false` para todos os outros valores, incluindo nulo.

Sintaxe

```
IS_ARRAY(super_expression)
```

Argumentos

super_expression

Uma expressão ou coluna SUPER.

Tipo de retorno

BOOLEAN

Exemplos

Para verificar se `[1,2]` é uma matriz usando a função IS_ARRAY, use o exemplo a seguir.

```
SELECT IS_ARRAY(JSON_PARSE('[1,2]'));

+----------+
| is_array |
+----------+
| true     |
+----------+
```

Função IS_BIGINT

Verifica se um valor é um `BIGINT`. A função IS_BIGINT retorna `true` para números de escala 0 no intervalo de 64 bits. Caso contrário, a função retorna `false` para todos os outros valores, incluindo números nulos e de ponto flutuante.

A função IS_BIGINT é um superconjunto de IS_INTEGER.

Sintaxe

```
IS_BIGINT(super_expression)
```

Argumentos

super_expression

Uma expressão ou coluna SUPER.

Tipo de retorno

BOOLEAN

Exemplos

Para verificar se 5 é um BIGINT utilizando a função IS_BIGINT, use o exemplo a seguir.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES (5);

SELECT s, IS_BIGINT(s) FROM t;

+---+-----------+
| s | is_bigint |
+---+-----------+
| 5 | true      |
+---+-----------+
```

Função IS_BOOLEAN

Verifica se um valor é um BOOLEAN. A função IS_BOOLEAN retorna `true` para boolianos JSON constantes. A função retorna `false` para quaisquer outros valores, incluindo nulo.

Sintaxe

```
IS_BOOLEAN(super_expression)
```

Argumentos

super_expression

Uma expressão ou coluna SUPER.

Tipo de retorno

BOOLEAN

Exemplos

Para verificar se TRUE é um BOOLEAN usando a função IS_BOOLEAN, use o exemplo a seguir.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES (TRUE);

SELECT s, IS_BOOLEAN(s) FROM t;

+------+------------+
|  s   | is_boolean |
+------+------------+
| true | true       |
+------+------------+
```

Função IS_CHAR

Verifica se um valor é um CHAR. A função IS_CHAR retorna `true` para strings que têm apenas caracteres ASCII, porque o tipo CHAR pode armazenar somente caracteres que estão no formato ASCII. A função retorna `false` para quaisquer outros valores.

Sintaxe

```
IS_CHAR(super_expression)
```

Argumentos

super_expression

Uma expressão ou coluna SUPER.

Tipo de retorno

BOOLEAN

Exemplos

Para verificar se `t` é um CHAR usando a função IS_CHAR, use o exemplo a seguir.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES ('t');

SELECT s, IS_CHAR(s) FROM t;
```

```
+-----+---------+
|  s  | is_char |
+-----+---------+
| "t" | true    |
+-----+---------+
```

Função IS_DECIMAL

Verifica se um valor é um `DECIMAL`. A função IS_DECIMAL retorna `true` para números que não são pontos flutuantes. A função retorna `false` para quaisquer outros valores, incluindo nulo.

A função IS_DECIMAL é um superconjunto de IS_BIGINT.

Sintaxe

```
IS_DECIMAL(super_expression)
```

Argumentos

super_expression

Uma expressão ou coluna `SUPER`.

Tipo de retorno

`BOOLEAN`

Exemplos

Para verificar se `1.22` é um `DECIMAL` usando a função IS_DECIMAL, use o exemplo a seguir.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES (1.22);

SELECT s, IS_DECIMAL(s) FROM t;

+------+------------+
|  s   | is_decimal |
+------+------------+
| 1.22 | true       |
+------+------------+
```

Função IS_FLOAT

Verifica se um valor é um número de ponto flutuante. A função IS_FLOAT retorna `true` para números de ponto flutuante (`FLOAT4` e `FLOAT8`). A função retorna `false` para quaisquer outros valores.

O conjunto de IS_DECIMAL e o conjunto IS_FLOAT são disjuntos.

Sintaxe

```
IS_FLOAT(super_expression)
```

Argumentos

super_expression

Uma expressão ou coluna `SUPER`.

Tipo de retorno

`BOOLEAN`

Exemplos

Para verificar se `2.22::FLOAT` é um `FLOAT` usando a função IS_FLOAT, use o exemplo a seguir.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES(2.22::FLOAT);

SELECT s, IS_FLOAT(s) FROM t;

+---------+----------+
|    s    | is_float |
+---------+----------+
| 2.22e+0 | true     |
+---------+----------+
```

Função IS_INTEGER

Retorna `true` para números de escala 0 no intervalo de 32 bits, e `false` para qualquer outra coisa (incluindo números nulos e de ponto flutuante).

A função IS_INTEGER é um superconjunto da função IS_SMALLINT.

Sintaxe

```
IS_INTEGER(super_expression)
```

Argumentos

super_expression

Uma expressão ou coluna SUPER.

Tipo de retorno

BOOLEAN

Exemplos

Para verificar se 5 é um INTEGER usando a função IS_INTEGER, use o exemplo a seguir.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES (5);

SELECT s, IS_INTEGER(s) FROM t;

+---+------------+
| s | is_integer |
+---+------------+
| 5 | true       |
+---+------------+
```

Função IS_OBJECT

Verifica se uma variável é um objeto. A função IS_OBJECT retorna `true` para objetos, incluindo objetos vazios. A função retorna `false` para quaisquer outros valores, incluindo nulo.

Sintaxe

```
IS_OBJECT(super_expression)
```

Argumentos

super_expression

Uma expressão ou coluna SUPER.

Tipo de retorno

BOOLEAN

Exemplos

Para verificar se `{"name": "Joe"}` é um objeto usando a função IS_OBJECT, use o exemplo a seguir.

```
CREATE TABLE t(s super);

INSERT INTO t VALUES (JSON_PARSE('{"name": "Joe"}'));

SELECT s, IS_OBJECT(s) FROM t;

+----------------+-----------+
|       s        | is_object |
+----------------+-----------+
| {"name":"Joe"} | true      |
+----------------+-----------+
```

Função IS_SCALAR

Verifica se uma variável é escalar. A função IS_SCALAR retorna `true` para qualquer valor que não seja uma matriz ou um objeto. A função retorna `false` para quaisquer outros valores, incluindo nulo.

O conjunto de IS_ARRAY, IS_OBJECT e IS_SCALAR cobre todos os valores, exceto nulos.

Sintaxe

```
IS_SCALAR(super_expression)
```

Argumentos

super_expression

Uma expressão ou coluna SUPER.

Tipo de retorno

BOOLEAN

Exemplos

Para verificar se `{"name": "Joe"}` é um escalar usando a função IS_SCALAR, use o exemplo a seguir.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES (JSON_PARSE('{"name": "Joe"}'));

SELECT s, IS_SCALAR(s.name) FROM t;

+----------------+-----------+
|       s        | is_scalar |
+----------------+-----------+
| {"name":"Joe"} | true      |
+----------------+-----------+
```

Função IS_SMALLINT

Verifica se uma variável é `SMALLINT`. A função IS_SMALLINT retorna `true` para números de escala 0 no intervalo de 16 bits. A função retorna `false` para quaisquer outros valores, incluindo números nulos e de ponto flutuante.

Sintaxe

```
IS_SMALLINT(super_expression)
```

Argumentos

super_expression

Uma expressão ou coluna SUPER.

Return

BOOLEAN

Exemplos

Para verificar se 5 é um SMALLINT usando a função IS_SMALLINT, use o exemplo a seguir.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES (5);

SELECT s, IS_SMALLINT(s) FROM t;

+---+-------------+
| s | is_smallint |
+---+-------------+
| 5 | true        |
+---+-------------+
```

Função IS_VARCHAR

Verifica se uma variável é VARCHAR. A função IS_VARCHAR retorna `true` para todas as strings. A função retorna `false` para quaisquer outros valores.

A função IS_VARCHAR é um superconjunto da função IS_CHAR.

Sintaxe

```
IS_VARCHAR(super_expression)
```

Argumentos

super_expression

Uma expressão ou coluna SUPER.

Tipo de retorno

BOOLEAN

Exemplos

Para verificar se abc é um VARCHAR usando a função IS_VARCHAR, use o exemplo a seguir.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES ('abc');

SELECT s, IS_VARCHAR(s) FROM t;

+-------+------------+
|   s   | is_varchar |
+-------+------------+
| "abc" | true       |
+-------+------------+
```

Função JSON_SIZE

A função JSON_SIZE retorna o número de bytes na expressão SUPER fornecida quando serializada em uma string.

Sintaxe

```
JSON_SIZE(super_expression)
```

Argumentos

super_expression

Uma constante ou expressão SUPER.

Tipo de retorno

INTEGER

A função JSON_SIZE retorna um INTEGER indicando o número de bytes na string de entrada. Esse valor é diferente do número de caracteres. Por exemplo, o caractere UTF-8 #, um ponto preto, tem 3 bytes de tamanho, mesmo que tenha um caractere.

Observações de uso

JSON_SIZE (x) é funcionalmente idêntico a OCTET_LENGTH (JSON_SERIALIZE). No entanto, observe que JSON_SERIALIZE retorna um erro quando a expressão SUPER fornecida excede o limite VARCHAR do sistema quando serializada. JSON_SIZE não tem essa limitação.

Exemplos

Para retornar o tamanho de um valor SUPER serializado para uma string, use o exemplo a seguir.

```
SELECT JSON_SIZE(JSON_PARSE('[10001,10002,"#"]'));

+-----------+
| json_size |
+-----------+
|        19 |
+-----------+
```

Observe que a expressão SUPER fornecida tem 17 caracteres, mas # é um caractere de 3 bytes, então JSON_SIZE retorna 19.

Função JSON_TYPEOF

A função escalar JSON_TYPEOF retorna um VARCHAR com valores boolianos, número, string, objeto, matriz ou nulo, dependendo do tipo dinâmico do valor SUPER.

Sintaxe

```
JSON_TYPEOF(super_expression)
```

Argumentos

super_expression

Uma expressão ou coluna SUPER.

Tipo de retorno

VARCHAR

Exemplos

Para verificar o tipo de JSON da matriz [1,2] usando a função JSON_TYPEOF, use o exemplo a seguir.

```
SELECT JSON_TYPEOF(ARRAY(1,2));

+-------------+
```

```
| json_typeof |
+-------------+
| array       |
+-------------+
```

Para verificar o tipo de JSON do objeto `{"name":"Joe"}` usando a função JSON_TYPEOF, use o exemplo a seguir.

```
SELECT JSON_TYPEOF(JSON_PARSE('{"name":"Joe"}'));

+-------------+
| json_typeof |
+-------------+
| object      |
+-------------+
```

SIZE

Retorna o tamanho binário na memória de uma constante ou expressão do tipo SUPER como um INTEGER.

Sintaxe

```
SIZE(super_expression)
```

Argumentos

super_expression

Uma constante ou expressão de tipo SUPER.

Tipo de retorno

INTEGER

Exemplos

Para usar SIZE a fim de obter o tamanho na memória de várias expressões do tipo SUPER, use o exemplo a seguir.

```
CREATE TABLE test_super_size(a SUPER);
```

```
INSERT INTO test_super_size
VALUES
  (null),
  (TRUE),
  (JSON_PARSE('[0,1,2,3]')),
  (JSON_PARSE('{"a":0,"b":1,"c":2,"d":3}'))
;

SELECT a, SIZE(a)
FROM test_super_size
ORDER BY 2, 1;

+---------------------------+------+
|             a             | size |
+---------------------------+------+
| true                      |    4 |
| NULL                      |    4 |
| [0,1,2,3]                 |   23 |
| {"a":0,"b":1,"c":2,"d":3} |   52 |
+---------------------------+------+
```

Funções e operadores VARBYTE

As funções e os operadores do Amazon Redshift compatíveis com o tipo de dados VARBYTE incluem:

- Operadores VARBYTE
- FROM_HEX
- FROM_VARBYTE
- GETBIT
- TO_HEX
- TO_VARBYTE
- CONCAT
- LEN
- Função LENGTH
- OCTET_LENGTH
- Função SUBSTRING

Operadores VARBYTE

A tabela a seguir lista os operadores VARBYTE. O operador trabalha com valor binário do tipo de dados VARBYTE. Se uma ou ambas as entradas forem nulas, o resultado será null.

Operadores compatíveis

Operador	Descrição	Tipo de retorno
<	Menor que	BOOLEAN
<=	Menor ou igual a	BOOLEAN
=	Equal	BOOLEAN
>	Maior que	BOOLEAN
>=	Maior ou igual a	BOOLEAN
!= ou <>	Not equal	BOOLEAN
||	Concatenação	VARBYTE
+	Concatenação	VARBYTE
~	Bitwise not	VARBYTE
&	Bitwise and	VARBYTE
|	Bitwise or	VARBYTE
#	Bitwise xor	VARBYTE

Exemplos

Nos exemplos a seguir, o valor de `'a'::VARBYTE` é 61, e o valor de `'b'::VARBYTE` é 62. O `::` lança as strings no tipo de dados VARBYTE. Para obter mais informações sobre os tipos de dados de transmissão, consulte CAST.

Para comparar se `'a'` é menor que `'b'` usando o operador <, use o exemplo a seguir.

```
SELECT 'a'::VARBYTE < 'b'::VARBYTE AS less_than;

+-----------+
| less_than |
+-----------+
| true      |
+-----------+
```

Para comparar se `'a'` é igual a `'b'` usando o operador =, use o exemplo a seguir.

```
SELECT 'a'::VARBYTE = 'b'::VARBYTE AS equal;

+-------+
| equal |
+-------+
| false |
+-------+
```

Para concatenar dois valores binários usando o operador `||`, use o exemplo a seguir.

```
SELECT 'a'::VARBYTE || 'b'::VARBYTE AS concat;

+--------+
| concat |
+--------+
|   6162 |
+--------+
```

Para concatenar dois valores binários usando o operador +, use o exemplo a seguir.

```
SELECT 'a'::VARBYTE + 'b'::VARBYTE AS concat;

+--------+
| concat |
```

```
+--------+
|   6162 |
+--------+
```

Para negar cada bit do valor binário da entrada usando a função FROM_VARBYTE, use o exemplo a seguir. A string `'a'` é avaliada como `01100001`. Para ter mais informações, consulte FROM_VARBYTE.

```
SELECT FROM_VARBYTE(~'a'::VARBYTE, 'binary');

+--------------+
| from_varbyte |
+--------------+
|     10011110 |
+--------------+
```

Para aplicar o operador & nos dois valores binários de entrada, use o exemplo a seguir. A string `'a'` é avaliada como `01100001` e `'b'` é avaliada como `01100010`.

```
SELECT FROM_VARBYTE('a'::VARBYTE & 'b'::VARBYTE, 'binary');

+--------------+
| from_varbyte |
+--------------+
|     01100000 |
+--------------+
```

Função FROM_HEX

FROM_HEX converte hexadecimal para valor binário.

Sintaxe

```
FROM_HEX(hex_string)
```

Argumentos

hex_string

String hexadecimal de tipo de dados VARCHAR ou TEXT a ser convertida. O formato deve ser um valor literal.

Tipo de retorno

VARBYTE

Exemplos

Para converter a representação hexadecimal de '6162' para um valor binário, use o exemplo a seguir. O resultado é exibido automaticamente como a representação hexadecimal do valor binário.

```
SELECT FROM_HEX('6162');

+----------+
| from_hex |
+----------+
|     6162 |
+----------+
```

Função FROM_VARBYTE

FROM_VARBYTE converte para um valor binário em uma cadeia de caracteres no formato especificado.

Sintaxe

```
FROM_VARBYTE(binary_value, format)
```

Argumentos

binary_value

Um valor binário do tipo de dados VARBYTE.

format

O formato da cadeia de caracteres retornada. Os valores válidos que não diferenciam maiúsculas e minúsculas são hex, binary, utf8 (também utf-8 e utf_8) e base64.

Tipo de retorno

VARCHAR

Exemplos

Para converter o valor binário `'ab'` para hexadecimal, use o exemplo a seguir.

```
SELECT FROM_VARBYTE('ab', 'hex');

+--------------+
| from_varbyte |
+--------------+
|         6162 |
+--------------+
```

Para retornar a representação binária de `'4d'`, use o exemplo a seguir. A representação binária de `'4d'` é a string de caracteres `01001101`.

```
SELECT FROM_VARBYTE(FROM_HEX('4d'), 'binary');

+--------------+
| from_varbyte |
+--------------+
|     01001101 |
+--------------+
```

Função GETBIT

GETBIT retorna o valor de bit de um valor binário no índice especificado.

Sintaxe

```
GETBIT(binary_value, index)
```

Argumentos

binary_value

Um valor binário do tipo de dados VARBYTE.

índice

Um número de índice de bit no valor binário que é retornado. O valor binário é uma matriz de bits de base 0 que é indexada a partir do bit mais à direita (bit menos relevante) para o bit mais à esquerda (bit mais relevante).

Tipo de retorno

`INTEGER`

Exemplos

Para retornar o bit no índice 2 do valor binário `from_hex('4d')`, use o exemplo a seguir. A representação binária de `'4d'` é `01001101`.

```
SELECT GETBIT(FROM_HEX('4d'), 2);

+--------+
| getbit |
+--------+
|      1 |
+--------+
```

Para retornar o bit em oito locais de índice do valor binário retornado por `from_hex('4d')`, use o exemplo a seguir. A representação binária de `'4d'` é `01001101`.

```
SELECT GETBIT(FROM_HEX('4d'), 7), GETBIT(FROM_HEX('4d'), 6),
  GETBIT(FROM_HEX('4d'), 5), GETBIT(FROM_HEX('4d'), 4),
  GETBIT(FROM_HEX('4d'), 3), GETBIT(FROM_HEX('4d'), 2),
  GETBIT(FROM_HEX('4d'), 1), GETBIT(FROM_HEX('4d'), 0);

+--------+--------+--------+--------+--------+--------+--------+--------+
| getbit | getbit | getbit | getbit | getbit | getbit | getbit | getbit |
+--------+--------+--------+--------+--------+--------+--------+--------+
|      0 |      1 |      0 |      0 |      1 |      1 |      0 |      1 |
+--------+--------+--------+--------+--------+--------+--------+--------+
```

Função TO_HEX

TO_HEX converte um número ou valor binário para uma representação hexadecimal.

Sintaxe

```
TO_HEX(value)
```

Argumentos

value

Um número ou valor binário (VARBYTE) a ser convertido.

Tipo de retorno

VARCHAR

Exemplos

Para converter um número em sua representação hexadecimal, use o exemplo a seguir.

```
SELECT TO_HEX(2147676847);

+----------+
|  to_hex  |
+----------+
| 8002f2af |
+----------+
```

Para converter a representação VARBYTE de `'abc'` em um número hexadecimal, use o exemplo a seguir.

```
SELECT TO_HEX('abc'::VARBYTE);

+--------+
| to_hex |
+--------+
| 616263 |
+--------+
```

Para criar uma tabela, inserir a representação VARBYTE de `'abc'` para um número hexadecimal e selecionar a coluna com o valor, use o exemplo a seguir.

```
CREATE TABLE t (vc VARCHAR);
INSERT INTO t SELECT TO_HEX('abc'::VARBYTE);
SELECT vc FROM t;

+--------+
```

```
|   vc   |
+--------+
| 616263 |
+--------+
```

Para mostrar que quando a transmissão de um valor VARBYTE para VARCHAR o formato é UTF-8, use o exemplo a seguir.

```
CREATE TABLE t (vc VARCHAR);
INSERT INTO t SELECT 'abc'::VARBYTE::VARCHAR;

SELECT vc FROM t;

+-----+
| vc  |
+-----+
| abc |
+-----+
```

Função TO_VARBYTE

TO_VARBYTE converte uma string no formato especificado para um valor binário.

Sintaxe

```
TO_VARBYTE(string, format)
```

Argumentos

string

Uma string CHAR ou VARCHAR.

format

O formato da string de entrada. Os valores válidos que não diferenciam maiúsculas e minúsculas são hex, binary, utf8 (também utf-8 e utf_8) e base64.

Tipo de retorno

VARBYTE

Exemplos

Para converter o hex 6162 em um valor binário, use o exemplo a seguir. O resultado é exibido automaticamente como a representação hexadecimal do valor binário.

```
SELECT TO_VARBYTE('6162', 'hex');

+------------+
| to_varbyte |
+------------+
|       6162 |
+------------+
```

Para retornar a representação binária de 4d, use o exemplo a seguir. A representação binária de '4d' é 01001101.

```
SELECT TO_VARBYTE('01001101', 'binary');

+------------+
| to_varbyte |
+------------+
|         4d |
+------------+
```

Para converter a string 'a' em UTF-8 em um valor binário, use o exemplo a seguir. O resultado é exibido automaticamente como a representação hexadecimal do valor binário.

```
SELECT TO_VARBYTE('a', 'utf8');

+------------+
| to_varbyte |
+------------+
|         61 |
+------------+
```

Para converter a string '4' em hexadecimal em um valor binário, use o exemplo a seguir. Se o comprimento da string hexadecimal for um número ímpar, adiciona-se um 0 para formar um número hexadecimal válido.

```
SELECT TO_VARBYTE('4', 'hex');
```

```
+------------+
| to_varbyte |
+------------+
|         04 |
+------------+
```

Funções de janela

Usando funções da janela, é possível criar consultas analíticas empresariais de forma mais eficiente. Funções de janela operam em uma partição ou "janela" de um conjunto de resultados e retornam um valor para cada linha naquela janela. Por outro lado, funções sem janela executam seus cálculos em relação a cada linha no conjunto de resultados. Diferente de funções de grupo que agregam linhas de resultado, as funções de janela retêm todas as linhas na expressão da tabela.

Os valores retornados são calculados usando valores dos conjuntos de linhas dessa janela. Para cada linha da tabela, a janela define um conjunto de linhas que é usado para computar atributos adicionais. Um janela é definida usando uma especificação de janela (a cláusula OVER) se baseia em três conceitos principais:

- Particionamento da janela, que forma grupos de linhas (cláusula PARTITION)
- Ordenação da janela, que define uma ordem ou sequência de linhas dentro de cada partição (cláusula ORDER BY)
- Quadros da janela, que são definidos em relação a cada linha para restringir ainda mais o conjunto de linhas (especificação de ROWS)

As funções da janela são o último conjunto de operações executadas em uma consulta, exceto pela cláusula ORDER BY final. Todas as junções e todas as cláusulas WHERE, GROUP BY e HAVING são concluídas antes do processamento das funções da janela. Portanto, as funções da janela podem aparecer somente na lista de seleção ou na cláusula ORDER BY. Você pode usar várias funções da janela em uma única consulta com diferentes cláusulas de quadro. Você também pode usar funções da janela em outras expressões escalares, tal como CASE.

As funções da janela não podem ser aninhadas. Por exemplo, uma função agregada SUM pode aparecer dentro de uma função de janela SUM, mas uma função de janela SUM não pode aparecer dentro de outra função de janela SUM. O seguinte não é possível porque uma função de janela está aninhada em outra função de janela.

```
SELECT SUM(SUM(selectcol) OVER (PARTITION BY ordercol)) OVER (Partition by ordercol)
 FROM t;
```

Resumo da sintaxe de funções da janela

As funções de janela seguem uma sintaxe padrão, mostrada a seguir.

```
function (expression) OVER (
[ PARTITION BY expr_list ]
[ ORDER BY order_list [ frame_clause ] ] )
```

Aqui, function é uma das funções descritas nesta seção.

A expr_list é como indicado a seguir.

```
expression | column_name [, expr_list ]
```

A order_list é como a seguir.

```
expression | column_name [ ASC | DESC ]
[ NULLS FIRST | NULLS LAST ]
[, order_list ]
```

O frame_clause é como a seguir.

```
ROWS
{ UNBOUNDED PRECEDING | unsigned_value PRECEDING | CURRENT ROW } |

{ BETWEEN
{ UNBOUNDED PRECEDING | unsigned_value { PRECEDING | FOLLOWING } | CURRENT ROW}
AND
{ UNBOUNDED FOLLOWING | unsigned_value { PRECEDING | FOLLOWING } | CURRENT ROW }}
```

Argumentos

Função

A função de janela. Para obter detalhes, consulte as descrições individuais da função.

OVER

A cláusula que define a especificação da janela. A cláusula OVER é obrigatória para funções da janela e diferencia funções da janela de outras funções SQL.

PARTITION BY expr_list

(Opcional) A cláusula PARTITION BY subdivide o conjunto de resultados em partições, bem como a cláusula GROUP BY. Se uma cláusula de partição estiver presente, a função será calculada para as linhas em cada partição. Se nenhuma cláusula de partição estiver especificada, uma única partição contém a tabela inteira e a função é computada para esta tabela completa.

As funções de classificação DENSE_RANK, NTILE, RANK e ROW_NUMBER exigem uma comparação global de todas as linhas no conjunto de resultados. Quando uma cláusula PARTITION BY é utilizada, o otimizador de consulta pode executar cada agregação em paralelo, distribuindo a workload em várias fatias de acordo com as partições. Se a cláusula PARTITION BY não estiver presente, a etapa de agregação deverá ser executada em série em uma única fatia, o que poderá ter um impacto negativo considerável na performance, sobretudo para grandes clusters.

O Amazon Redshift não oferece suporte a literais de string nas cláusulas PARTITION BY.

ORDER BY order_list

(Opcional) A função da janela é aplicada às linhas dentro de cada partição classificada de acordo com a especificação do pedido em ORDER BY. Esta cláusula ORDER BY é diferente e totalmente não relacionada a uma cláusula ORDER BY na frame_clause. A cláusula ORDER BY pode ser usada sem a cláusula PARTITION BY.

Para as funções de classificação, a cláusula ORDER BY identifica as medidas para os valores de classificação. Para funções de agregação, as linhas particionadas devem ser ordenadas antes que a função agregada seja computada para cada quadro. Para obter mais informações sobre os tipos de função da janela, consulte Funções de janela.

Os identificadores de coluna ou expressões que avaliam os identificadores de coluna são obrigatórios na lista de ordenação. Nem constantes ou expressões constantes podem ser usadas como substitutos para nomes de coluna.

Valores NULL são tratados como seu próprio grupo, ordenados e classificados de acordo com a opção NULLS FIRST ou NULLS LAST. Por padrão, os valores NULL são ordenados e classificados por último na ordem ASC e são ordenados e classificados primeiro na ordem DESC.

O Amazon Redshift não oferece suporte a literais de string nas cláusulas ORDER BY.

Se a cláusula ORDER BY for omitida, a ordem das linhas não será determinística.

> **Note**
>
> Em qualquer sistema paralelo, como o Amazon Redshift, quando uma cláusula ORDER BY não produz uma ordem única e total dos dados, a ordem das linhas não é determinística. Ou seja, se a expressão ORDER BY produzir valores duplicados (uma ordenação parcial), a ordem de retorno dessas linhas pode variar de uma execução do Amazon Redshift para a próxima. Por sua vez, funções da janela podem retornar resultados inesperados ou inconsistentes. Para obter mais informações, consulte Ordenação exclusiva de dados para funções da janela.

column_name

Nome de uma coluna a ser particionada por ou ordenada por.

ASC | DESC

Opção que define a ordem de classificação para a expressão, da seguinte forma:

- ASC: ascendente (por exemplo, de valores numéricos menores para maiores e de "A" a "Z" para strings de caracteres). Se nenhuma opção é especificada, os dados são classificados na ordem ascendente por padrão.
- DESC: descendente (de valores numéricos maiores para menores; de "Z" a "A" para strings).

NULLS FIRST | NULLS LAST

Opção que especifica se NULLS devem ser ordenados primeiro, antes de valores não nulos, ou por último, após valores não nulos. Por padrão, NULLs são ordenados e classificados por último na ordem ASC e ordenados e classificados primeiro na ordem DESC.

frame_clause

Para funções agregadas, a cláusula do quadro refina ainda mais o conjunto de linhas na janela de uma função ao usar ORDER BY. Com a cláusula de quadro, é possível incluir ou excluir conjuntos de linhas no resultado ordenado. A cláusula de quadro consiste na palavra-chave ROWS e nos especificadores associados.

A cláusula frame não se aplica a funções de classificação. Além disso, a cláusula frame não é necessária quando nenhuma cláusula ORDER BY é usada na cláusula OVER para uma função

agregada. Se uma cláusula ORDER BY é usada para uma função agregada, uma cláusula de quadro explícita é necessária.

Quando nenhuma cláusula ORDER BY é especificada, o quadro implícito é ilimitado, equivalente a ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING.

ROWS

Especificando um deslocamento físico da linha atual especificando um deslocamento físico da linha atual.

Essa cláusula especifica as linhas na janela ou particionamento atual ao qual o valor da linha atual dever ser combinado. Ela usa os argumentos que especificam a posição da linha, que pode ser antes ou depois da linha atual. O ponto de referência para todos os quadros de janela é a linha atual. Cada linha se torna a linha atual, por sua vez, à medida que o quadro de janela avança pela partição.

O quadro pode ser um conjunto simples de linhas até e incluindo a linha atual.

```
{UNBOUNDED PRECEDING | offset PRECEDING | CURRENT ROW}
```

Ou pode ser um conjunto de linhas entre dois limites.

```
BETWEEN
{ UNBOUNDED PRECEDING | offset { PRECEDING | FOLLOWING } | CURRENT ROW }
AND
{ UNBOUNDED FOLLOWING | offset { PRECEDING | FOLLOWING } | CURRENT ROW }
```

UNBOUNDED PRECEDING indica que a janela começa na primeira linha da partição; deslocamento PRECEDING indica que a janela começa um número de linhas equivalentes ao valor do deslocamento antes da linha atual. UNBOUNDED PRECEDING é o padrão.

CURRENT ROW indica que a janela começa ou termina na linha atual.

UNBOUNDED FOLLOWING indica que a janela termina na última linha da partição; deslocamento FOLLOWING indica que a janela termina um número de linhas equivalentes ao valor do deslocamento depois da linha atual.

O offset identifica um número físico de linhas antes ou depois da linha atual. Nesse caso, o deslocamento deve ser uma constante que retorna um valor numérico positivo. Por exemplo, 5 FOLLOWING termina o quadro 5 linhas após a linha atual.

Onde BETWEEN não é especificado, o quadro é limitado implicitamente pela linha atual. Por exemplo, `ROWS 5 PRECEDING` é igual a `ROWS BETWEEN 5 PRECEDING AND CURRENT ROW`. Além disso, `ROWS UNBOUNDED FOLLOWING` é igual a `ROWS BETWEEN CURRENT ROW AND UNBOUNDED FOLLOWING`.

> **Note**
>
> Você não pode especificar um quadro em que o limite inicial seja maior do que o limite final. Por exemplo, você não pode especificar nenhum dos quadros a seguir.
>
> ```
> between 5 following and 5 preceding
> between current row and 2 preceding
> between 3 following and current row
> ```

Ordenação exclusiva de dados para funções da janela

Se uma cláusula ORDER BY para uma função da janela não produz uma ordem única e total dos dados, a ordem das linhas não é determinística. Se a expressão ORDER BY produzir valores duplicados (uma ordenação parcial), a ordem de retorno dessas linhas pode variar em várias execuções. Nesse caso, as funções da janela também podem retornar resultados inesperados ou inconsistentes.

Por exemplo, a consulta a seguir retorna resultados diferentes ao longo de várias execuções. Esses resultados diferentes ocorrem porque `order by dateid` não produz uma ordenação exclusiva dos dados para a função da janela SUM.

```
select dateid, pricepaid,
sum(pricepaid) over(order by dateid rows unbounded preceding) as sumpaid
from sales
group by dateid, pricepaid;

dateid | pricepaid |   sumpaid
--------+-----------+-------------
1827 |   1730.00 |     1730.00
1827 |    708.00 |     2438.00
1827 |    234.00 |     2672.00
...

select dateid, pricepaid,
```

```
sum(pricepaid) over(order by dateid rows unbounded preceding) as sumpaid
from sales
group by dateid, pricepaid;

dateid | pricepaid |   sumpaid
--------+-----------+-------------
1827 |    234.00 |      234.00
1827 |    472.00 |      706.00
1827 |    347.00 |     1053.00
...
```

Nesse caso, adicionar uma segunda coluna ORDER BY à função da janela pode resolver o problema.

```
select dateid, pricepaid,
sum(pricepaid) over(order by dateid, pricepaid rows unbounded preceding) as sumpaid
from sales
group by dateid, pricepaid;

dateid | pricepaid | sumpaid
--------+-----------+---------
1827 |    234.00 |  234.00
1827 |    337.00 |  571.00
1827 |    347.00 |  918.00
...
```

Funções compatíveis

O Amazon Redshift oferece suporte a dois tipos de funções da janela: agregação e classificação.

Veja a seguir as funções agregadas compatíveis:

- Função de janela AVG
- Função de janela COUNT
- Função de janela CUME_DIST
- Função de janela DENSE_RANK
- Função de janela FIRST_VALUE
- Função de janela LAG
- Função de janela LAST_VALUE
- Função de janela LEAD

- Função de janela LISTAGG
- Função de janela MAX
- Função de janela MEDIAN
- Função de janela MIN
- Função de janela NTH_VALUE
- Função de janela PERCENTILE_CONT
- Função de janela PERCENTILE_DISC
- Função de janela RATIO_TO_REPORT
- Funções de janela STDDEV_SAMP e STDDEV_POP (STDDEV_SAMP e STDDEV são sinônimos)
- Função de janela SUM
- Funções de janela VAR_SAMP e VAR_POP (VAR_SAMP e VARIANCE são sinônimos)

Veja a seguir as funções de classificação compatíveis:

- Função de janela DENSE_RANK
- Função de janela NTILE
- Função de janela PERCENT_RANK
- Função de janela RANK
- Função de janela ROW_NUMBER

Amostra de tabela para exemplos de funções de janela

É possível encontrar exemplos de função de janela específicos com cada descrição de função. Alguns dos exemplos usam uma tabela chamada WINSALES, que contém 11 linhas, conforme mostrado a seguir.

SALESID	DATEID	SELLERID	BUYERID	QTY	QTY_SHIPPED
30001	8/2/2003	3	B	10	10
10001	12/24/2003	1	C	10	10
10005	12/24/2003	1	A	30	

SALESID	DATEID	SELLERID	BUYERID	QTY	QTY_SHIPPED
40001	1/9/2004	4	A	40	
10006	1/18/2004	1	C	10	
20001	2/12/2004	2	B	20	20
40005	2/12/2004	4	A	10	10
20002	2/16/2004	2	C	20	20
30003	4/18/2004	3	B	15	
30004	4/18/2004	3	B	20	
30007	9/7/2004	3	C	30	

O seguinte script cria e preenche a tabela de amostra WINSALES.

```
CREATE TABLE winsales(
  salesid int,
  dateid date,
  sellerid int,
  buyerid char(10),
  qty int,
  qty_shipped int);

INSERT INTO winsales VALUES
  (30001, '8/2/2003', 3, 'b', 10, 10),
  (10001, '12/24/2003', 1, 'c', 10, 10),
  (10005, '12/24/2003', 1, 'a', 30, null),
  (40001, '1/9/2004', 4, 'a', 40, null),
  (10006, '1/18/2004', 1, 'c', 10, null),
  (20001, '2/12/2004', 2, 'b', 20, 20),
  (40005, '2/12/2004', 4, 'a', 10, 10),
  (20002, '2/16/2004', 2, 'c', 20, 20),
  (30003, '4/18/2004', 3, 'b', 15, null),
  (30004, '4/18/2004', 3, 'b', 20, null),
  (30007, '9/7/2004', 3, 'c', 30, null);
```

Função de janela AVG

A função de janela AVG retorna a média (meio aritmético) dos valores de expressão de entrada. A função AVG funciona com valores numéricos e ignora valores NULL.

Sintaxe

```
AVG ( [ALL ] expression ) OVER
(
[ PARTITION BY expr_list ]
[ ORDER BY order_list
                        frame_clause ]
)
```

Argumentos

expressão

A coluna ou expressão de destino na qual a função opera.

ALL

Com o argumento ALL, a função retém todos os valores duplicados da expressão para contagem. ALL é o padrão. DISTINCT não é compatível.

OVER

Especifica as cláusulas de janela das funções de agregação. A cláusula OVER distingue funções de agregação de janela das funções de agregação de conjuntos normais.

PARTITION BY expr_list

Define a janela para a função AVG em termos de uma ou mais expressões.

ORDER BY order_list

Classifica as linhas dentro de cada partição. Se nenhuma PARTITION BY for especificada, ORDER BY usa a tabela completa.

frame_clause

Se uma cláusula ORDER BY é usada para uma função agregada, uma cláusula de quadro explícita é necessária. A cláusula de quadro refina o conjunto de linhas na janela de uma função,

incluindo ou excluindo conjuntos de linhas no resultado ordenado. A cláusula de quadro consiste na palavra-chave ROWS e nos especificadores associados. Consulte Resumo da sintaxe de funções da janela.

Tipos de dados

Os tipos de argumentos compatíveis com a função AVG são SMALLINT, INTEGER, BIGINT, NUMERIC, DECIMAL, REAL e DOUBLE PRECISION.

Os tipos de retorno compatíveis com a função AVG são:

- BIGINT para argumentos SMALLINT ou INTEGER
- NUMERIC para argumentos BIGINT
- DOUBLE PRECISION para argumentos de ponto flutuante

Exemplos

O seguinte exemplo calcula uma média móvel das quantidades vendidas por data; ordene os resultados por ID de data e ID de vendas:

```
select salesid, dateid, sellerid, qty,
avg(qty) over
(order by dateid, salesid rows unbounded preceding) as avg
from winsales
order by 2,1;

salesid |   dateid   | sellerid | qty | avg
---------+------------+----------+-----+-----
30001 | 2003-08-02 |        3 |  10 |  10
10001 | 2003-12-24 |        1 |  10 |  10
10005 | 2003-12-24 |        1 |  30 |  16
40001 | 2004-01-09 |        4 |  40 |  22
10006 | 2004-01-18 |        1 |  10 |  20
20001 | 2004-02-12 |        2 |  20 |  20
40005 | 2004-02-12 |        4 |  10 |  18
20002 | 2004-02-16 |        2 |  20 |  18
30003 | 2004-04-18 |        3 |  15 |  18
30004 | 2004-04-18 |        3 |  20 |  18
30007 | 2004-09-07 |        3 |  30 |  19
(11 rows)
```

Para uma descrição da tabela WINSALES, consulte Amostra de tabela para exemplos de funções de janela.

Função de janela COUNT

A função de janela COUNT conta as linhas definidas pela expressão.

A função COUNT tem duas variações. COUNT (*) conta todas as linhas na tabela de destino independente se elas contêm nulls ou não. COUNT (expressão) computa o número de linhas com valores não NULL em uma coluna ou expressão específica.

Sintaxe

```
COUNT ( * | [ ALL ] expression) OVER
(
[ PARTITION BY expr_list ]
[ ORDER BY order_list
                        frame_clause ]
)
```

Argumentos

expressão

A coluna ou expressão de destino na qual a função opera.

ALL

Com o argumento ALL, a função retém todos os valores duplicados da expressão para contagem. ALL é o padrão. DISTINCT não é compatível.

OVER

Especifica as cláusulas de janela das funções de agregação. A cláusula OVER distingue funções de agregação de janela das funções de agregação de conjuntos normais.

PARTITION BY expr_list

Define a janela para a função COUNT em termos de uma ou mais expressões.

ORDER BY order_list

Classifica as linhas dentro de cada partição. Se nenhuma PARTITION BY for especificada, ORDER BY usa a tabela completa.

frame_clause

Se uma cláusula ORDER BY é usada para uma função agregada, uma cláusula de quadro explícita é necessária. A cláusula de quadro refina o conjunto de linhas na janela de uma função, incluindo ou excluindo conjuntos de linhas no resultado ordenado. A cláusula de quadro consiste na palavra-chave ROWS e nos especificadores associados. Consulte Resumo da sintaxe de funções da janela.

Tipos de dados

A função COUNT é compatível com todos os tipos de dados de argumento.

O tipo de retorno compatível com a função COUNT é BIGINT.

Exemplos

O exemplo a seguir mostra o ID de vendas, a quantidade e a contagem de todas as linhas desde o início da janela de dados:

```
select salesid, qty,
count(*) over (order by salesid rows unbounded preceding) as count
from winsales
order by salesid;

salesid | qty | count
---------+-----+-----
10001 |  10 |   1
10005 |  30 |   2
10006 |  10 |   3
20001 |  20 |   4
20002 |  20 |   5
30001 |  10 |   6
30003 |  15 |   7
30004 |  20 |   8
30007 |  30 |   9
40001 |  40 |   10
40005 |  10 |   11
(11 rows)
```

Para uma descrição da tabela WINSALES, consulte Amostra de tabela para exemplos de funções de janela.

O exemplo a seguir mostra como o ID de vendas, a quantidade e a contagem de linhas não nulas desde o início da janela de dados. (Na tabela WINSALES, a coluna QTY_SHIPPED contém alguns NULLs.)

```
select salesid, qty, qty_shipped,
count(qty_shipped)
over (order by salesid rows unbounded preceding) as count
from winsales
order by salesid;

salesid | qty | qty_shipped | count
---------+-----+-------------+-------
10001 |  10 |          10 |   1
10005 |  30 |             |   1
10006 |  10 |             |   1
20001 |  20 |          20 |   2
20002 |  20 |          20 |   3
30001 |  10 |          10 |   4
30003 |  15 |             |   4
30004 |  20 |             |   4
30007 |  30 |             |   4
40001 |  40 |             |   4
40005 |  10 |          10 |   5
(11 rows)
```

Função de janela CUME_DIST

Calcula a distribuição cumulativa de um valor em uma janela ou partição. Assumindo uma ordem ascendente, a distribuição cumulativa é determinada usando esta fórmula:

```
count of rows with values <= x / count of rows in the window or partition
```

onde x é igual ao valor na linha atual da coluna especificada na cláusula ORDER BY. O seguinte conjunto de dados ilustra O uso desta fórmula:

```
Row# Value   Calculation    CUME_DIST
1        2500    (1)/(5)    0.2
2        2600    (2)/(5)    0.4
3        2800    (3)/(5)    0.6
4        2900    (4)/(5)    0.8
5        3100    (5)/(5)    1.0
```

O intervalo de valor de retorno é >0 a 1, inclusive.

Sintaxe

```
CUME_DIST ()
OVER (
[ PARTITION BY partition_expression ]
[ ORDER BY order_list ]
)
```

Argumentos

OVER

Uma cláusula que especifica o particionamento da janela. A cláusula OVER não pode conter uma especificação de quadro da janela.

PARTITION BY partition_expression

Opcional. Uma expressão que define o intervalo de registros para cada grupo na cláusula OVER.

ORDER BY order_list

A expressão na qual calcular a distribuição cumulativa. A expressão deve ter um tipo de dados numérico ou ser implicitamente conversível para um. Se ORDER BY for omitida, o valor de retorno será 1 para todas as linhas.

Se ORDER BY não produzir uma ordem única, a ordem das linhas não é determinística. Para obter mais informações, consulte Ordenação exclusiva de dados para funções da janela.

Tipo de retorno

FLOAT8

Exemplos

O seguinte exemplo calcula a distribuição cumulativa da quantidade para cada vendedor:

```
select sellerid, qty, cume_dist()
over (partition by sellerid order by qty)
from winsales;

sellerid   qty    cume_dist
```

```
-----------------------------------------------
1          10.00    0.33
1          10.64    0.67
1          30.37    1
3          10.04    0.25
3          15.15    0.5
3          20.75    0.75
3          30.55    1
2          20.09    0.5
2          20.12    1
4          10.12    0.5
4          40.23    1
```

Para uma descrição da tabela WINSALES, consulte Amostra de tabela para exemplos de funções de janela.

Função de janela DENSE_RANK

A função de janela DENSE_RANK determina a classificação de um valor em um grupo de valores com base na expressão ORDER BY da cláusula OVER. Se a cláusula opcional PARTITION BY estiver presente, as classificações são redefinidas para cada grupo de linhas. Linhas com valores iguais para os critérios de classificação recebem a mesma classificação. A função DENSE_RANK difere de RANK em um aspecto: se duas ou mais linhas empatarem, não há uma lacuna na sequência de valores classificados. Por exemplo, se duas linhas são classificadas como 1, a classificação seguinte é 2.

Você pode ter funções de classificação com diferentes cláusulas PARTITION BY e ORDER BY na mesma consulta.

Sintaxe

```
DENSE_RANK() OVER
(
[ PARTITION BY expr_list ]
[ ORDER BY order_list ]
)
```

Argumentos

( )

A função não aceita argumentos, mas os parênteses vazios são necessários.

OVER

As cláusulas de janela para a função DENSE_RANK.

PARTITION BY expr_list

(Opcional) Uma ou várias expressões que definem a janela.

ORDER BY order_list

(Opcional) A expressão na qual os valores de classificação se baseiam. Se nenhuma PARTITION BY for especificada, ORDER BY usa a tabela completa. Se ORDER BY for omitida, o valor de retorno será 1 para todas as linhas.

Se ORDER BY não produzir uma ordem única, a ordem das linhas não é determinística. Para obter mais informações, consulte Ordenação exclusiva de dados para funções da janela.

Tipo de retorno

`BIGINT`

Exemplos

Os exemplos a seguir usam a tabela de amostra para funções de janela. Para obter mais informações, consulte Amostra de tabela para exemplos de funções de janela.

O exemplo a seguir ordena a tabela pela quantidade vendida e atribui uma classificação densa e uma classificação regular a cada linha. Os resultados são classificados após a aplicação dos resultados da função de janela.

```
SELECT salesid, qty,
DENSE_RANK() OVER(ORDER BY qty DESC) AS d_rnk,
RANK() OVER(ORDER BY qty DESC) AS rnk
FROM winsales
ORDER BY 2,1;

+---------+-----+-------+-----+
| salesid | qty | d_rnk | rnk |
+---------+-----+-------+-----+
|   10001 |  10 |     5 |   8 |
|   10006 |  10 |     5 |   8 |
|   30001 |  10 |     5 |   8 |
```

```
|   40005 |  10 |     5 |   8 |
|   30003 |  15 |     4 |   7 |
|   20001 |  20 |     3 |   4 |
|   20002 |  20 |     3 |   4 |
|   30004 |  20 |     3 |   4 |
|   10005 |  30 |     2 |   2 |
|   30007 |  30 |     2 |   2 |
|   40001 |  40 |     1 |   1 |
+---------+-----+-------+-----+
```

Observe a diferença nas classificações atribuídas ao mesmo conjunto de linhas quando as funções DENSE_RANK e RANK são usadas lado a lado na mesma consulta.

O exemplo a seguir particiona a tabela por sellerid, ordena cada partição pela quantidade e atribui uma classificação densa a cada linha. Os resultados são classificados após a aplicação dos resultados da função de janela.

```
SELECT salesid, sellerid, qty,
DENSE_RANK() OVER(PARTITION BY sellerid ORDER BY qty DESC) AS d_rnk
FROM winsales
ORDER BY 2,3,1;

+---------+----------+-----+-------+
| salesid | sellerid | qty | d_rnk |
+---------+----------+-----+-------+
|   10001 |        1 |  10 |     2 |
|   10006 |        1 |  10 |     2 |
|   10005 |        1 |  30 |     1 |
|   20001 |        2 |  20 |     1 |
|   20002 |        2 |  20 |     1 |
|   30001 |        3 |  10 |     4 |
|   30003 |        3 |  15 |     3 |
|   30004 |        3 |  20 |     2 |
|   30007 |        3 |  30 |     1 |
|   40005 |        4 |  10 |     2 |
|   40001 |        4 |  40 |     1 |
+---------+----------+-----+-------+
```

Para utilizar o último exemplo com sucesso, use o comando a seguir para inserir uma linha na tabela WINSALES. Essa linha tem o mesmo buyerid, sellerid e qtysold de outra linha. Isso fará com que duas linhas empatem no último exemplo e, assim, mostrará a diferença entre as funções DENSE_RANK e RANK.

```
INSERT INTO winsales VALUES(30009, '2/2/2003', 3, 'b', 20, NULL);
```

O exemplo a seguir particiona a tabela por buyerid e sellerid, ordena cada partição pela quantidade e atribui uma classificação densa e uma classificação regular a cada linha. Os resultados são classificados após a aplicação da função de janela.

```
SELECT salesid, sellerid, qty, buyerid,
DENSE_RANK() OVER(PARTITION BY buyerid, sellerid ORDER BY qty DESC) AS d_rnk,
RANK() OVER (PARTITION BY buyerid, sellerid ORDER BY qty DESC) AS rnk
FROM winsales
ORDER BY rnk;

+---------+----------+-----+---------+-------+-----+
| salesid | sellerid | qty | buyerid | d_rnk | rnk |
+---------+----------+-----+---------+-------+-----+
|   20001 |        2 |  20 | b       |     1 |   1 |
|   30007 |        3 |  30 | c       |     1 |   1 |
|   10006 |        1 |  10 | c       |     1 |   1 |
|   10005 |        1 |  30 | a       |     1 |   1 |
|   20002 |        2 |  20 | c       |     1 |   1 |
|   30009 |        3 |  20 | b       |     1 |   1 |
|   40001 |        4 |  40 | a       |     1 |   1 |
|   30004 |        3 |  20 | b       |     1 |   1 |
|   10001 |        1 |  10 | c       |     1 |   1 |
|   40005 |        4 |  10 | a       |     2 |   2 |
|   30003 |        3 |  15 | b       |     2 |   3 |
|   30001 |        3 |  10 | b       |     3 |   4 |
+---------+----------+-----+---------+-------+-----+
```

Função de janela FIRST_VALUE

Considerando um conjunto de linhas ordenado, FIRST_VALUE retorna o valor da expressão especificada em relação à primeira linha no quadro de janela.

Para obter informações sobre como selecionar a última linha no quadro, consulte Função de janela LAST_VALUE.

Sintaxe

```
FIRST_VALUE( expression )[ IGNORE NULLS | RESPECT NULLS ]
OVER (
[ PARTITION BY expr_list ]
```

```
[ ORDER BY order_list frame_clause ]
)
```

Argumentos

expressão

A coluna ou expressão de destino na qual a função opera.

IGNORE NULLS

Quando essa opção é usada com FIRST_VALUE, a função retorna o primeiro valor no quadro que não seja NULL (ou NULL se todos os valores forem NULL).

RESPECT NULLS

Indica que o Amazon Redshift deve incluir valores nulos na determinação de qual linha usar. RESPECT NULLS é compatível por padrão se você não especificar IGNORE NULLS.

OVER

Introduz as cláusulas de janela para a função.

PARTITION BY expr_list

Define a janela para a função em termos de uma ou mais expressões.

ORDER BY order_list

Classifica as linhas dentro de cada partição. Se nenhuma cláusula PARTITION BY for especificada, ORDER BY classifica a tabela inteira. Se você especificar uma cláusula ORDER BY, você também deve especificar uma frame_clause.

Os resultados da função FIRST_VALUE dependem da ordem dos dados. Os resultados são não determinísticos nos seguintes casos:

- Quando uma cláusula ORDER BY é especificada e uma partição contém dois valores diferentes para uma expressão
- Quando uma expressão avalia para valores diferentes que correspondem ao mesmo valor na lista ORDER BY.

frame_clause

Se uma cláusula ORDER BY é usada para uma função agregada, uma cláusula de quadro explícita é necessária. A cláusula de quadro refina o conjunto de linhas na janela de uma função, incluindo ou excluindo conjuntos de linhas no resultado ordenado. A cláusula de quadro consiste

na palavra-chave ROWS e nos especificadores associados. Consulte Resumo da sintaxe de funções da janela.

Tipo de retorno

Essas funções são compatíveis com expressões que usam tipos de dados primitivos do Amazon Redshift. O tipo de retorno é igual ao tipo de dados da expressão.

Exemplos

Os exemplos a seguir usam a tabela VENUE dos dados de amostra TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

O seguinte exemplo retorna a capacidade de acomodação para cada local de evento da tabela VENUE com os resultados ordenados por capacidade (alta a baixa). A função FIRST_VALUE é usada para selecionar o local de evento que corresponde à primeira linha no quadro: nesse caso, a linha com o mais alto número de assentos. Os resultados são particionados por estado, portanto quando o valor VENUESTATE muda, um novo primeiro valor é selecionado. O quadro da janela não é vinculado, portanto o mesmo primeiro valor é selecionado para cada linha em cada partição.

Para a Califórnia, `Qualcomm Stadium` tem mais alto número de assentos (`70561`), portanto esse nome é o primeiro valor para todas as linhas da partição `CA`.

```
select venuestate, venueseats, venuename,
first_value(venuename)
over(partition by venuestate
order by venueseats desc
rows between unbounded preceding and unbounded following)
from (select * from venue where venueseats >0)
order by venuestate;

venuestate | venueseats |          venuename            |         first_value
-----------+------------+-------------------------------
+-----------------------------
CA         |      70561 | Qualcomm Stadium              | Qualcomm Stadium
CA         |      69843 | Monster Park                  | Qualcomm Stadium
CA         |      63026 | McAfee Coliseum               | Qualcomm Stadium
CA         |      56000 | Dodger Stadium                | Qualcomm Stadium
CA         |      45050 | Angel Stadium of Anaheim      | Qualcomm Stadium
CA         |      42445 | PETCO Park                    | Qualcomm Stadium
CA         |      41503 | AT&T Park                     | Qualcomm Stadium
CA         |      22000 | Shoreline Amphitheatre        | Qualcomm Stadium
```

```
CO         |      76125 | INVESCO Field                 | INVESCO Field
CO         |      50445 | Coors Field                   | INVESCO Field
DC         |      41888 | Nationals Park                | Nationals Park
FL         |      74916 | Dolphin Stadium               | Dolphin Stadium
FL         |      73800 | Jacksonville Municipal Stadium | Dolphin Stadium
FL         |      65647 | Raymond James Stadium         | Dolphin Stadium
FL         |      36048 | Tropicana Field               | Dolphin Stadium
...
```

O seguinte exemplo mostra o uso da opção IGNORE NULLS e depende da inclusão de uma nova linha na tabela VENUE:

```
insert into venue values(2000,null,'Stanford','CA',90000);
```

Essa nova linha contém um valor NULL para a coluna VENUENAME. Agora, repita a consulta para FIRST_VALUE que foi mostrada anteriormente nesta seção:

```
select venuestate, venueseats, venuename,
first_value(venuename)
over(partition by venuestate
order by venueseats desc
rows between unbounded preceding and unbounded following)
from (select * from venue where venueseats >0)
order by venuestate;

venuestate | venueseats |         venuename          | first_value
-----------+------------+----------------------------+-------------
CA         |      90000 | NULL                       | NULL
CA         |      70561 | Qualcomm Stadium           | NULL
CA         |      69843 | Monster Park               | NULL
...
```

Como a nova linha contém o valor mais alto de VENUESEATS (90000) e o VENUENAME é NULL, a função FIRST_VALUE retorna o NULL para a partição CA. Para ignorar linhas como essa na avaliação da função, adicione a opção IGNORE NULLS ao argumento da função:

```
select venuestate, venueseats, venuename,
first_value(venuename) ignore nulls
over(partition by venuestate
order by venueseats desc
rows between unbounded preceding and unbounded following)
from (select * from venue where venuestate='CA')
```

```
order by venuestate;

venuestate | venueseats |         venuename          |   first_value
-----------+------------+----------------------------+-----------------
CA         |      90000 | NULL                       | Qualcomm Stadium
CA         |      70561 | Qualcomm Stadium           | Qualcomm Stadium
CA         |      69843 | Monster Park               | Qualcomm Stadium
...
```

Função de janela LAG

A função de janela LAG retorna os valores para uma linha em determinado deslocamento acima (antes) da linha atual na partição.

Sintaxe

```
LAG (value_expr [, offset ])
[ IGNORE NULLS | RESPECT NULLS ]
OVER ( [ PARTITION BY window_partition ] ORDER BY window_ordering )
```

Argumentos

value_expr

A coluna ou expressão de destino na qual a função opera.

deslocamento

Um parâmetro opcional que especifica o número de linhas antes da linha atual para as quais retornar valores. Este deslocamento pode ser um inteiro constante ou uma expressão que avalia para um inteiro. Se você não especificar um deslocamento, o Amazon Redshift usará 1 como o valor padrão. Um deslocamento de `0` indica a linha atual.

IGNORE NULLS

Uma especificação opcional que indica que o Amazon Redshift deve ignorar valores nulos na determinação de qual linha usar. Valores nulos são incluídos se IGNORE NULLS não for listada.

> **Note**
>
> Você pode usar uma expressão NVL ou COALESCE para substituir os valores nulos por outro valor. Para obter mais informações, consulte Funções NVL e COALESCE.

RESPECT NULLS

Indica que o Amazon Redshift deve incluir valores nulos na determinação de qual linha usar. RESPECT NULLS é compatível por padrão se você não especificar IGNORE NULLS.

OVER

Especifica o particionamento e ordem da janela. A cláusula OVER não pode conter uma especificação de quadro da janela.

PARTITION BY window_partition

Um argumento ideal que define o intervalo de registros para cada grupo na cláusula OVER.

ORDER BY window_ordering

Classifica as linhas dentro de cada partição.

A função da janela LAG é compatível com expressões que usam qualquer um dos tipos de dados dos Amazon Redshift. O tipo de retorno é igual ao tipo de value_expr.

Exemplos

O seguinte exemplo mostra a quantidade de ingressos vendidos para o comprador com ID de comprador 3 e a hora que o comprador 3 adquiriu os ingressos. Para comparar cada venda com venda anterior para o comprador 3, a consulta retorna a quantidade anterior vendida em cada venda. Já que não há compras antes de 1/16/2008, o primeiro valor de quantidade vendida anteriormente é nulo:

```
select buyerid, saletime, qtysold,
lag(qtysold,1) over (order by buyerid, saletime) as prev_qtysold
from sales where buyerid = 3 order by buyerid, saletime;

buyerid |      saletime       | qtysold | prev_qtysold
---------+---------------------+---------+--------------
3 | 2008-01-16 01:06:09 |       1 |
3 | 2008-01-28 02:10:01 |       1 |            1
3 | 2008-03-12 10:39:53 |       1 |            1
3 | 2008-03-13 02:56:07 |       1 |            1
3 | 2008-03-29 08:21:39 |       2 |            1
3 | 2008-04-27 02:39:01 |       1 |            2
3 | 2008-08-16 07:04:37 |       2 |            1
```

```
3 | 2008-08-22 11:45:26 |       2 |            2
3 | 2008-09-12 09:11:25 |       1 |            2
3 | 2008-10-01 06:22:37 |       1 |            1
3 | 2008-10-20 01:55:51 |       2 |            1
3 | 2008-10-28 01:30:40 |       1 |            2
(12 rows)
```

Função de janela LAST_VALUE

Considerando um conjunto de linhas ordenadas, a função LAST_VALUE retorna o valor da expressão em relação à última linha no quadro.

Para obter informações sobre como selecionar a primeira linha no quadro, consulte Função de janela FIRST_VALUE.

Sintaxe

```
LAST_VALUE( expression )[ IGNORE NULLS | RESPECT NULLS ]
OVER (
[ PARTITION BY expr_list ]
[ ORDER BY order_list frame_clause ]
)
```

Argumentos

expressão

A coluna ou expressão de destino na qual a função opera.

IGNORE NULLS

A função retorna o último valor no quadro que não seja NULL (ou NULL se todos os valores forem NULL).

RESPECT NULLS

Indica que o Amazon Redshift deve incluir valores nulos na determinação de qual linha usar. RESPECT NULLS é compatível por padrão se você não especificar IGNORE NULLS.

OVER

Introduz as cláusulas de janela para a função.

PARTITION BY expr_list

Define a janela para a função em termos de uma ou mais expressões.

ORDER BY order_list

Classifica as linhas dentro de cada partição. Se nenhuma cláusula PARTITION BY for especificada, ORDER BY classifica a tabela inteira. Se você especificar uma cláusula ORDER BY, você também deve especificar uma frame_clause.

Os resultados dependem da ordem dos dados. Os resultados são não determinísticos nos seguintes casos:

- Quando uma cláusula ORDER BY é especificada e uma partição contém dois valores diferentes para uma expressão
- Quando uma expressão avalia para valores diferentes que correspondem ao mesmo valor na lista ORDER BY.

frame_clause

Se uma cláusula ORDER BY é usada para uma função agregada, uma cláusula de quadro explícita é necessária. A cláusula de quadro refina o conjunto de linhas na janela de uma função, incluindo ou excluindo conjuntos de linhas no resultado ordenado. A cláusula de quadro consiste na palavra-chave ROWS e nos especificadores associados. Consulte Resumo da sintaxe de funções da janela.

Tipo de retorno

Essas funções são compatíveis com expressões que usam tipos de dados primitivos do Amazon Redshift. O tipo de retorno é igual ao tipo de dados da expressão.

Exemplos

Os exemplos a seguir usam a tabela VENUE dos dados de amostra TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

O seguinte exemplo retorna a capacidade de acomodação para cada local de evento da tabela VENUE com os resultados ordenados por capacidade (alta a baixa). A função LAST_VALUE é usada para selecionar o local de evento que corresponde à última linha no quadro: nesse caso, a linha com o mais baixo número de assentos. Os resultados são particionados por estado, portanto quando o

valor VENUESTATE muda, um novo último valor é selecionado. O quadro da janela não é vinculado, portanto o mesmo último valor é selecionado para cada linha em cada partição.

Para a Califórnia, `Shoreline Amphitheatre` será retornado para cada linha na partição, pois tem o menor número de assentos (`22000`).

```
select venuestate, venueseats, venuename,
last_value(venuename)
over(partition by venuestate
order by venueseats desc
rows between unbounded preceding and unbounded following)
from (select * from venue where venueseats >0)
order by venuestate;

venuestate | venueseats |           venuename            |          last_value
-----------+------------+--------------------------------
+-----------------------------
CA         |      70561 | Qualcomm Stadium               | Shoreline Amphitheatre
CA         |      69843 | Monster Park                   | Shoreline Amphitheatre
CA         |      63026 | McAfee Coliseum                | Shoreline Amphitheatre
CA         |      56000 | Dodger Stadium                 | Shoreline Amphitheatre
CA         |      45050 | Angel Stadium of Anaheim       | Shoreline Amphitheatre
CA         |      42445 | PETCO Park                     | Shoreline Amphitheatre
CA         |      41503 | AT&T Park                      | Shoreline Amphitheatre
CA         |      22000 | Shoreline Amphitheatre         | Shoreline Amphitheatre
CO         |      76125 | INVESCO Field                  | Coors Field
CO         |      50445 | Coors Field                    | Coors Field
DC         |      41888 | Nationals Park                 | Nationals Park
FL         |      74916 | Dolphin Stadium                | Tropicana Field
FL         |      73800 | Jacksonville Municipal Stadium | Tropicana Field
FL         |      65647 | Raymond James Stadium          | Tropicana Field
FL         |      36048 | Tropicana Field                | Tropicana Field
...
```

Função de janela LEAD

A função de janela LEAD retorna os valores para uma linha em determinado deslocamento abaixo (depois) da linha atual na partição.

Sintaxe

```
LEAD (value_expr [, offset ])
```

```
[ IGNORE NULLS | RESPECT NULLS ]
OVER ( [ PARTITION BY window_partition ] ORDER BY window_ordering )
```

Argumentos

value_expr

A coluna ou expressão de destino na qual a função opera.

deslocamento

Um parâmetro opcional que especifica o número de linhas abaixo da linha atual para as quais retornar valores. Este deslocamento pode ser um inteiro constante ou uma expressão que avalia para um inteiro. Se você não especificar um deslocamento, o Amazon Redshift usará 1 como o valor padrão. Um deslocamento de `0` indica a linha atual.

IGNORE NULLS

Uma especificação opcional que indica que o Amazon Redshift deve ignorar valores nulos na determinação de qual linha usar. Valores nulos são incluídos se IGNORE NULLS não for listada.

> **Note**
>
> Você pode usar uma expressão NVL ou COALESCE para substituir os valores nulos por outro valor. Para obter mais informações, consulte Funções NVL e COALESCE.

RESPECT NULLS

Indica que o Amazon Redshift deve incluir valores nulos na determinação de qual linha usar. RESPECT NULLS é compatível por padrão se você não especificar IGNORE NULLS.

OVER

Especifica o particionamento e ordem da janela. A cláusula OVER não pode conter uma especificação de quadro da janela.

PARTITION BY window_partition

Um argumento ideal que define o intervalo de registros para cada grupo na cláusula OVER.

ORDER BY window_ordering

Classifica as linhas dentro de cada partição.

A função da janela LEAD é compatível com expressões que usam qualquer um dos tipos de dados do Amazon Redshift. O tipo de retorno é igual ao tipo de value_expr.

Exemplos

O seguinte exemplo fornece a comissão para eventos na tabela SALES para os quais ingressos foram vendidos em 1º de janeiro de 2008 e 2 de janeiro de 2008, assim como a comissão paga por vendas de ingressos para a venda subsequente. Os exemplos a seguir usa o banco de dados de exemplo de TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

```
SELECT eventid, commission, saletime, LEAD(commission, 1) over ( ORDER BY saletime ) AS
 next_comm
FROM sales
WHERE saletime BETWEEN '2008-01-09 00:00:00' AND '2008-01-10 12:59:59'
LIMIT 10;

+---------+------------+---------------------+-----------+
| eventid | commission |      saletime       | next_comm |
+---------+------------+---------------------+-----------+
|    1664 |       13.2 | 2008-01-09 01:00:21 |      69.6 |
|     184 |       69.6 | 2008-01-09 01:00:36 |     116.1 |
|    6870 |      116.1 | 2008-01-09 01:02:37 |      11.1 |
|    3718 |       11.1 | 2008-01-09 01:05:19 |     205.5 |
|    6772 |      205.5 | 2008-01-09 01:14:04 |      38.4 |
|    3074 |       38.4 | 2008-01-09 01:26:50 |     209.4 |
|    5254 |      209.4 | 2008-01-09 01:29:16 |      26.4 |
|    3724 |       26.4 | 2008-01-09 01:40:09 |      57.6 |
|    5303 |       57.6 | 2008-01-09 01:40:21 |      51.6 |
|    3678 |       51.6 | 2008-01-09 01:42:54 |      43.8 |
+---------+------------+---------------------+-----------+
```

O exemplo a seguir fornece a diferença máxima entre a comissão de eventos na tabela SALES e a comissão paga pela venda de ingressos para a venda subsequente do mesmo evento. O exemplo a seguir mostra como usar LEAD com uma cláusula GROUP BY. Como as funções de janela não são permitidas em cláusulas agregadas, este exemplo usa uma subconsulta. Os exemplos a seguir usa o banco de dados de exemplo de TICKIT. Para obter mais informações, consulte Banco de dados de exemplo.

```
SELECT eventid, eventname, max(next_comm_diff) as max_commission_difference
FROM
(
```

```
    SELECT sales.eventid, eventname, commission - LEAD(commission, 1) over (ORDER BY
 sales.eventid, saletime) AS next_comm_diff
    FROM sales JOIN event ON sales.eventid = event.eventid
)
GROUP BY eventid, eventname
ORDER BY eventid

LIMIT 10

| eventid | eventname                  | max_commission_difference |
+---------+----------------------------+---------------------------+
| 1       | Gotterdammerung            | 7.95                      |
| 2       | Boris Godunov              | 227.85                    |
| 3       | Salome                     | 1350.9                    |
| 4       | La Cenerentola (Cinderella) | 790.05                    |
| 5       | Il Trovatore               | 214.05                    |
| 6       | L Elisir d Amore           | 510.9                     |
| 7       | Doctor Atomic              | 180.6                     |
| 9       | The Fly                    | 147                       |
| 10      | Rigoletto                  | 186.6                     |
+---------+----------------------------+---------------------------+
```

Função de janela LISTAGG

Para cada grupo em uma consulta, a função de janela LISTAGG ordena as linhas desse grupo de acordo com a expressão ORDER BY e, depois, concatena os valores em uma única string.

Sintaxe

```
LISTAGG( [DISTINCT] expression [, 'delimiter' ] )
[ WITHIN GROUP (ORDER BY order_list) ]
OVER ( [PARTITION BY partition_expression] )
```

Argumentos

DISTINCT

(Opcional) Uma cláusula que elimina valores duplicados da expressão especificada antes de concatenar. Os espaços à esquerda são ignorados, portanto, as strings `'a'` e `'a '` são tratadas como duplicatas. LISTAGG usa o primeiro valor encontrado. Para obter mais informações, consulte Significância de espaços em branco.

aggregate_expression

Qualquer expressão válida (tal como um nome de coluna) que forneça os valores para agregar. Valores NULL e strings vazias são ignoradas.

delimitador

(Opcional) A constante de string que separará os valores concatenados. O padrão é NULL.

WITHIN GROUP (ORDER BY order_list)

(Opcional) Uma cláusula que especifica a ordem de classificação dos valores agregados. Determinística somente se ORDER BY fornecer uma ordem exclusiva. O padrão é agregar todas as linhas e retornar um único valor.

OVER

Uma cláusula que especifica o particionamento da janela. A cláusula OVER não pode conter uma especificação de ordenação de janela ou de quadro de janela.

PARTITION BY partition_expression

(Opcional) Define o intervalo de registros de cada grupo na cláusula OVER.

Retornos

Se o conjunto de resultados for superior a 16 milhões de bytes, LISTAGG exibirá o seguinte erro:

```
Invalid operation: Result size exceeds LISTAGG limit
```

Exemplos

Os seguintes exemplos usam a tabela WINSALES. Para uma descrição da tabela WINSALES, consulte Amostra de tabela para exemplos de funções de janela.

O seguinte exemplo retorna uma lista de IDs de vendedor, ordenados por ID de vendedor.

```
select listagg(sellerid)
within group (order by sellerid)
over() from winsales;

  listagg
------------
 11122333344
...
```

```
...
 11122333344
 11122333344
   (11 rows)
```

O seguinte exemplo retorna uma lista de IDs de vendedor para o comprador B, ordenados por data.

```
select listagg(sellerid)
within group (order by dateid)
over () as seller
from winsales
where buyerid = 'b' ;

  seller
---------
    3233
    3233
    3233
    3233
```

O seguinte exemplo retorna uma lista separada por vírgula das datas de vendas para o comprador B.

```
select listagg(dateid,',')
within group (order by sellerid desc,salesid asc)
over () as dates
from winsales
where buyerid  = 'b';

             dates
-----------------------------------------
2003-08-02,2004-04-18,2004-04-18,2004-02-12
2003-08-02,2004-04-18,2004-04-18,2004-02-12
2003-08-02,2004-04-18,2004-04-18,2004-02-12
2003-08-02,2004-04-18,2004-04-18,2004-02-12
```

O exemplo a seguir usa DISTINCT para retornar uma lista de datas de vendas exclusivas para o comprador B.

```
select listagg(distinct dateid,',')
within group (order by sellerid desc,salesid asc)
over () as dates
from winsales
```

```
where buyerid  = 'b';

           dates
-------------------------------
2003-08-02,2004-04-18,2004-02-12
2003-08-02,2004-04-18,2004-02-12
2003-08-02,2004-04-18,2004-02-12
2003-08-02,2004-04-18,2004-02-12
```

O seguinte exemplo retorna uma lista separada por vírgulas dos IDs de venda para cada ID de comprador.

```
select buyerid,
listagg(salesid,',')
within group (order by salesid)
over (partition by buyerid) as sales_id
from winsales
order by buyerid;

+---------+------------------------+
| buyerid |        sales_id        |
+---------+------------------------+
| a       | 10005,40001,40005      |
| a       | 10005,40001,40005      |
| a       | 10005,40001,40005      |
| b       | 20001,30001,30003,30004 |
| b       | 20001,30001,30003,30004 |
| b       | 20001,30001,30003,30004 |
| b       | 20001,30001,30003,30004 |
| c       | 10001,10006,20002,30007 |
| c       | 10001,10006,20002,30007 |
| c       | 10001,10006,20002,30007 |
| c       | 10001,10006,20002,30007 |
+---------+------------------------+
```

O exemplo a seguir demonstra o suporte ao LISTAGG com resultados concatenados de até 16 milhões de bytes:

```
CREATE TABLE large_data (
    id INT,
    content VARCHAR(65535)
);
```

```
INSERT INTO large_data VALUES
    (1, REPEAT('A', 65535)),
    (2, REPEAT('B', 65535)),
    (3, REPEAT('C', 65535));

SELECT LEN(LISTAGG(content, ',') WITHIN GROUP (ORDER BY id)) AS total_length
FROM large_data;

 total_length
--------------
       196607
```

Função de janela MAX

A função MAX de janela retorna o máximo dos valores de entrada da expressão. A função MAX funciona com valores numéricos e ignora valores NULL.

Sintaxe

```
MAX ( [ ALL ] expression ) OVER
(
[ PARTITION BY expr_list ]
[ ORDER BY order_list frame_clause ]
)
```

Argumentos

expressão

A coluna ou expressão de destino na qual a função opera.

ALL

Com o argumento ALL, a função retém todos os valores duplicados da expressão. ALL é o padrão. DISTINCT não é compatível.

OVER

A cláusula especifica as cláusulas de janela para as funções de agregação. A cláusula OVER distingue funções de agregação de janela das funções de agregação de conjuntos normais.

PARTITION BY expr_list

Define a janela para a função MAX em termos de uma ou mais expressões.

ORDER BY order_list

Classifica as linhas dentro de cada partição. Se nenhuma PARTITION BY for especificada, ORDER BY usa a tabela completa.

frame_clause

Se uma cláusula ORDER BY é usada para uma função agregada, uma cláusula de quadro explícita é necessária. A cláusula de quadro refina o conjunto de linhas na janela de uma função, incluindo ou excluindo conjuntos de linhas no resultado ordenado. A cláusula de quadro consiste na palavra-chave ROWS e nos especificadores associados. Consulte Resumo da sintaxe de funções da janela.

Tipos de dados

Aceita qualquer tipo de dados como entrada. Retorna o mesmo tipo de dados da expressão.

Exemplos

O seguinte exemplo mostra o ID de vendas, a quantidade e a quantidade máxima desde o início da janela de dados:

```
select salesid, qty,
max(qty) over (order by salesid rows unbounded preceding) as max
from winsales
order by salesid;

salesid | qty | max
---------+-----+-----
10001 |  10 |  10
10005 |  30 |  30
10006 |  10 |  30
20001 |  20 |  30
20002 |  20 |  30
30001 |  10 |  30
30003 |  15 |  30
30004 |  20 |  30
30007 |  30 |  30
40001 |  40 |  40
40005 |  10 |  40
(11 rows)
```

Para uma descrição da tabela WINSALES, consulte Amostra de tabela para exemplos de funções de janela.

O seguinte exemplo mostra o salesid, a quantidade e a quantidade máxima em um quadro restrito:

```
select salesid, qty,
max(qty) over (order by salesid rows between 2 preceding and 1 preceding) as max
from winsales
order by salesid;

salesid | qty | max
---------+-----+-----
10001 |  10 |
10005 |  30 |  10
10006 |  10 |  30
20001 |  20 |  30
20002 |  20 |  20
30001 |  10 |  20
30003 |  15 |  20
30004 |  20 |  15
30007 |  30 |  20
40001 |  40 |  30
40005 |  10 |  40
(11 rows)
```

Função de janela MEDIAN

Calcula o valor mediano para o intervalo valores em uma janela ou partição. Valores NULL no intervalo são ignorados.

MEDIAN é uma função de distribuição inversa que assume um modelo de distribuição contínua.

Sintaxe

```
MEDIAN ( median_expression )
OVER ( [ PARTITION BY partition_expression ] )
```

Argumentos

median_expression

Uma expressão, tal como um nome de coluna, que fornece os valores para os quais determinar a mediana. A expressão deve ter um tipo de dados numérico ou de datetime ou ser implicitamente conversível para um.

OVER

Uma cláusula que especifica o particionamento da janela. A cláusula OVER não pode conter uma especificação de ordenação de janela ou de quadro de janela.

PARTITION BY partition_expression

Opcional. Uma expressão que define o intervalo de registros para cada grupo na cláusula OVER.

Tipos de dados

O tipo de retorno é determinado pelo tipo de dados de median_expression. A tabela a seguir mostra o tipo de retorno para cada tipo de dados de median_expression.

Tipo de entrada	Tipo de retorno
INT2, INT4, INT8, NUMERIC, DECIMAL	DECIMAL
FLOAT, DOUBLE	DOUBLE
DATE	DATE

Observações de uso

Se o argumento de median_expression é um tipo de dados DECIMAL com a precisão máxima de 38 dígitos, é possível que MEDIAN retorne um resultado impreciso ou um erro. Se o valor de retorno da função MEDIAN excede 38 dígitos, o resultado é truncado, o que causa a perda de precisão. Se, durante a interpolação, um resultado intermediário excede a precisão máxima, um excedente numérico ocorre e função retorna um erro. Para evitar essas condições, recomendamos o uso de um tipo de dados com menor precisão ou a conversão do argumento median_expression para uma precisão mais baixa.

Por exemplo, uma função SUM com um argumento DECIMAL retorna uma precisão padrão de 38 dígitos. A escala do resultado é a mesma que a escala do argumento. Portanto, por exemplo, uma SUM de uma coluna DECIMAL(5,2) retorna um tipo de dados DECIMAL(38,2).

O seguinte exemplo usa uma função SUM no argumento median_expression de uma função MEDIAN. O tipo de dados de coluna PRICEPAID é DECIMAL (8,2), portanto a função SUM retorna DECIMAL(38,2).

```
select salesid, sum(pricepaid), median(sum(pricepaid))
over() from sales where salesid < 10 group by salesid;
```

Para evitar a perda potencial de precisão ou um erro de sobrecarga, converta o resultado para um tipo de dados DECIMAL com menor precisão, conforme exibido no exemplo a seguir.

```
select salesid, sum(pricepaid), median(sum(pricepaid)::decimal(30,2))
over() from sales where salesid < 10 group by salesid;
```

Exemplos

O seguinte exemplo calcula a quantidade mediana de vendas para cada vendedor:

```
select sellerid, qty, median(qty)
over (partition by sellerid)
from winsales
order by sellerid;


sellerid qty median
---------------------------
1  10 10.0
1  10 10.0
1  30 10.0
2  20 20.0
2  20 20.0
3  10 17.5
3  15 17.5
3  20 17.5
3  30 17.5
4  10 25.0
4  40 25.0
```

Para uma descrição da tabela WINSALES, consulte Amostra de tabela para exemplos de funções de janela.

Função de janela MIN

A função MIN de janela retorna o mínimo dos valores de entrada da expressão. A função MIN funciona com valores numéricos e ignora valores NULL.

Sintaxe

```
MIN ( [ ALL ] expression ) OVER
(
[ PARTITION BY expr_list ]
[ ORDER BY order_list frame_clause ]
)
```

Argumentos

expressão

A coluna ou expressão de destino na qual a função opera.

ALL

Com o argumento ALL, a função retém todos os valores duplicados da expressão. ALL é o padrão. DISTINCT não é compatível.

OVER

Especifica as cláusulas de janela das funções de agregação. A cláusula OVER distingue funções de agregação de janela das funções de agregação de conjuntos normais.

PARTITION BY expr_list

Define a janela para a função MIN em termos de uma ou mais expressões.

ORDER BY order_list

Classifica as linhas dentro de cada partição. Se nenhuma PARTITION BY for especificada, ORDER BY usa a tabela completa.

frame_clause

Se uma cláusula ORDER BY é usada para uma função agregada, uma cláusula de quadro explícita é necessária. A cláusula de quadro refina o conjunto de linhas na janela de uma função, incluindo ou excluindo conjuntos de linhas no resultado ordenado. A cláusula de quadro consiste

na palavra-chave ROWS e nos especificadores associados. Consulte Resumo da sintaxe de funções da janela.

Tipos de dados

Aceita qualquer tipo de dados como entrada. Retorna o mesmo tipo de dados da expressão.

Exemplos

O seguinte exemplo mostra o ID de vendas, a quantidade e a quantidade mínima desde o início da janela de dados:

```
select salesid, qty,
min(qty) over
(order by salesid rows unbounded preceding)
from winsales
order by salesid;

salesid | qty | min
---------+-----+-----
10001 |  10 |  10
10005 |  30 |  10
10006 |  10 |  10
20001 |  20 |  10
20002 |  20 |  10
30001 |  10 |  10
30003 |  15 |  10
30004 |  20 |  10
30007 |  30 |  10
40001 |  40 |  10
40005 |  10 |  10
(11 rows)
```

Para uma descrição da tabela WINSALES, consulte Amostra de tabela para exemplos de funções de janela.

O seguinte exemplo mostra o ID de vendas, a quantidade e a quantidade mínima em um quadro restrito:

```
select salesid, qty,
min(qty) over
(order by salesid rows between 2 preceding and 1 preceding) as min
```

```
from winsales
order by salesid;

salesid | qty | min
---------+-----+-----
10001 |  10 |
10005 |  30 |  10
10006 |  10 |  10
20001 |  20 |  10
20002 |  20 |  10
30001 |  10 |  20
30003 |  15 |  10
30004 |  20 |  10
30007 |  30 |  15
40001 |  40 |  20
40005 |  10 |  30
(11 rows)
```

Função de janela NTH_VALUE

A função de janela NTH_VALUE retorna o valor de expressão da linha especificada do quadro da janela em relação à primeira linha da janela.

Sintaxe

```
NTH_VALUE (expr, offset)
[ IGNORE NULLS | RESPECT NULLS ]
OVER
( [ PARTITION BY window_partition ]
[ ORDER BY window_ordering
                        frame_clause ] )
```

Argumentos

expr

A coluna ou expressão de destino na qual a função opera.

deslocamento

Determina o número de linha relativo a primeira linha na janela para a qual retornar a expressão. O deslocamento pode ser uma constante ou uma expressão e deve ser um inteiro positivo que seja maior que 0.

IGNORE NULLS

Uma especificação opcional que indica que o Amazon Redshift deve ignorar valores nulos na determinação de qual linha usar. Valores nulos são incluídos se IGNORE NULLS não for listada.

RESPECT NULLS

Indica que o Amazon Redshift deve incluir valores nulos na determinação de qual linha usar. RESPECT NULLS é compatível por padrão se você não especificar IGNORE NULLS.

OVER

Especifica o particionamento da janela, ordem e quadro da janela.

PARTITION BY window_partition

Define o intervalo de registros para cada grupo na cláusula OVER.

ORDER BY window_ordering

Classifica as linhas dentro de cada partição. Se ORDER BY for omitido, o quadro padrão consiste em todas as linhas na partição.

frame_clause

Se uma cláusula ORDER BY é usada para uma função agregada, uma cláusula de quadro explícita é necessária. A cláusula de quadro refina o conjunto de linhas na janela de uma função, incluindo ou excluindo conjuntos de linhas no resultado ordenado. A cláusula de quadro consiste na palavra-chave ROWS e nos especificadores associados. Consulte Resumo da sintaxe de funções da janela.

A função da janela NTH_VALUE é compatível com expressões que usam qualquer um dos tipos de dados do Amazon Redshift. O tipo de retorno é igual ao tipo de expr.

Exemplos

O seguinte exemplo mostra o número de assentos no terceiro maior local de eventos na Califórnia, Flórida e Nova Iorque, comparados ao número de assentos em outros locais de evento nesses estados:

```
select venuestate, venuename, venueseats,
nth_value(venueseats, 3)
ignore nulls
over(partition by venuestate order by venueseats desc
rows between unbounded preceding and unbounded following)
```

```
as third_most_seats
from (select * from venue where venueseats > 0 and
venuestate in('CA', 'FL', 'NY'))
order by venuestate;

venuestate |           venuename            | venueseats | third_most_seats
-----------+--------------------------------+------------+------------------
CA         | Qualcomm Stadium               |      70561 |            63026
CA         | Monster Park                   |      69843 |            63026
CA         | McAfee Coliseum                |      63026 |            63026
CA         | Dodger Stadium                 |      56000 |            63026
CA         | Angel Stadium of Anaheim       |      45050 |            63026
CA         | PETCO Park                     |      42445 |            63026
CA         | AT&T Park                      |      41503 |            63026
CA         | Shoreline Amphitheatre         |      22000 |            63026
FL         | Dolphin Stadium                |      74916 |            65647
FL         | Jacksonville Municipal Stadium |      73800 |            65647
FL         | Raymond James Stadium          |      65647 |            65647
FL         | Tropicana Field                |      36048 |            65647
NY         | Ralph Wilson Stadium           |      73967 |            20000
NY         | Yankee Stadium                 |      52325 |            20000
NY         | Madison Square Garden          |      20000 |            20000
(15 rows)
```

Função de janela NTILE

A função de janela NTILE divide as linhas ordenadas na partição no número especificado de grupos classificados de tamanho o mais igual possível e retorna o grupo em que dada linha se encontra.

Sintaxe

```
NTILE (expr)
OVER (
[ PARTITION BY expression_list ]
[ ORDER BY order_list ]
)
```

Argumentos

expr

O número de grupos de classificação, devendo resultar em um valor inteiro positivo (maior que 0) para cada partição. O argumento da expr não deve ser anulável.

OVER

Uma cláusula que especifica o particionamento e ordenação da janela. A cláusula OVER não pode conter uma especificação de quadro da janela.

PARTITION BY window_partition

Opcional. O intervalo de registros para cada grupo na cláusula OVER.

ORDER BY window_ordering

Opcional. Uma expressão que classifica as linhas dentro de cada partição. Se a cláusula ORDER BY for omitida, o comportamento da classificação será o mesmo.

Se ORDER BY não produzir uma ordenação exclusiva, a ordem das linhas será não determinística. Para obter mais informações, consulte Ordenação exclusiva de dados para funções da janela.

Tipo de retorno

BIGINT

Exemplos

As seguintes classificações de exemplo classifica o preço pago por ingressos de Hamlet em 26 de agosto de 2008 em quatro grupos de classificação. O conjunto de resultados é 17 linhas, divididas quase uniformemente entre as classificações 1 a 4:

```
select eventname, caldate, pricepaid, ntile(4)
over(order by pricepaid desc) from sales, event, date
where sales.eventid=event.eventid and event.dateid=date.dateid and eventname='Hamlet'
and caldate='2008-08-26'
order by 4;

eventname |  caldate   | pricepaid | ntile
----------+------------+-----------+-------
Hamlet    | 2008-08-26 |   1883.00 |     1
Hamlet    | 2008-08-26 |   1065.00 |     1
Hamlet    | 2008-08-26 |    589.00 |     1
Hamlet    | 2008-08-26 |    530.00 |     1
Hamlet    | 2008-08-26 |    472.00 |     1
Hamlet    | 2008-08-26 |    460.00 |     2
```

```
Hamlet    | 2008-08-26 |    355.00 |     2
Hamlet    | 2008-08-26 |    334.00 |     2
Hamlet    | 2008-08-26 |    296.00 |     2
Hamlet    | 2008-08-26 |    230.00 |     3
Hamlet    | 2008-08-26 |    216.00 |     3
Hamlet    | 2008-08-26 |    212.00 |     3
Hamlet    | 2008-08-26 |    106.00 |     3
Hamlet    | 2008-08-26 |    100.00 |     4
Hamlet    | 2008-08-26 |     94.00 |     4
Hamlet    | 2008-08-26 |     53.00 |     4
Hamlet    | 2008-08-26 |     25.00 |     4
(17 rows)
```

Função de janela PERCENT_RANK

Calcula a classificação percentual de dada linha. A classificação percentual é determinada usando esta fórmula:

```
(x - 1) / (the number of rows in the window or partition - 1)
```

onde x é a classificação da linha atual. O seguinte conjunto de dados ilustra O uso desta fórmula:

```
Row# Value Rank Calculation PERCENT_RANK
1 15 1 (1-1)/(7-1) 0.0000
2 20 2 (2-1)/(7-1) 0.1666
3 20 2 (2-1)/(7-1) 0.1666
4 20 2 (2-1)/(7-1) 0.1666
5 30 5 (5-1)/(7-1) 0.6666
6 30 5 (5-1)/(7-1) 0.6666
7 40 7 (7-1)/(7-1) 1.0000
```

O intervalo de valor de retorno é 0 a 1, inclusive. A primeira linha em qualquer conjunto tem um PERCENT_RANK de 0.

Sintaxe

```
PERCENT_RANK ()
OVER (
[ PARTITION BY partition_expression ]
[ ORDER BY order_list ]
)
```

Argumentos

( )

A função não aceita argumentos, mas os parênteses vazios são necessários.

OVER

Uma cláusula que especifica o particionamento da janela. A cláusula OVER não pode conter uma especificação de quadro da janela.

PARTITION BY partition_expression

Opcional. Uma expressão que define o intervalo de registros para cada grupo na cláusula OVER.

ORDER BY order_list

Opcional. A expressão na qual calcular a classificação percentual. A expressão deve ter um tipo de dados numérico ou ser implicitamente conversível para um. Se ORDER BY for omitida, o valor de retorno será 0 para todas as linhas.

Se ORDER BY não produzir uma ordenação exclusiva, a ordem das linhas será não determinística. Para obter mais informações, consulte Ordenação exclusiva de dados para funções da janela.

Tipo de retorno

FLOAT8

Exemplos

O seguinte exemplo calcula a classificação percentual das quantidades de vendas para cada vendedor:

```
select sellerid, qty, percent_rank()
over (partition by sellerid order by qty)
from winsales;

sellerid qty  percent_rank
----------------------------------------
1  10.00  0.0
1  10.64  0.5
1  30.37  1.0
```

```
3  10.04  0.0
3  15.15  0.33
3  20.75  0.67
3  30.55  1.0
2  20.09  0.0
2  20.12  1.0
4  10.12  0.0
4  40.23  1.0
```

Para uma descrição da tabela WINSALES, consulte Amostra de tabela para exemplos de funções de janela.

Função de janela PERCENTILE_CONT

PERCENTILE_CONT é uma função de distribuição inversa que assume um modelo de distribuição contínua. Ela pega um valor percentil e uma especificação de classificação e retorna um valor intercalar que cairia dentro do valor percentil fornecido em relação à especificação de classificação.

PERCENTILE_CONT computa uma interpolação linear entre valores após ordená-los. Usando o valor percentil `(P)` e o número de linhas não nulas `(N)` no grupo de agregação, a função computa o número da linha após ordenar as linhas de acordo com a especificação de classificação. Esse número de linha `(RN)` é computado de acordo com a fórmula `RN = (1+ (P*(N-1))`. O resultado final da função agregada é computado por interpolação linear entre os valores das linhas nos números de linha `CRN = CEILING(RN)` e `FRN = FLOOR(RN)`.

O resultado final será o seguinte.

Se `(CRN = FRN = RN)`, o resultado é `(value of expression from row at RN)`

Caso contrário, o resultado é o seguinte:

`(CRN - RN) * (value of expression for row at FRN) + (RN - FRN) * (value of expression for row at CRN)`.

Você pode especificar somente a cláusula PARTITION na cláusula OVER. Se PARTITION é especificada, para cada linha, PERCENTILE_CONT retorna o valor que cairia no percentil especificado entre um conjunto de valores dentro de dada partição.

Sintaxe

```
PERCENTILE_CONT ( percentile )
```

```
WITHIN GROUP (ORDER BY expr)
OVER (  [ PARTITION BY expr_list ]  )
```

Argumentos

percentil

Constante numérica entre 0 e 1. Nulls são ignorados no cálculo.

WITHIN GROUP (ORDER BY expr)

Especifica valores numéricos ou de data/hora para classificação e computação do percentil.

OVER

Especifica o particionamento da janela. A cláusula OVER não pode conter uma especificação de ordenação de janela ou de quadro de janela.

PARTITION BY expr

Argumento opcional que define o intervalo de registros para cada grupo na cláusula OVER.

Retornos

O tipo de retorno é determinado pelo tipo de dados da expressão ORDER BY na cláusula WITHIN GROUP. A tabela a seguir mostra o tipo de retorno para cada tipo de dados da expressão ORDER BY.

Tipo de entrada	Tipo de retorno
INT2, INT4, INT8, NUMERIC, DECIMAL	DECIMAL
FLOAT, DOUBLE	DOUBLE
DATE	DATE
TIMESTAMP	TIMESTAMP

Observações de uso

Se a expressão ORDER BY é um tipo de dados DECIMAL com a precisão máxima de 38 dígitos, é possível que PERCENTILE_CONT retorne um resultado impreciso ou um erro. Se o valor de retorno

da função PERCENTILE_CONT excede 38 dígitos, o resultado é truncado, o que causa a perda de precisão. Se, durante a interpolação, um resultado intermediário excede a precisão máxima, um excedente numérico ocorre e função retorna um erro. Para evitar essas condições, recomendamos o uso de um tipo de dados com menor precisão ou a conversão da expressão ORDER BY para uma precisão mais baixa.

Por exemplo, uma função SUM com um argumento DECIMAL retorna uma precisão padrão de 38 dígitos. A escala do resultado é a mesma que a escala do argumento. Portanto, por exemplo, uma SUM de uma coluna DECIMAL(5,2) retorna um tipo de dados DECIMAL(38,2).

O seguinte exemplo usa uma função SUM na cláusula ORDER BY de uma função PERCENTILE_CONT. O tipo de dados de coluna PRICEPAID é DECIMAL (8,2), portanto a função SUM retorna DECIMAL(38,2).

```
select salesid, sum(pricepaid), percentile_cont(0.6)
within group (order by sum(pricepaid) desc) over()
from sales where salesid < 10 group by salesid;
```

Para evitar a perda potencial de precisão ou um erro de sobrecarga, converta o resultado para um tipo de dados DECIMAL com menor precisão, conforme exibido no exemplo a seguir.

```
select salesid, sum(pricepaid), percentile_cont(0.6)
within group (order by sum(pricepaid)::decimal(30,2) desc) over()
from sales where salesid < 10 group by salesid;
```

Exemplos

Os seguintes exemplos usam a tabela WINSALES. Para uma descrição da tabela WINSALES, consulte Amostra de tabela para exemplos de funções de janela.

```
select sellerid, qty, percentile_cont(0.5)
within group (order by qty)
over() as median from winsales;

 sellerid | qty | median
----------+-----+--------
        1 |  10 |   20.0
        1 |  10 |   20.0
        3 |  10 |   20.0
        4 |  10 |   20.0
```

```
        3 |  15 |   20.0
        2 |  20 |   20.0
        3 |  20 |   20.0
        2 |  20 |   20.0
        3 |  30 |   20.0
        1 |  30 |   20.0
        4 |  40 |   20.0
(11 rows)
```

```
select sellerid, qty, percentile_cont(0.5)
within group (order by qty)
over(partition by sellerid) as median from winsales;

 sellerid | qty | median
----------+-----+--------
        2 |  20 |   20.0
        2 |  20 |   20.0
        4 |  10 |   25.0
        4 |  40 |   25.0
        1 |  10 |   10.0
        1 |  10 |   10.0
        1 |  30 |   10.0
        3 |  10 |   17.5
        3 |  15 |   17.5
        3 |  20 |   17.5
        3 |  30 |   17.5
(11 rows)
```

O seguinte exemplo calcula o PERCENTILE_CONT e PERCENTILE_DISC das vendas de ingressos para vendedores no estado de Washington.

```
SELECT sellerid, state, sum(qtysold*pricepaid) sales,
percentile_cont(0.6) within group (order by sum(qtysold*pricepaid::decimal(14,2) )
 desc) over(),
percentile_disc(0.6) within group (order by sum(qtysold*pricepaid::decimal(14,2) )
 desc) over()
from sales s, users u
where s.sellerid = u.userid and state = 'WA' and sellerid < 1000
group by sellerid, state;

 sellerid | state |  sales  | percentile_cont | percentile_disc
----------+-------+---------+-----------------+-----------------
      127 | WA    | 6076.00 |         2044.20 |         1531.00
```

```
      787 | WA    | 6035.00 |         2044.20 |         1531.00
      381 | WA    | 5881.00 |         2044.20 |         1531.00
      777 | WA    | 2814.00 |         2044.20 |         1531.00
       33 | WA    | 1531.00 |         2044.20 |         1531.00
      800 | WA    | 1476.00 |         2044.20 |         1531.00
        1 | WA    | 1177.00 |         2044.20 |         1531.00
(7 rows)
```

Função de janela PERCENTILE_DISC

PERCENTILE_DISC é uma função de distribuição inversa que assume um modelo de distribuição discreta. Ela pega um valor percentil e uma especificação de classificação e retorna um elemento do conjunto fornecido.

Para determinado valor percentil P, PERCENTILE_DISC classifica os valores da expressão na cláusula ORDER BY e retorna o valor com o menor valor de distribuição cumulativa (em relação à mesma especificação de classificação) que for maior que ou igual a P.

Você pode especificar somente a cláusula PARTITION na cláusula OVER.

Sintaxe

```
PERCENTILE_DISC ( percentile )
WITHIN GROUP (ORDER BY expr)
OVER (  [ PARTITION BY expr_list ]  )
```

Argumentos

percentil

Constante numérica entre 0 e 1. Nulls são ignorados no cálculo.

WITHIN GROUP (ORDER BY expr)

Especifica valores numéricos ou de data/hora para classificação e computação do percentil.

OVER

Especifica o particionamento da janela. A cláusula OVER não pode conter uma especificação de ordenação de janela ou de quadro de janela.

PARTITION BY expr

Argumento opcional que define o intervalo de registros para cada grupo na cláusula OVER.

Retornos

O mesmo tipo de dados que a expressão ORDER BY na cláusula WITHIN GROUP.

Exemplos

Os exemplos a seguir usam a tabela WINSALES. Para uma descrição da tabela WINSALES, consulte Amostra de tabela para exemplos de funções de janela.

```
SELECT sellerid, qty, PERCENTILE_DISC(0.5)
WITHIN GROUP (ORDER BY qty)
OVER() AS MEDIAN FROM winsales;

+----------+-----+--------+
| sellerid | qty | median |
+----------+-----+--------+
| 3        | 10  | 20     |
| 1        | 10  | 20     |
| 1        | 10  | 20     |
| 4        | 10  | 20     |
| 3        | 15  | 20     |
| 2        | 20  | 20     |
| 2        | 20  | 20     |
| 3        | 20  | 20     |
| 1        | 30  | 20     |
| 3        | 30  | 20     |
| 4        | 40  | 20     |
+----------+-----+--------+

SELECT sellerid, qty, PERCENTILE_DISC(0.5)
WITHIN GROUP (ORDER BY qty)
OVER(PARTITION BY sellerid) AS MEDIAN FROM winsales;

+----------+-----+--------+
| sellerid | qty | median |
+----------+-----+--------+
| 4        | 10  | 10     |
| 4        | 40  | 10     |
| 3        | 10  | 15     |
| 3        | 15  | 15     |
| 3        | 20  | 15     |
| 3        | 30  | 15     |
| 2        | 20  | 20     |
| 2        | 20  | 20     |
```

```
| 1         | 10  | 10     |
| 1         | 10  | 10     |
| 1         | 30  | 10     |
+----------+-----+--------+
```

Para encontrar PERCENTILE_DISC(0.25) e PERCENTILE_DISC(0.75) da quantidade quando particionada pelo ID do vendedor, use os exemplos a seguir.

```
SELECT sellerid, qty, PERCENTILE_DISC(0.25)
WITHIN GROUP (ORDER BY qty)
OVER(PARTITION BY sellerid) AS quartile1 FROM winsales;

+----------+-----+-----------+
| sellerid | qty | quartile1 |
+----------+-----+-----------+
| 4        | 10  | 10        |
| 4        | 40  | 10        |
| 2        | 20  | 20        |
| 2        | 20  | 20        |
| 3        | 10  | 10        |
| 3        | 15  | 10        |
| 3        | 20  | 10        |
| 3        | 30  | 10        |
| 1        | 10  | 10        |
| 1        | 10  | 10        |
| 1        | 30  | 10        |
+----------+-----+-----------+

SELECT sellerid, qty, PERCENTILE_DISC(0.75)
WITHIN GROUP (ORDER BY qty)
OVER(PARTITION BY sellerid) AS quartile3 FROM winsales;

+----------+-----+-----------+
| sellerid | qty | quartile3 |
+----------+-----+-----------+
| 3        | 10  | 20        |
| 3        | 15  | 20        |
| 3        | 20  | 20        |
| 3        | 30  | 20        |
| 4        | 10  | 40        |
| 4        | 40  | 40        |
| 2        | 20  | 20        |
| 2        | 20  | 20        |
```

```
| 1         | 10  | 30         |
| 1         | 10  | 30         |
| 1         | 30  | 30         |
+----------+-----+-----------+
```

Função de janela RANK

A função de janela RANK determina a classificação de um valor em um grupo de valores com base na expressão ORDER BY da cláusula OVER. Se a cláusula opcional PARTITION BY estiver presente, as classificações são redefinidas para cada grupo de linhas. Linhas com valores iguais para os critérios de classificação recebem a mesma classificação. O Amazon Redshift adiciona o número de linhas amarradas à classificação amarrada para calcular a próxima classificação e, assim, as classificações podem não ser números consecutivos. Por exemplo, se duas linhas são classificadas como 1, a classificação seguinte é 3.

RANK difere de Função de janela DENSE_RANK em um aspecto: para DENSE_RANK, se duas ou mais linhas empatarem, não há uma lacuna na sequência de valores classificados. Por exemplo, se duas linhas são classificadas como 1, a classificação seguinte é 2.

Você pode ter funções de classificação com diferentes cláusulas PARTITION BY e ORDER BY na mesma consulta.

Sintaxe

```
RANK () OVER
(
[ PARTITION BY expr_list ]
[ ORDER BY order_list ]
)
```

Argumentos

( )

A função não aceita argumentos, mas os parênteses vazios são necessários.

OVER

As cláusulas de janela para a função RANK.

PARTITION BY expr_list

Opcional. Uma ou várias expressões que definem a janela.

ORDER BY order_list

Opcional. Define as colunas nas quais os valores de classificação se baseiam. Se nenhuma PARTITION BY for especificada, ORDER BY usa a tabela completa. Se ORDER BY for omitida, o valor de retorno será 1 para todas as linhas.

Se ORDER BY não produzir uma ordenação exclusiva, a ordem das linhas será não determinística. Para obter mais informações, consulte Ordenação exclusiva de dados para funções da janela.

Tipo de retorno

INTEGER

Exemplos

O exemplo a seguir ordena a tabela pela quantidade vendida (padrão crescente) e atribui uma classificação a cada linha. O valor de classificação de 1 é o valor de classificação mais alto. Os resultados são classificados após a aplicação dos resultados da função de janela:

```
select salesid, qty,
rank() over (order by qty) as rnk
from winsales
order by 2,1;

salesid | qty | rnk
--------+-----+-----
10001 |  10 |  1
10006 |  10 |  1
30001 |  10 |  1
40005 |  10 |  1
30003 |  15 |  5
20001 |  20 |  6
20002 |  20 |  6
30004 |  20 |  6
10005 |  30 |  9
30007 |  30 |  9
40001 |  40 |  11
(11 rows)
```

Observe que a cláusula ORDER BY externa neste exemplo inclui as colunas 2 e 1 para garantir que o Amazon Redshift retorne resultados classificados de forma consistente sempre que esta consulta

for executada. Por exemplo, as linhas com IDs de venda 10.001 e 10.006 têm valores idênticos de QTY e RNK. Ordenar o conjunto de resultados final pela coluna 1 garante que a linha 10.001 sempre caia antes de 10.006. Para uma descrição da tabela WINSALES, consulte Amostra de tabela para exemplos de funções de janela.

No exemplo a seguir, a ordenação é revertida para a função da janela (`order by qty desc`). Agora, o valor de classificação mais alto se aplica ao valor de QTY mais alto.

```
select salesid, qty,
rank() over (order by qty desc) as rank
from winsales
order by 2,1;

 salesid | qty | rank
---------+-----+-----
   10001 |  10 |   8
   10006 |  10 |   8
   30001 |  10 |   8
   40005 |  10 |   8
   30003 |  15 |   7
   20001 |  20 |   4
   20002 |  20 |   4
   30004 |  20 |   4
   10005 |  30 |   2
   30007 |  30 |   2
   40001 |  40 |   1
(11 rows)
```

Para uma descrição da tabela WINSALES, consulte Amostra de tabela para exemplos de funções de janela.

O exemplo a seguir particiona a tabela por SELLERID e ordena cada partição pela quantidade (em ordem decrescente) e atribui uma classificação a cada linha. Os resultados são classificados após a aplicação dos resultados da função de janela.

```
select salesid, sellerid, qty, rank() over
(partition by sellerid
order by qty desc) as rank
from winsales
order by 2,3,1;

salesid | sellerid | qty | rank
```

```
--------+----------+-----+-----
  10001 |        1 |  10 |  2
  10006 |        1 |  10 |  2
  10005 |        1 |  30 |  1
  20001 |        2 |  20 |  1
  20002 |        2 |  20 |  1
  30001 |        3 |  10 |  4
  30003 |        3 |  15 |  3
  30004 |        3 |  20 |  2
  30007 |        3 |  30 |  1
  40005 |        4 |  10 |  2
  40001 |        4 |  40 |  1
(11 rows)
```

Função de janela RATIO_TO_REPORT

Calcula a proporção de um valor para a soma dos valores em uma janela ou partição. O valor da proporção de relatório é determinado usando a fórmula:

`value of` ratio_expression `argument for the current row / sum of` ratio_expression `argument for the window or partition`

O seguinte conjunto de dados ilustra O uso desta fórmula:

```
Row# Value Calculation RATIO_TO_REPORT
1 2500 (2500)/(13900) 0.1798
2 2600 (2600)/(13900) 0.1870
3 2800 (2800)/(13900) 0.2014
4 2900 (2900)/(13900) 0.2086
5 3100 (3100)/(13900) 0.2230
```

O intervalo de valor de retorno é 0 a 1, inclusive. Se ratio_expression for NULL, o valor de retorno será NULL. Se um valor em partition_expression for exclusivo, a função retornará 1 para esse valor.

Sintaxe

```
RATIO_TO_REPORT ( ratio_expression )
OVER ( [ PARTITION BY partition_expression ] )
```

Argumentos

ratio_expression

Uma expressão, tal como um nome de coluna, que fornece o valor para o qual determinar a proporção. A expressão deve ter um tipo de dados numérico ou ser implicitamente conversível para um.

Você não pode usar qualquer outra função analítica em ratio_expression.

OVER

Uma cláusula que especifica o particionamento da janela. A cláusula OVER não pode conter uma especificação de ordenação de janela ou de quadro de janela.

PARTITION BY partition_expression

Opcional. Uma expressão que define o intervalo de registros para cada grupo na cláusula OVER.

Tipo de retorno

FLOAT8

Exemplos

Os exemplos a seguir usam a tabela WINSALES. Para obter informações sobre como criar a tabela WINSALES, consulte Amostra de tabela para exemplos de funções de janela.

O exemplo a seguir calcula o valor da proporção em relação ao relatório de cada linha da quantidade de um vendedor em relação ao total de todas as quantidades do vendedor.

```
select sellerid, qty, ratio_to_report(qty)
over()
from winsales
order by sellerid;

sellerid  qty    ratio_to_report
-------------------------------------
1         30     0.13953488372093023
1         10     0.046511627906976744
1         10     0.046511627906976744
2         20     0.09302325581395349
2         20     0.09302325581395349
```

```
3          30     0.13953488372093023
3          20     0.09302325581395349
3          15     0.06976744186046512
3          10     0.046511627906976744
4          10     0.046511627906976744
4          40     0.18604651162790697
```

O seguinte exemplo calcula as proporções das quantidades de vendas para cada vendedor por partição:

```
select sellerid, qty, ratio_to_report(qty)
over(partition by sellerid)
from winsales;

sellerid   qty    ratio_to_report
-----------------------------------------
2          20     0.5
2          20     0.5
4          40     0.8
4          10     0.2
1          10     0.2
1          30     0.6
1          10     0.2
3          10     0.13333333333333333
3          15     0.2
3          20     0.26666666666666666
3          30     0.4
```

Função de janela ROW_NUMBER

Atribui um número ordinal da linha atual em um grupo de linhas, contando a partir de 1, com base na expressão ORDER BY da cláusula OVER. Se a cláusula opcional PARTITION BY estiver presente, os números ordinais são redefinidos para cada grupo de linhas. As linhas com valores iguais para as expressões ORDER BY recebem os diferentes números de linha de forma não determinística.

Sintaxe

```
ROW_NUMBER() OVER(
  [ PARTITION BY expr_list ]
  [ ORDER BY order_list ]
)
```

Argumentos

( )

A função não aceita argumentos, mas os parênteses vazios são necessários.

OVER

A cláusula de função de janela para a função ROW_NUMBER.

PARTITION BY expr_list

Opcional. Uma ou mais expressões de coluna que dividem os resultados em conjuntos de linhas.

ORDER BY order_list

Opcional. Uma ou mais expressões de coluna que definem a ordem das linhas em um conjunto. Se nenhuma PARTITION BY for especificada, ORDER BY usa a tabela completa.

Se ORDER BY não produzir uma ordenação exclusiva ou for omitido, a ordem das linhas será não determinística. Para obter mais informações, consulte Ordenação exclusiva de dados para funções da janela.

Tipo de retorno

BIGINT

Exemplos

Os exemplos a seguir usam a tabela `WINSALES`. Para uma descrição da tabela `WINSALES`, consulte Amostra de tabela para exemplos de funções de janela.

O exemplo a seguir ordena a tabela por QTY (em ordem ascendente) e, então, atribui um número a cada linha. Os resultados são classificados após a aplicação dos resultados da função de janela.

```
SELECT salesid, sellerid, qty,
ROW_NUMBER() OVER(
   ORDER BY qty ASC) AS row
FROM winsales
ORDER BY 4,1;

salesid   sellerid   qty   row
---------+----------+-----+-----
```

```
   30001 |        3 |  10 |   1
   10001 |        1 |  10 |   2
   10006 |        1 |  10 |   3
   40005 |        4 |  10 |   4
   30003 |        3 |  15 |   5
   20001 |        2 |  20 |   6
   20002 |        2 |  20 |   7
   30004 |        3 |  20 |   8
   10005 |        1 |  30 |   9
   30007 |        3 |  30 |  10
   40001 |        4 |  40 |  11
```

O seguinte exemplo particiona a tabela por SELLERID e ordena cada partição por QTY (na ordem ascendente) e, então, atribui um número de linha para cada linha. Os resultados são classificados após a aplicação dos resultados da função de janela.

```
SELECT salesid, sellerid, qty,
ROW_NUMBER() OVER(
  PARTITION BY sellerid
  ORDER BY qty ASC) AS row_by_seller
FROM winsales
ORDER BY 2,4;

 salesid | sellerid | qty | row_by_seller
---------+----------+-----+-----
   10001 |        1 |  10 |   1
   10006 |        1 |  10 |   2
   10005 |        1 |  30 |   3
   20001 |        2 |  20 |   1
   20002 |        2 |  20 |   2
   30001 |        3 |  10 |   1
   30003 |        3 |  15 |   2
   30004 |        3 |  20 |   3
   30007 |        3 |  30 |   4
   40005 |        4 |  10 |   1
   40001 |        4 |  40 |   2
```

O exemplo a seguir mostra os resultados quando as cláusulas opcionais não estão sendo usadas.

```
SELECT salesid, sellerid, qty, ROW_NUMBER() OVER() AS row
FROM winsales
ORDER BY 4,1;
```

```
salesid   sellerid   qty   row
---------+----------+-----+-----
   30001 |        3 |  10 |   1
   10001 |        1 |  10 |   2
   10005 |        1 |  30 |   3
   40001 |        4 |  40 |   4
   10006 |        1 |  10 |   5
   20001 |        2 |  20 |   6
   40005 |        4 |  10 |   7
   20002 |        2 |  20 |   8
   30003 |        3 |  15 |   9
   30004 |        3 |  20 |  10
   30007 |        3 |  30 |  11
```

Funções de janela STDDEV_SAMP e STDDEV_POP

As funções da janela STDDEV_SAMP e STDDEV_POP retornam o desvio padrão da amostra e da população de um conjunto de valores numéricos (número inteiro, decimal ou ponto flutuante). Consulte também Funções STDDEV_SAMP e STDDEV_POP.

STDDEV_SAMP e STDDEV são sinônimos para a mesma função.

Sintaxe

```
STDDEV_SAMP | STDDEV | STDDEV_POP
( [ ALL ] expression ) OVER
(
[ PARTITION BY expr_list ]
[ ORDER BY order_list
                        frame_clause ]
)
```

Argumentos

expressão

A coluna ou expressão de destino na qual a função opera.

ALL

Com o argumento ALL, a função retém todos os valores duplicados da expressão. ALL é o padrão. DISTINCT não é compatível.

OVER

Especifica as cláusulas de janela das funções de agregação. A cláusula OVER distingue funções de agregação de janela das funções de agregação de conjuntos normais.

PARTITION BY expr_list

Define a janela para a função em termos de uma ou mais expressões.

ORDER BY order_list

Classifica as linhas dentro de cada partição. Se nenhuma PARTITION BY for especificada, ORDER BY usa a tabela completa.

frame_clause

Se uma cláusula ORDER BY é usada para uma função agregada, uma cláusula de quadro explícita é necessária. A cláusula de quadro refina o conjunto de linhas na janela de uma função, incluindo ou excluindo conjuntos de linhas no resultado ordenado. A cláusula de quadro consiste na palavra-chave ROWS e nos especificadores associados. Consulte Resumo da sintaxe de funções da janela.

Tipos de dados

Os tipos de argumentos compatíveis com a função STDDEV são SMALLINT, INTEGER, BIGINT, NUMERIC, DECIMAL, REAL e DOUBLE PRECISION.

Independente do tipo de dados da expressão, o tipo de retorno de uma função STDDEV é um número de precisão dupla.

Exemplos

O seguinte exemplo mostra como usar as funções STDDEV_POP e VAR_POP como funções da janela. A consulta computa a variação de população e o desvio padrão de população para os valores PRICEPAID na tabela SALES.

```
select salesid, dateid, pricepaid,
round(stddev_pop(pricepaid) over
(order by dateid, salesid rows unbounded preceding)) as stddevpop,
round(var_pop(pricepaid) over
(order by dateid, salesid rows unbounded preceding)) as varpop
```

```
from sales
order by 2,1;

salesid | dateid | pricepaid | stddevpop | varpop
--------+--------+-----------+-----------+---------
  33095 |   1827 |    234.00 |         0 |       0
  65082 |   1827 |    472.00 |       119 |   14161
  88268 |   1827 |    836.00 |       248 |   61283
  97197 |   1827 |    708.00 |       230 |   53019
 110328 |   1827 |    347.00 |       223 |   49845
 110917 |   1827 |    337.00 |       215 |   46159
 150314 |   1827 |    688.00 |       211 |   44414
 157751 |   1827 |   1730.00 |       447 |  199679
 165890 |   1827 |   4192.00 |      1185 | 1403323
...
```

A amostra de desvio padrão e funções de variação podem ser usadas da mesma forma.

Função de janela SUM

A função de janela SUM retorna a soma dos valores de entrada da coluna ou expressão. A função SUM funciona com valores numéricos e ignora valores NULL.

Sintaxe

```
SUM ( [ ALL ] expression ) OVER
(
[ PARTITION BY expr_list ]
[ ORDER BY order_list
                        frame_clause ]
)
```

Argumentos

expressão

A coluna ou expressão de destino na qual a função opera.

ALL

Com o argumento ALL, a função retém todos os valores duplicados da expressão. ALL é o padrão. DISTINCT não é compatível.

OVER

Especifica as cláusulas de janela das funções de agregação. A cláusula OVER distingue funções de agregação de janela das funções de agregação de conjuntos normais.

PARTITION BY expr_list

Define a janela para a função SUM em termos de uma ou mais expressões.

ORDER BY order_list

Classifica as linhas dentro de cada partição. Se nenhuma PARTITION BY for especificada, ORDER BY usa a tabela completa.

frame_clause

Se uma cláusula ORDER BY é usada para uma função agregada, uma cláusula de quadro explícita é necessária. A cláusula de quadro refina o conjunto de linhas na janela de uma função, incluindo ou excluindo conjuntos de linhas no resultado ordenado. A cláusula de quadro consiste na palavra-chave ROWS e nos especificadores associados. Consulte Resumo da sintaxe de funções da janela.

Tipos de dados

Os tipos de argumentos compatíveis com a função SUM são SMALLINT, INTEGER, BIGINT, NUMERIC, DECIMAL, REAL e DOUBLE PRECISION.

Os tipos de retorno compatíveis com a função SUM são:

- BIGINT para argumentos SMALLINT ou INTEGER
- NUMERIC para argumentos BIGINT
- DOUBLE PRECISION para argumentos de ponto flutuante

Exemplos

O seguinte exemplo cria uma soma cumulativa (contínua) das quantidades de vendas solicitadas por data e ID de vendas:

```
select salesid, dateid, sellerid, qty,
sum(qty) over (order by dateid, salesid rows unbounded preceding) as sum
from winsales
```

```
order by 2,1;

salesid |   dateid   | sellerid | qty | sum
--------+------------+----------+-----+-----
30001 | 2003-08-02 |        3 |  10 |  10
10001 | 2003-12-24 |        1 |  10 |  20
10005 | 2003-12-24 |        1 |  30 |  50
40001 | 2004-01-09 |        4 |  40 |  90
10006 | 2004-01-18 |        1 |  10 | 100
20001 | 2004-02-12 |        2 |  20 | 120
40005 | 2004-02-12 |        4 |  10 | 130
20002 | 2004-02-16 |        2 |  20 | 150
30003 | 2004-04-18 |        3 |  15 | 165
30004 | 2004-04-18 |        3 |  20 | 185
30007 | 2004-09-07 |        3 |  30 | 215
(11 rows)
```

Para uma descrição da tabela WINSALES, consulte Amostra de tabela para exemplos de funções de janela.

O seguinte exemplo cria uma soma cumulativa (contínua) de quantidades de vendas por data, particiona os resultados por ID do vendedor e ordena os resultados por data e ID de vendas na partição:

```
select salesid, dateid, sellerid, qty,
sum(qty) over (partition by sellerid
order by dateid, salesid rows unbounded preceding) as sum
from winsales
order by 2,1;

salesid |   dateid   | sellerid | qty | sum
--------+------------+----------+-----+-----
30001 | 2003-08-02 |        3 |  10 |  10
10001 | 2003-12-24 |        1 |  10 |  10
10005 | 2003-12-24 |        1 |  30 |  40
40001 | 2004-01-09 |        4 |  40 |  40
10006 | 2004-01-18 |        1 |  10 |  50
20001 | 2004-02-12 |        2 |  20 |  20
40005 | 2004-02-12 |        4 |  10 |  50
20002 | 2004-02-16 |        2 |  20 |  40
30003 | 2004-04-18 |        3 |  15 |  25
30004 | 2004-04-18 |        3 |  20 |  45
30007 | 2004-09-07 |        3 |  30 |  75
```

```
(11 rows)
```

O seguinte exemplo numera todas as linhas sequencialmente no conjunto de resultados, ordenadas pelas colunas SELLERID e SALESID:

```
select salesid, sellerid, qty,
sum(1) over (order by sellerid, salesid rows unbounded preceding) as rownum
from winsales
order by 2,1;

salesid | sellerid |  qty | rownum
--------+----------+------+--------
10001 |        1 |   10 |      1
10005 |        1 |   30 |      2
10006 |        1 |   10 |      3
20001 |        2 |   20 |      4
20002 |        2 |   20 |      5
30001 |        3 |   10 |      6
30003 |        3 |   15 |      7
30004 |        3 |   20 |      8
30007 |        3 |   30 |      9
40001 |        4 |   40 |     10
40005 |        4 |   10 |     11
(11 rows)
```

Para uma descrição da tabela WINSALES, consulte Amostra de tabela para exemplos de funções de janela.

O seguinte exemplo numera todas as linhas sequencialmente no conjunto de resultados, particiona os resultados por SELLERID e ordena os resultados por SELLERID e SALESID na partição:

```
select salesid, sellerid, qty,
sum(1) over (partition by sellerid
order by sellerid, salesid rows unbounded preceding) as rownum
from winsales
order by 2,1;

salesid | sellerid | qty | rownum
---------+----------+-----+--------
10001 |        1 |  10 |      1
10005 |        1 |  30 |      2
10006 |        1 |  10 |      3
```

```
20001 |        2 |  20 |      1
20002 |        2 |  20 |      2
30001 |        3 |  10 |      1
30003 |        3 |  15 |      2
30004 |        3 |  20 |      3
30007 |        3 |  30 |      4
40001 |        4 |  40 |      1
40005 |        4 |  10 |      2
(11 rows)
```

Funções de janela VAR_SAMP e VAR_POP

As funções da janela VAR_SAMP e VAR_POP retornam a variação da amostra e da população de um conjunto de valores numéricos (número inteiro, decimal ou ponto flutuante). Consulte também Funções VAR_SAMP e VAR_POP.

VAR_SAMP e VARIANCE são sinônimos para a mesma função.

Sintaxe

```
VAR_SAMP | VARIANCE | VAR_POP
( [ ALL ] expression ) OVER
(
[ PARTITION BY expr_list ]
[ ORDER BY order_list
                        frame_clause ]
)
```

Argumentos

expressão

A coluna ou expressão de destino na qual a função opera.

ALL

Com o argumento ALL, a função retém todos os valores duplicados da expressão. ALL é o padrão. DISTINCT não é compatível.

OVER

Especifica as cláusulas de janela das funções de agregação. A cláusula OVER distingue funções de agregação de janela das funções de agregação de conjuntos normais.

PARTITION BY expr_list

Define a janela para a função em termos de uma ou mais expressões.

ORDER BY order_list

Classifica as linhas dentro de cada partição. Se nenhuma PARTITION BY for especificada, ORDER BY usa a tabela completa.

frame_clause

Se uma cláusula ORDER BY é usada para uma função agregada, uma cláusula de quadro explícita é necessária. A cláusula de quadro refina o conjunto de linhas na janela de uma função, incluindo ou excluindo conjuntos de linhas no resultado ordenado. A cláusula de quadro consiste na palavra-chave ROWS e nos especificadores associados. Consulte Resumo da sintaxe de funções da janela.

Tipos de dados

Os tipos de argumentos compatíveis com a função VARIANCE são SMALLINT, INTEGER, BIGINT, NUMERIC, DECIMAL, REAL e DOUBLE PRECISION.

Independente do tipo de dados da expressão, o tipo de retorno de uma função VARIANCE é um número de precisão dupla.

Funções de administração do sistema

Tópicos

- CHANGE_QUERY_PRIORITY
- CHANGE_SESSION_PRIORITY
- CHANGE_USER_PRIORITY
- CURRENT_SETTING
- PG_CANCEL_BACKEND
- PG_TERMINATE_BACKEND
- REBOOT_CLUSTER
- SET_CONFIG

O Amazon Redshift é compatível com várias funções de administração de sistema.

CHANGE_QUERY_PRIORITY

CHANGE_QUERY_PRIORITY permite que superusuários modifiquem a prioridade de uma consulta que está em execução ou aguardando no gerenciamento de workload (WLM).

Essa função permite que superusuários alterem imediatamente a prioridade de qualquer consulta no sistema. Somente uma consulta, uma sessão ou um usuário pode ser executado com a prioridade `CRITICAL`.

Sintaxe

```
CHANGE_QUERY_PRIORITY(query_id, priority)
```

Argumentos

query_id

O identificador da consulta cuja prioridade é alterada. Requer um valor `INTEGER`.

priority

A nova prioridade que será atribuída à consulta. Esse argumento deve ser uma string com o valor `CRITICAL`, `HIGHEST`, `HIGH`, `NORMAL`, `LOW` ou `LOWEST`.

Tipo de retorno

Nenhum

Exemplos

Para mostrar a coluna `query_priority` na tabela do sistema STV_WLM_QUERY_STATE, use o exemplo a seguir.

```
SELECT query, service_class, query_priority, state
FROM stv_wlm_query_state WHERE service_class = 101;

+-------+---------------+----------------+---------+
| query | service_class | query_priority |  state  |
+-------+---------------+----------------+---------+
|  1076 |           101 | Lowest         | Running |
|  1075 |           101 | Lowest         | Running |
+-------+---------------+----------------+---------+
```

Para mostrar os resultados de um superusuário que executa a função `change_query_priority` a fim de alterar a prioridade para `CRITICAL`, use o exemplo a seguir.

```
SELECT CHANGE_QUERY_PRIORITY(1076, 'Critical');

+-------------------------------------------------------------------------------+
|                            change_query_priority                              |
+-------------------------------------------------------------------------------+
| Succeeded to change query priority. Priority changed from Lowest to Critical. |
+-------------------------------------------------------------------------------+
```

CHANGE_SESSION_PRIORITY

CHANGE_SESSION_PRIORITY permite que superusuários alterem imediatamente a prioridade de qualquer sessão no sistema. Somente uma consulta, uma sessão ou um usuário pode ser executado com a prioridade `CRITICAL`.

Sintaxe

```
CHANGE_SESSION_PRIORITY(pid, priority)
```

Argumentos

pid

O identificador de processo da sessão cuja prioridade é alterada. O valor `-1` se refere à sessão atual. Requer um valor `INTEGER`.

priority

A nova prioridade que será atribuída à sessão. Esse argumento deve ser uma string com o valor `CRITICAL`, `HIGHEST`, `HIGH`, `NORMAL`, `LOW` ou `LOWEST`.

Tipo de retorno

Nenhum

Exemplos

Para retornar o identificador do processo do servidor que lida com a sessão atual, use o exemplo a seguir.

```
SELECT pg_backend_pid();

+----------------+
| pg_backend_pid |
+----------------+
|          30311 |
+----------------+
```

Neste exemplo, a prioridade da sessão atual é alterada para `LOWEST`.

```
SELECT CHANGE_SESSION_PRIORITY(30311, 'Lowest');

+-----------------------------------------------------------------------------------------
+
|                                change_session_priority
 |
+-----------------------------------------------------------------------------------------
+
| Succeeded to change session priority. Changed session (pid:30311) priority to lowest.
 |
+-----------------------------------------------------------------------------------------
+
```

Neste exemplo, a prioridade da sessão atual é alterada para `HIGH`.

```
SELECT CHANGE_SESSION_PRIORITY(-1, 'High');

+-----------------------------------------------------------------------------------------
+
|                                    change_session_priority
           |
+-----------------------------------------------------------------------------------------
+
| Succeeded to change session priority. Changed session (pid:30311) priority from
 lowest to high. |
+-----------------------------------------------------------------------------------------
+
```

Para criar um procedimento armazenado que altera a prioridade de uma sessão, use o exemplo a seguir. A permissão para executar esse procedimento armazenado é concedida ao usuário do banco de dados `test_user`.

```
CREATE OR REPLACE PROCEDURE sp_priority_low(pid IN int, result OUT varchar)
AS $$
BEGIN
  SELECT CHANGE_SESSION_PRIORITY(pid, 'low') into result;
END;
$$ LANGUAGE plpgsql
SECURITY DEFINER;
GRANT EXECUTE ON PROCEDURE sp_priority_low(int) TO test_user;
```

Depois, o usuário do banco de dados denominado `test_user` chama o procedimento.

```
CALL sp_priority_low(pg_backend_pid());

+-----------------------------------------------------+
|                        result                       |
+-----------------------------------------------------+
| Success. Change session (pid:13155) priority to low. |
+-----------------------------------------------------+
```

CHANGE_USER_PRIORITY

CHANGE_USER_PRIORITY permite que superusuários modifiquem a prioridade de todas as consultas emitidas por um usuário que estejam em execução ou aguardando no gerenciamento de workload (WLM). Somente uma consulta, uma sessão ou um usuário pode ser executado com a prioridade `CRITICAL`.

Sintaxe

```
CHANGE_USER_PRIORITY(user_name, priority)
```

Argumentos

user_name

O nome do usuário do banco de dados cuja prioridade da consulta é alterada.

priority

A nova prioridade que será atribuída a todas as consultas emitidas por `user_name`. Esse argumento deve ser uma string com o valor `CRITICAL`, `HIGHEST`, `HIGH`, `NORMAL`, `LOW`, `LOWEST`

ou RESET. Somente superusuários podem alterar a prioridade para CRITICAL. Alterar a prioridade para RESET remove a configuração de prioridade de user_name.

Tipo de retorno

Nenhum

Exemplos

Para alterar a prioridade do usuário analysis_user para LOWEST, use o exemplo a seguir.

```
SELECT CHANGE_USER_PRIORITY('analysis_user', 'lowest');

+-------------------------------------------------------------------------------------+
|                               change_user_priority                                  |
+-------------------------------------------------------------------------------------+
| Succeeded to change user priority. Changed user (analysis_user) priority to lowest. |
+-------------------------------------------------------------------------------------+
```

Para alterar a prioridade para LOW, use o exemplo a seguir.

```
SELECT CHANGE_USER_PRIORITY('analysis_user', 'low');

+-------------------------------------------------------------------------------------------------------
+
|                                    change_user_priority
        |
+-------------------------------------------------------------------------------------------------------
+
| Succeeded to change user priority. Changed user (analysis_user) priority from Lowest
 to low. |
+-------------------------------------------------------------------------------------------------------
+
```

Para redefinir a prioridade, use o exemplo a seguir.

```
SELECT CHANGE_USER_PRIORITY('analysis_user', 'reset');

+-----------------------------------------------------+
|                change_user_priority                 |
+-----------------------------------------------------+
| Succeeded to reset priority for user (analysis_user). |
```

```
+-----------------------------------------------------+
```

CURRENT_SETTING

CURRENT_SETTING retorna o valor atual do parâmetro de configuração especificado.

Essa função é equivalente ao comando SHOW.

Sintaxe

```
current_setting('parameter')
```

A instrução a seguir retorna o valor atual da variável de contexto de sessão especificada.

```
current_setting('variable_name')
current_setting('variable_name'[, error_if_undefined])
```

Argumentos

Parâmetro

Valor de parâmetro a exibir. Para obter uma lista dos parâmetros de configuração, consulte Referência da configuração

variable_name

O nome da variável a ser exibida. Ela deve ser uma constante de string para variáveis de contexto de sessão.

error_if_undefined

(Opcional) Um valor booliano opcional que especifica o comportamento caso o nome da variável não exista. Quando error_if_undefined é definido como TRUE, que é o padrão, o Amazon Redshift lança um erro. Quando error_if_undefined é definido como FALSE, o Amazon Redshift retorna NULL. O Amazon Redshift é compatível com o parâmetro error_if_undefined somente para variáveis de contexto de sessão. Isso não pode ser usado quando a entrada é um parâmetro de configuração.

Tipo de retorno

Retorna uma string CHAR ou VARCHAR.

Exemplos

Para retornar a configuração atual para o parâmetro `query_group`, use o exemplo a seguir.

```
SELECT CURRENT_SETTING('query_group');

+-----------------+
| current_setting |
+-----------------+
| unset           |
+-----------------+
```

Para retornar a configuração atual para a variável `app_context.user_id`, use o exemplo a seguir.

```
SELECT CURRENT_SETTING('app_context.user_id', FALSE);
```

PG_CANCEL_BACKEND

Cancela uma consulta. PG_CANCEL_BACKEND é funcionalmente equivalente ao comando CANCEL. Você pode cancelar consultas atualmente em execução pelo seu usuário. Superusuários podem cancelar qualquer consulta.

Sintaxe

```
pg_cancel_backend( pid )
```

Argumentos

pid

O ID de processo (PID) da consulta a ser cancelada. Você não pode cancelar uma consulta especificando um ID de consulta; você deve especificar o ID de processo da consulta. Requer um valor `INTEGER`.

Tipo de retorno

Nenhum

Observações de uso

Se consultas em várias sessões têm bloqueios na mesma tabela, você pode usar a função PG_TERMINATE_BACKEND para encerrar uma das sessões, o que força todas as transações atualmente em execução na sessão encerrada a liberar todos os bloqueios e reverter a transação. Consulte a tabela de catálogo PG__LOCKS para visualizar os bloqueios atuais. Se você não puder cancelar uma consulta pois ela está em um bloco de transação (BEGIN… END), você pode encerrar a sessão na qual a consulta está em execução usando a função PG_TERMINATE_BACKEND.

Exemplos

Para cancelar uma consulta atualmente em execução, recupere primeiro o ID de processo para a consulta que você deseja cancelar. Para determinar os IDs de processo para todas as consultas em execução atualmente, execute o comando a seguir.

```
SELECT pid, TRIM(starttime) AS start,
duration, TRIM(user_name) AS user,
SUBSTRING(query,1,40) AS querytxt
FROM stv_recents
WHERE status = 'Running';

+-----+-----------------------+----------+--------+-----------------------------+
| pid |       starttime       | duration |  user  |          querytxt           |
+-----+-----------------------+----------+--------+-----------------------------+
| 802 | 2013-10-14 09:19:03.55 |      132 | dwuser | select venuename from venue |
| 834 | 2013-10-14 08:33:49.47 |  1250414 | dwuser | select * from listing;      |
| 964 | 2013-10-14 08:30:43.29 |   326179 | dwuser | select sellerid from sales  |
+-----+-----------------------+----------+--------+-----------------------------+
```

Para cancelar a consulta com o ID de processo 802, use o exemplo a seguir.

```
SELECT PG_CANCEL_BACKEND(802);
```

PG_TERMINATE_BACKEND

Encerra uma sessão. Você pode encerrar uma sessão de propriedade de seu usuário. Um superusuário pode encerrar qualquer sessão.

Sintaxe

```
pg_terminate_backend( pid )
```

Argumentos

pid

O ID de processo da sessão a ser encerrada. Requer um valor `INTEGER`.

Tipo de retorno

Nenhum

Observações de uso

Se você estiver próximos de alcançar o limite de conexões simultâneas, use PG_TERMINATE_BACKEND para encerrar sessões ociosas e liberar as conexões. Para obter mais informações, consulte Limites no Amazon Redshift.

Se consultas em várias sessões têm bloqueios na mesma tabela, você pode usar a função PG_TERMINATE_BACKEND para encerrar uma das sessões, o que força todas as transações atualmente em execução na sessão encerrada a liberar todos os bloqueios e reverter a transação. Consulte a tabela de catálogo PG_LOCKS para visualizar os bloqueios atuais.

Se uma consulta não estiver em um bloco de transação (BEGIN… END), você pode cancelar a consulta usando o comando CANCEL ou a função PG_CANCEL_BACKEND.

Exemplos

Para consultar a tabela SVV_TRANSACTIONS a fim de visualizar todos os bloqueios em vigor para transações atuais, use o exemplo a seguir..

```
SELECT * FROM svv_transactions;

+----------+--------+-------+------+--------------------+----------------
+----------------------+----------+---------+
| txn_owner | txn_db |  xid  | pid  |      txn_start      |    lock_mode    |
 lockable_object_type | relation | granted |
+----------+--------+-------+------+--------------------+----------------
+----------------------+----------+---------+
| rsuser    | dev    | 96178 | 8585 | 2017-04-12 20:13:07 | AccessShareLock | relation
           |    51940 | true    |
| rsuser    | dev    | 96178 | 8585 | 2017-04-12 20:13:07 | AccessShareLock | relation
           |    52000 | true    |
| rsuser    | dev    | 96178 | 8585 | 2017-04-12 20:13:07 | AccessShareLock | relation
           |   108623 | true    |
```

```
| rsuser    | dev    | 96178 | 8585 | 2017-04-12 20:13:07 | ExclusiveLock   |
 transactionid        |          | true    |
+-----------+--------+-------+------+---------------------+-----------------
+----------------------+----------+---------+
```

Para encerrar a sessão com os bloqueios, use o exemplo a seguir.

```
SELECT PG_TERMINATE_BACKEND(8585);
```

REBOOT_CLUSTER

Reinicializa o cluster do Amazon Redshift sem encerrar as conexões com o cluster. Você deve ser um superusuário do banco de dados para executar este comando.

Após a conclusão dessa reinicialização suave, o cluster do Amazon Redshift retorna um erro à aplicação do usuário e exige que a aplicação do usuário reenvie quaisquer transações ou consultas interrompidas pela reinicialização suave.

Sintaxe

```
SELECT REBOOT_CLUSTER();
```

SET_CONFIG

Define um parâmetro de configuração para uma nova definição.

Essa função é equivalente ao comando SET em SQL.

Sintaxe

```
SET_CONFIG('parameter', 'new_value' , is_local)
```

A instrução a seguir define uma variável de contexto de sessão para uma nova configuração.

```
set_config('variable_name', 'new_value' , is_local)
```

Argumentos

Parâmetro

Parâmetro a definir.

variable_name

O nome da variável a ser definida.

new_value

Novo valor para o parâmetro.

is_local

Se true, o valor de parâmetro se aplica somente à transação atual. Os valores válidos `true` ou `1` e `false` ou `0`.

Tipo de retorno

Retorna uma string `CHAR` ou `VARCHAR`.

Exemplos

Para definir o valor do parâmetro `query_group` como `test` somente para a transação atual, use o exemplo a seguir.

```
SELECT SET_CONFIG('query_group', 'test', true);

+------------+
| set_config |
+------------+
| test       |
+------------+
```

Para definir variáveis de contexto de sessão, use o exemplo a seguir.

```
SELECT SET_CONFIG('app.username', 'cuddy', FALSE);
```

Funções de informação do sistema

O Amazon Redshift é compatível com diversas funções de informação de sistema.

Tópicos

- CURRENT_AWS_ACCOUNT

- CURRENT_DATABASE
- CURRENT_NAMESPACE
- CURRENT_SCHEMA
- CURRENT_SCHEMAS
- CURRENT_SESSION_ARN
- CURRENT_USER
- CURRENT_USER_ID
- DB_COLLATION
- DEFAULT_IAM_ROLE
- GET_MOUNTED_ROLE
- HAS_ASSUMEROLE_PRIVILEGE
- HAS_DATABASE_PRIVILEGE
- HAS_SCHEMA_PRIVILEGE
- HAS_TABLE_PRIVILEGE
- LAST_USER_QUERY_ID
- PG_BACKEND_PID
- PG_GET_COLS
- PG_GET_GRANTEE_BY_IAM_ROLE
- PG_GET_IAM_ROLE_BY_USER
- PG_GET_LATE_BINDING_VIEW_COLS
- PG_GET_SESSION_ROLES
- PG_LAST_COPY_COUNT
- PG_LAST_COPY_ID
- PG_LAST_UNLOAD_ID
- PG_LAST_QUERY_ID
- PG_LAST_UNLOAD_COUNT
- Função SLICE_NUM
- USER

- ROLE_IS_MEMBER_OF
- USER_IS_MEMBER_OF
- VERSION

CURRENT_AWS_ACCOUNT

Retorna a conta da AWS associada ao cluster do Amazon Redshift que enviou uma consulta.

Sintaxe

```
current_aws_account
```

Tipo de retorno

Retorna um número inteiro.

Exemplo

A consulta a seguir retorna o nome do banco de dados atual.

```
select user, current_aws_account;
current_user | current_account
-------------+---------------
dwuser       | 987654321

(1 row)
```

CURRENT_DATABASE

Retorna o nome do banco de dados ao qual você está atualmente conectado.

Sintaxe

```
current_database()
```

Tipo de retorno

Retorna uma string CHAR ou VARCHAR.

Exemplo

A consulta a seguir retorna o nome do banco de dados atual.

```
select current_database();

current_database
------------------
tickit
(1 row)
```

CURRENT_NAMESPACE

Retorna o namespace de cluster do cluster atual do Amazon Redshift. O namespace do cluster do Amazon Redshift é a ID exclusiva do cluster do Amazon Redshift.

Sintaxe

```
current_namespace
```

Tipo de retorno

Retorna uma string CHAR ou VARCHAR.

Exemplo

A consulta a seguir retorna o nome do namespace atual.

```
select user, current_namespace;
current_user | current_namespace
-------------+--------------------------------------
dwuser       | 86b5169f-01dc-4a6f-9fbb-e2e24359e9a8

(1 row)
```

CURRENT_SCHEMA

Retorna o nome do esquema à frente do caminho de pesquisa. Este esquema será utilizado para todas as tabelas ou outros objetos nomeados que foram criados sem especificação de um esquema de destino.

Sintaxe

> **Note**
>
> Essa é uma função do nó de liderança. Essa função retorna um erro se fizer referência a uma tabela criada por usuário, a uma tabela de sistema STL ou STV ou a uma exibição de sistema SVV ou SVL.

```
current_schema()
```

Tipo de retorno

CURRENT_SCHEMA retorna uma string CHAR ou VARCHAR.

Exemplos

A seguinte consulta retorna o esquema atual:

```
select current_schema();

current_schema
---------------
public
(1 row)
```

CURRENT_SCHEMAS

Retorna uma matriz de nomes de todos os esquemas no caminho de pesquisa atual. O caminho de pesquisa atual é definido no parâmetro search_path.

Sintaxe

> **Note**
>
> Essa é uma função do nó de liderança. Essa função retorna um erro se fizer referência a uma tabela criada por usuário, a uma tabela de sistema STL ou STV ou a uma exibição de sistema SVV ou SVL.

```
current_schemas(include_implicit)
```

Argumento

include_implicit

Se true, especifica que o caminho de pesquisa deve incluir todos os esquemas de sistema incluídos implicitamente. Os valores válidos são `true` e `false`. Normalmente, se `true`, este parâmetro retorna o esquema `pg_catalog` além do esquema atual.

Tipo de retorno

Retorna uma string CHAR ou VARCHAR.

Exemplos

O seguinte exemplo retorna os nomes dos esquemas no caminho de pesquisa atual, sem incluir esquemas de sistema incluídos implicitamente:

```
select current_schemas(false);

current_schemas
-----------------
{public}
(1 row)
```

O seguinte exemplo retorna os nomes dos esquemas no caminho de pesquisa atual, incluindo esquemas de sistema incluídos implicitamente:

```
select current_schemas(true);

current_schemas
---------------------
{pg_catalog,public}
(1 row)
```

CURRENT_SESSION_ARN

Retorna o ARN do usuário global atualmente autorizado. Usuários globais apresentam a mesma identidade em todas as contas, clusters e grupos de trabalho de tecnologia sem servidor do Redshift.

Usuários globais fazem login por meio do Centro de Identidade do IAM ou por meio da autenticação de sessão baseada no IAM. Os usuários do data lake são usuários da AWS globais.

Essa função normalmente é usada no contexto do uso de visualizações do AWS Glue com vários dialetos. Consulte mais informações sobre o gerenciamento de identidades com o Centro de Identidade do IAM e o Redshift em Conectar o Redshift ao IAM Identity Center para proporcionar aos usuários uma experiência de logon único. Consulte mais informações sobre visualizações do Glue com vários dialetos em Criação de exibições no AWS Glue Glue Data Catalog (visualização).

Sintaxe

```
current_session_arn()
```

Tipo de retorno

Retorna uma string VARCHAR do usuário autenticado globalmente ou um valor nulo.

Observações de uso

Os usuários locais não são compatíveis e resultam em uma resposta nula.

Exemplo

A seguinte consulta retorna o nome do ARN da sessão atual:

```
SELECT current_session_arn();

current_session_arn
--------------
arn:aws:iam::123456789012:user/user
(1 row)
```

CURRENT_USER

Retorna o nome de usuário do atual usuário "efetivo" do banco de dados, conforme aplicável para verificação de permissões. Geralmente, esse nome de usuário será o mesmo que o usuário de sessão; porém, isso pode ocasionalmente alterado por superusuários.

> **Note**
>
> Não use parênteses finais ao chamar CURRENT_USER.

Sintaxe

```
current_user
```

Tipo de retorno

CURRENT_USER exibe um tipo de dados NAME e pode ser convertido como uma string CHAR ou VARCHAR.

Observações de uso

Se um procedimento armazenado tiver sido criado usando a opção SECURITY DEFINER do comando CREATE_PROCEDURE, ao invocar a função CURRENT_USER de dentro do procedimento armazenado, o Amazon Redshift retornará o nome de usuário do proprietário do procedimento armazenado.

Exemplo

A seguinte consulta retorna o nome do atual usuário do banco de dados:

```
select current_user;

current_user
--------------
dwuser
(1 row)
```

CURRENT_USER_ID

Retorna o identificador exclusivo para o usuário do Amazon Redshift conectado à sessão atual.

Sintaxe

```
CURRENT_USER_ID
```

Tipo de retorno

A função CURRENT_USER_ID retorna um inteiro.

Exemplos

O seguinte exemplo retorna o nome de usuário e o ID do usuário atual para esta sessão:

```
select user, current_user_id;

 current_user | current_user_id
--------------+-----------------
   dwuser     |               1
(1 row)
```

DB_COLLATION

Retorna a configuração de agrupamento do banco de dados atual.

Sintaxe

```
db_collation()
```

Tipo de retorno

Retorna uma string VARCHAR que representa o agrupamento do banco de dados atual. Os valores possíveis são `case_sensitive` ou `case_insensitive`.

Exemplo

O exemplo a seguir retorna o agrupamento do banco de dados atual.

```
select db_collation();

db_collation
----------------
case_sensitive
(1 row)
```

DEFAULT_IAM_ROLE

Retorna a função padrão do IAM associada atualmente ao cluster do Amazon Redshift. A função retornará “none” se não houver nenhuma função padrão do IAM associada.

Sintaxe

```
select default_iam_role();
```

Tipo de retorno

Retorna uma string VARCHAR.

Exemplo

O exemplo a seguir retorna a função padrão do IAM associada atualmente ao cluster especificado do Amazon Redshift,

```
select default_iam_role();
              default_iam_role
---------------------------------------------
 arn:aws:iam::123456789012:role/myRedshiftRole
(1 row)
```

GET_MOUNTED_ROLE

Quando invocado como parte de uma visualização do AWS Glue com vários dialetos, permite retornar o perfil do IAM usado para montar o esquema ou banco de dados do Lake Formation. Ter vários dialetos significa que o SQL é compatível com diversos mecanismos de consulta, como o Amazon EMR e o Redshift. Consulte mais informações sobre visualizações do Glue com vários dialetos em Criação de exibições no AWS Glue Glue Data Catalog (visualização).

Sintaxe

```
get_mounted_role()
```

Tipo de retorno

Retorna uma string VARCHAR ou um valor nulo.

Observações de uso

Essa função retorna um valor nulo para qualquer caso de uso fora de uma visualização externa do Lake Formation.

Exemplo

A consulta a seguir retorna a identidade para montar o recurso do Lake Formation.

```
CREATE EXTERNAL PROTECTED VIEW external_schema.remote_view AS
```

```
SELECT mycol, get_mounted_role() FROM external_schema.remote_table;

mycol | get_mounted_role
--------------------------
1       arn:aws:iam::123456789012:role/salesrole
(1 row)
```

HAS_ASSUMEROLE_PRIVILEGE

Retorna booliano `true` (`t`) se o usuário tiver a função do IAM especificada com o privilégio de executar o comando especificado. A função retorna `false` (`f`) se o usuário não tiver a função do IAM especificada com o privilégio de executar o comando especificado. Para obter mais informações sobre privilégios, consulte GRANT.

Sintaxe

```
has_assumerole_privilege( [ user, ] iam_role_arn, cmd_type)
```

Argumentos

usuário

O nome do usuário para verificar os privilégios de função do IAM. O padrão é verificar o usuário atual. Superusuários e usuários podem usar essa função. No entanto, os usuários podem somente exibir seus próprios privilégios.

iam_role_arn

A função do IAM que recebeu os privilégios de comando.

cmd_type

O comando para o qual o acesso foi concedido. Os valores válidos são os seguintes:

- COPY
- UNLOAD
- EXTERNAL FUNCTION
- CREATE MODEL

Tipo de retorno

BOOLEAN

Exemplo

A consulta a seguir confirma que o usuário `reg_user1` tem o privilégio para a função `Redshift-S3-Read` para executar o comando COPY.

```
select has_assumerole_privilege('reg_user1', 'arn:aws:iam::123456789012:role/Redshift-S3-Read', 'copy');
```

```
has_assumerole_privilege
------------------------
true
(1 row)
```

HAS_DATABASE_PRIVILEGE

Retorna `true` se o usuário tem o privilégio especificado para o banco de dados especificado. Para obter mais informações sobre privilégios, consulte GRANT.

Sintaxe

> **Note**
>
> Essa é uma função do nó de liderança. Essa função retorna um erro se fizer referência a uma tabela criada por usuário, a uma tabela de sistema STL ou STV ou a uma exibição de sistema SVV ou SVL.

```
has_database_privilege( [ user, ] database, privilege)
```

Argumentos

usuário

O nome do usuário para verificar os privilégios do banco de dados. O padrão é verificar o usuário atual.

banco de dados

O banco de dados associado ao privilégio.

privilege

O privilégio de verificar. Os valores válidos são os seguintes:

- CREATE
- TEMPORARY
- TEMP

Tipo de retorno

Retorna uma string CHAR ou VARCHAR.

Exemplo

A consulta a seguir confirma que o usuário GUEST tem o privilégio TEMP no banco de dados TICKIT.

```
select has_database_privilege('guest', 'tickit', 'temp');

has_database_privilege
------------------------
true
(1 row)
```

HAS_SCHEMA_PRIVILEGE

Retorna `true` se o usuário tiver o privilégio especificado para o esquema especificado. Para obter mais informações sobre privilégios, consulte GRANT.

Sintaxe

> **Note**
>
> Essa é uma função do nó de liderança. Essa função retorna um erro se fizer referência a uma tabela criada por usuário, a uma tabela de sistema STL ou STV ou a uma exibição de sistema SVV ou SVL.

```
has_schema_privilege( [ user, ] schema, privilege)
```

Argumentos

usuário

O nome do usuário para verificar os privilégios do esquema. O padrão é verificar o usuário atual.

esquema

O esquema associado ao privilégio.

privilege

O privilégio de verificar. Os valores válidos são os seguintes:

- CREATE
- USAGE
- ALTER
- DROP

Tipo de retorno

Retorna uma string CHAR ou VARCHAR.

Exemplo

A seguinte consulta confirma que o usuário GUEST tem o privilégio CREATE no esquema PUBLIC:

```
select has_schema_privilege('guest', 'public', 'create');

has_schema_privilege
----------------------
true
(1 row)
```

HAS_TABLE_PRIVILEGE

Retorna `true` se o usuário tiver o privilégio especificado para a tabela especificada; caso contrário, retorna `false`.

Sintaxe

> **Note**
>
> Essa é uma função do nó de liderança. Essa função retorna um erro se fizer referência a uma tabela criada por usuário, a uma tabela de sistema STL ou STV ou a uma exibição de sistema SVV ou SVL. Para obter mais informações sobre privilégios, consulte GRANT.

```
has_table_privilege( [ user, ] table, privilege)
```

Argumentos

usuário

O nome do usuário para verificar os privilégios da tabela. O padrão é verificar o usuário atual.

table

Tabela associada ao privilégio.

privilege

Privilégio a verificar. Os valores válidos são os seguintes:

- SELECT
- INSERT
- UPDATE
- DELETE
- DROP
- REFERENCES

Tipo de retorno

BOOLEAN

Exemplos

A consulta a seguir descobre que o usuário GUEST não tem privilégio SELECT na tabela LISTING.

```
select has_table_privilege('guest', 'listing', 'select');
```

```
has_table_privilege
--------------------
false
```

A consulta a seguir lista os privilégios da tabela, incluindo seleção, inserção, atualização e exclusão, usando a saída das tabelas de catálogo pg_tables e pg_user. Isso é apenas um exemplo. Talvez seja necessário especificar um nome de esquema e nomes de tabelas do seu banco de dados. Para obter mais informações, consulte Consultar as tabelas de catálogo.

```
SELECT
     tablename
     ,usename
     ,HAS_TABLE_PRIVILEGE(users.usename, tablename, 'select') AS sel
     ,HAS_TABLE_PRIVILEGE(users.usename, tablename, 'insert') AS ins
     ,HAS_TABLE_PRIVILEGE(users.usename, tablename, 'update') AS upd
     ,HAS_TABLE_PRIVILEGE(users.usename, tablename, 'delete') AS del
FROM
(SELECT * from pg_tables
WHERE schemaname = 'public' and tablename in ('event','listing')) as tables
,(SELECT * FROM pg_user) AS users;

tablename | usename   |  sel   |  ins  |  upd  | del
----------+-----------+--------+-------+-------+-------
event     |  john     |  true  | true  | true  | true
event     |  sally    |  false | false | false | false
event     |  elsa     |  false | false | false | false
listing   |  john     |  true  | true  | true  | true
listing   |  sally    |  false | false | false | false
listing   |  elsa     |  false | false | false | false
```

A consulta anterior também contém uma junção cruzada. Para obter mais informações, consulte Exemplos de JOIN. Para consultar tabelas que não estão no esquema `public`, remova a condição `schemaname` da cláusula WHERE e use o exemplo a seguir antes da consulta.

```
SET SEARCH_PATH to 'schema_name';
```

LAST_USER_QUERY_ID

Retorna o ID da consulta mais recente concluída por um usuário na sessão atual. Se nenhuma consulta tiver sido executada na sessão atual, last_user_query_id retornará -1. A função não

retorna o ID de consulta para consultas executadas exclusivamente no nó líder. Para obter mais informações, consulte Função de apenas nó líder.

Sintaxe

```
last_user_query_id()
```

Tipo de retorno

Retorna um número inteiro.

Exemplo

A consulta a seguir retorna o ID da consulta concluída mais recente executada por um usuário na sessão atual.

```
select last_user_query_id();
```

A seguir estão os resultados.

```
last_user_query_id
-----------------------
    5437
(1 row)
```

A consulta a seguir retorna o ID e o texto da consulta concluída mais recente executada por um usuário na sessão atual.

```
select query_id, query_text from sys_query_history where query_id =
 last_user_query_id();
```

A seguir estão os resultados.

```
 query_id, query_text
---------
+-----------------------------------------------------------------------------------------------
 5556975 | select last_user_query_id() limit 100 --RequestID=<unique request ID>;
 TraceID=<unique trace ID>
```

PG_BACKEND_PID

Retorna o ID de processo (PID) do processo de servidor responsável pela sessão atual.

> **Note**
>
> O PID não é exclusivo globalmente. Ele pode ser reutilizado ao longo do tempo.

Sintaxe

```
pg_backend_pid()
```

Tipo de retorno

Retorna um número inteiro.

Exemplo

Você pode correlacionar PG_BACKEND_PID com tabelas de log para recuperar informações para a sessão atual. Por exemplo, a consulta a seguir retorna o ID de consulta e uma parte do texto da consulta para as consultas concluídas na sessão atual.

```
select query, substring(text,1,40)
from stl_querytext
where pid =  PG_BACKEND_PID()
order by query desc;

 query |                substring
-------+----------------------------------------
 14831 | select query, substring(text,1,40) from
 14827 | select query, substring(path,0,80) as pa
 14826 | copy category from 's3://dw-tickit/manif
 14825 | Count rows in target table
 14824 | unload ('select * from category') to 's3
(5 rows)
```

Você pode correlacionar PG_BACKEND_PID com a coluna pid nas seguintes tabelas de log (as exceções são indicadas entre parênteses):

- STL_CONNECTION_LOG

- STL_DDLTEXT
- STL_ERROR
- STL_QUERY
- STL_QUERYTEXT
- STL_SESSIONS (process)
- STL_TR_CONFLICT
- STL_UTILITYTEXT
- STV_ACTIVE_CURSORS
- STV_INFLIGHT
- STV_LOCKS (lock_owner_pid)
- STV_RECENTS (process_id)

PG_GET_COLS

Retorna os metadados de coluna para uma definição de tabela ou de exibição.

Sintaxe

```
pg_get_cols('name')
```

Argumentos

name

O nome de uma tabela ou visualização do Amazon Redshift. Para obter mais informações, consulte Nomes e identificadores.

Tipo de retorno

VARCHAR

Observações de uso

A função PG_GET_COLS retorna uma linha para cada coluna em uma definição de tabela ou de exibição. A linha contém uma lista, separada por vírgulas, com nome do esquema, nome do relacionamento, nome da coluna, tipo de dados e número da coluna. A formatação do resultado do SQL depende do cliente SQL utilizado.

Exemplos

Os exemplos a seguir exibem resultados de uma visualização denominada SALES_VW no esquema public e uma tabela de tabela denominada sales no esquema mytickit1 que são criadas pelo usuário no banco de dados conectado dev.

O exemplo a seguir exibe os metadados de coluna para uma visualização chamada SALES_VW.

```
select pg_get_cols('sales_vw');

pg_get_cols
---------------------------------------------------------------
(public,sales_vw,salesid,integer,1)
(public,sales_vw,listid,integer,2)
(public,sales_vw,sellerid,integer,3)
(public,sales_vw,buyerid,integer,4)
(public,sales_vw,eventid,integer,5)
(public,sales_vw,dateid,smallint,6)
(public,sales_vw,qtysold,smallint,7)
(public,sales_vw,pricepaid,"numeric(8,2)",8)
(public,sales_vw,commission,"numeric(8,2)",9)
(public,sales_vw,saletime,"timestamp without time zone",10)
```

O exemplo a seguir exibe os metadados de coluna para a visualização SALES_VW em formato de tabela.

```
select * from pg_get_cols('sales_vw')
cols(view_schema name, view_name name, col_name name, col_type varchar, col_num int);

view_schema | view_name | col_name   | col_type                    | col_num
------------+-----------+------------+-----------------------------+--------
public      | sales_vw  | salesid    | integer                     |       1
public      | sales_vw  | listid     | integer                     |       2
public      | sales_vw  | sellerid   | integer                     |       3
public      | sales_vw  | buyerid    | integer                     |       4
public      | sales_vw  | eventid    | integer                     |       5
public      | sales_vw  | dateid     | smallint                    |       6
public      | sales_vw  | qtysold    | smallint                    |       7
public      | sales_vw  | pricepaid  | numeric(8,2)                |       8
public      | sales_vw  | commission | numeric(8,2)                |       9
public      | sales_vw  | saletime   | timestamp without time zone |      10
```

O exemplo a seguir exibe os metadados de coluna para a tabela SALES no esquema mytickit1 em formato de tabela.

```
select * from pg_get_cols('"mytickit1"."sales"')
cols(view_schema name, view_name name, col_name name, col_type varchar, col_num int);

view_schema | view_name | col_name   | col_type                    | col_num
------------+-----------+------------+-----------------------------+--------
mytickit1   | sales     | salesid    | integer                     |       1
mytickit1   | sales     | listid     | integer                     |       2
mytickit1   | sales     | sellerid   | integer                     |       3
mytickit1   | sales     | buyerid    | integer                     |       4
mytickit1   | sales     | eventid    | integer                     |       5
mytickit1   | sales     | dateid     | smallint                    |       6
mytickit1   | sales     | qtysold    | smallint                    |       7
mytickit1   | sales     | pricepaid  | numeric(8,2)                |       8
mytickit1   | sales     | commission | numeric(8,2)                |       9
mytickit1   | sales     | saletime   | timestamp without time zone |      10
```

PG_GET_GRANTEE_BY_IAM_ROLE

Retorna todos os usuários e grupos que receberam uma função do IAM especificada.

Sintaxe

```
pg_get_grantee_by_iam_role('iam_role_arn')
```

Argumentos

iam_role_arn

A função do IAM para a qual retornar os usuários e grupos que receberam essa função.

Tipo de retorno

VARCHAR

Observações de uso

A função PG_GET_GRANTEE_BY_BY_IAM_ROLE retorna uma linha para cada usuário ou grupo. Cada linha contém o nome do favorecido, o tipo de favorecido e o privilégio concedido. Os valores possíveis para o tipo de favorecido são p para público, u para o usuário, e g para o grupo.

É preciso ser um superusuário para usar esta função.

Exemplo

O exemplo a seguir indica que a função do IAM `Redshift-S3-Write` é concedida a `group1` e `reg_user1`. Usuários no `group_1` pode especificar a função somente para operações COPY, e o usuário `reg_user1` pode especificar a função somente para executar operações UNLOAD.

```
select pg_get_grantee_by_iam_role('arn:aws:iam::123456789012:role/Redshift-S3-Write');
```

```
  pg_get_grantee_by_iam_role
-----------------------------
 (group_1,g,COPY)
 (reg_user1,u,UNLOAD)
```

O exemplo a seguir da função PG_GET_GRANTEE_BY_IAM_ROLE formata o resultado como uma tabela.

```
select grantee, grantee_type, cmd_type FROM
 pg_get_grantee_by_iam_role('arn:aws:iam::123456789012:role/Redshift-S3-Write')
 res_grantee(grantee text, grantee_type text, cmd_type text) ORDER BY 1,2,3;
```

```
  grantee  | grantee_type | cmd_type
-----------+--------------+----------
 group_1   | g            | COPY
 reg_user1 | u            | UNLOAD
```

PG_GET_IAM_ROLE_BY_USER

Retorna todas as funções do IAM e privilégios de comando concedidos a um usuário.

Sintaxe

```
pg_get_iam_role_by_user('name')
```

Argumentos

name

O nome do usuário para o qual retornar funções do IAM.

Tipo de retorno

VARCHAR

Observações de uso

A função PG_GET_IAM_ROLE_BY_USER retorna uma linha para cada conjunto de funções e privilégios de comando. A linha contém uma lista separada por vírgulas com nome de usuário, função do IAM e comando.

Um valor de `default` no resultado indica que o usuário pode especificar qualquer função disponível para executar o comando exibido.

É preciso ser um superusuário para usar esta função.

Exemplo

O exemplo a seguir indica que o usuário `reg_user1` pode especificar qualquer função do IAM disponível para executar operações COPY. O usuário também pode especificar a função `Redshift-S3-Write` para operações UNLOAD.

```
select pg_get_iam_role_by_user('reg_user1');
```

```
                            pg_get_iam_role_by_user
-------------------------------------------------------------------------------
 (reg_user1,default,COPY)
 (reg_user1,arn:aws:iam::123456789012:role/Redshift-S3-Write,COPY|UNLOAD)
```

O exemplo a seguir da função PG_GET_IAM_ROLE_BY_USER formata o resultado como uma tabela.

```
select username, iam_role, cmd FROM pg_get_iam_role_by_user('reg_user1')
 res_iam_role(username text, iam_role text, cmd text);
```

```
 username  |                    iam_role                     | cmd
-----------+-------------------------------------------------+------
 reg_user1 | default                                         | None
 reg_user1 | arn:aws:iam::123456789012:role/Redshift-S3-Read | COPY
```

PG_GET_LATE_BINDING_VIEW_COLS

Retorna os metadados de coluna para todas as exibições de vinculação tardia no banco de dados. Para obter mais informações, consulte Visualizações de vinculação tardia

Sintaxe

```
pg_get_late_binding_view_cols()
```

Tipo de retorno

VARCHAR

Observações de uso

A função PG_GET_LATE_BINDING_VIEW_COLS retorna uma linha para cada coluna nas exibições de vinculação tardia. A linha contém uma lista, separada por vírgulas, com nome do esquema, nome do relacionamento, nome da coluna, tipo de dados e número da coluna.

Exemplo

O exemplo a seguir retorna os metadados de coluna para todas as exibições de vinculação tardia.

```
select pg_get_late_binding_view_cols();

pg_get_late_binding_view_cols
------------------------------------------------------------
(public,myevent,eventname,"character varying(200)",1)
(public,sales_lbv,salesid,integer,1)
(public,sales_lbv,listid,integer,2)
(public,sales_lbv,sellerid,integer,3)
(public,sales_lbv,buyerid,integer,4)
(public,sales_lbv,eventid,integer,5)
(public,sales_lbv,dateid,smallint,6)
(public,sales_lbv,qtysold,smallint,7)
(public,sales_lbv,pricepaid,"numeric(8,2)",8)
(public,sales_lbv,commission,"numeric(8,2)",9)
(public,sales_lbv,saletime,"timestamp without time zone",10)
(public,event_lbv,eventid,integer,1)
(public,event_lbv,venueid,smallint,2)
(public,event_lbv,catid,smallint,3)
```

```
(public,event_lbv,dateid,smallint,4)
(public,event_lbv,eventname,"character varying(200)",5)
(public,event_lbv,starttime,"timestamp without time zone",6)
```

O exemplo a seguir retorna os metadados de coluna para todas as exibições de vinculação tardia em formato de tabela.

```
select * from pg_get_late_binding_view_cols() cols(view_schema name, view_name name,
 col_name name, col_type varchar, col_num int);
view_schema | view_name | col_name   | col_type                    | col_num
------------+-----------+------------+-----------------------------+--------
public      | sales_lbv | salesid    | integer                     |       1
public      | sales_lbv | listid     | integer                     |       2
public      | sales_lbv | sellerid   | integer                     |       3
public      | sales_lbv | buyerid    | integer                     |       4
public      | sales_lbv | eventid    | integer                     |       5
public      | sales_lbv | dateid     | smallint                    |       6
public      | sales_lbv | qtysold    | smallint                    |       7
public      | sales_lbv | pricepaid  | numeric(8,2)                |       8
public      | sales_lbv | commission | numeric(8,2)                |       9
public      | sales_lbv | saletime   | timestamp without time zone |      10
public      | event_lbv | eventid    | integer                     |       1
public      | event_lbv | venueid    | smallint                    |       2
public      | event_lbv | catid      | smallint                    |       3
public      | event_lbv | dateid     | smallint                    |       4
public      | event_lbv | eventname  | character varying(200)      |       5
public      | event_lbv | starttime  | timestamp without time zone |       6
```

PG_GET_SESSION_ROLES

Retorna os perfis de sessão do usuário conectado no momento. Os perfis de sessão de um usuário são os grupos definidos por um provedor de identidades (IdP) para o usuário conectado. Por exemplo, um provedor de identidades (IdP) como Microsoft Azure Active Directory (Azure AD) verifica a identidade do usuário e fornece os grupos externos dos quais o usuário faz parte durante o processo de login do usuário. Esses grupos externos são transformados em perfis do Amazon Redshift e ficam disponíveis durante a sessão atual. Esses perfis são chamados de perfis de sessão. Um administrador pode conceder privilégios a um perfil de sessão semelhante a outros perfis do Amazon Redshift. Para obter mais informações sobre o uso de perfis, consulte Regras de controle de acesso com base em função (RBAC). Para obter informações sobre como gerenciar identidades com um provedor de identidades (IdP), consulte Federação de um provedor de identidades (IdP) nativo para o Amazon Redshift no Guia de gerenciamento do Amazon Redshift.

Para visualizar os perfis definidos no catálogo do Amazon Redshift, consulte a visualização do sistema SVV_ROLES.

Sintaxe

```
pg_get_session_roles()
```

Tipo de retorno

Um conjunto de linhas que consiste em dois valores. O primeiro valor tem duas partes separadas por dois-pontos (:) que contém um `idp-namespace:role-name`. O `idp-namespace` é o namespace do provedor de identidades (IdP). O `role-name` é o nome do grupo externo no provedor de identidades (IdP). O segundo valor contém um `role-id`, que é o identificador do perfil.

Observações de uso

A função `PG_GET_SESSION_ROLES` retorna uma linha para cada perfil de sessão retornado.

Exemplos

O exemplo a seguir retorna uma linha para cada perfil do IdP do Azure Active Directory. As colunas retornadas são transmitidas para `sess_roles` com colunas `name` e `roleid`. Cada `name` consiste no namespace do Azure Active Directory e em um nome de grupo no Azure Active Directory.

```
SELECT * FROM pg_get_session_roles() AS sess_roles(name name, roleid integer);

name                   roleid
--------------------------------
my_aad:test_group_1   106204
my_aad:test_group_2   106205
my_aad:test_group_3   106206
my_aad:test_group_4   106207
my_aad:test_group_5   106208
```

O exemplo a seguir retorna uma linha para cada grupo do IAM do qual o usuário do IAM conectado no momento é membro. As colunas retornadas são transmitidas para `sess_roles` com colunas `name` e `roleid`. Cada `name` consiste no namespace do IAM e no nome do grupo do IAM.

```
SELECT * FROM pg_get_session_roles() AS sess_roles(name name, roleid integer);

name                   roleid
```

```
--------------------------------
IAM:myGroup           110332
```

PG_LAST_COPY_COUNT

Retorna o número de linhas que foram carregadas pelo último comando COPY executado na sessão atual. PG_LAST_COPY_COUNT é atualizado com o último COPY ID, que é o ID de consulta do último COPY que começou o processo de carregamento, mesmo que o carregamento tenha falhado. O ID da consulta e o ID de COPY são atualizados quando o comando COPY começa o processo de carregamento.

Se COPY falhar devido a um erro de sintaxe ou devido a privilégios insuficientes, o COPY ID não é atualizado e PG_LAST_COPY_COUNT retorna a contagem para o COPY anterior. Se nenhum comando COPY foi executado na sessão atual ou se o último COPY falhou durante o carregamento, PG_LAST_COPY_COUNT retorna 0. Para obter mais informações, consulte PG_LAST_COPY_ID.

Sintaxe

```
pg_last_copy_count()
```

Tipo de retorno

Retorna BIGINT.

Exemplo

A seguinte consulta retorna o número de linhas que foram carregadas pelo comando COPY mais recente na atual sessão.

```
select pg_last_copy_count();

pg_last_copy_count
--------------------
             192497
(1 row)
```

PG_LAST_COPY_ID

Retorna o ID de consulta do comando COPY executado mais recentemente na sessão atual. Se nenhum comando COPY tiver sido executado na sessão atual, PG_LAST_COPY_ID retorna -1.

O valor para PG_LAST_COPY_ID é atualizado quando o comando COPY começa o processo de carregamento. Se COPY falhar devido a dados de carregamento inválidos, o COPY ID é atualizado, portanto você pode usar PG_LAST_COPY_ID quando você consultar a tabela STL_LOAD_ERRORS. Se a transação COPY for revertida, o COPY ID não será atualizado.

O COPY ID não é atualizado se o comando COPY falhar devido a um erro que ocorra antes do começo do processo de carregamento, tal como um erro de sintaxe, erro de acesso, credenciais inválidas ou privilégios insuficientes. O COPY ID não é atualizado se COPY falhar durante a etapa de análise de compactação, que começa após uma conexão com êxito, mas antes do carregamento de dados.

Sintaxe

```
pg_last_copy_id()
```

Tipo de retorno

Retorna um número inteiro.

Exemplo

A seguinte consulta retorna o ID de consulta do comando COPY mais recente na atual sessão.

```
select pg_last_copy_id();

pg_last_copy_id
---------------
          5437
(1 row)
```

A consulta a seguir une as tabelas STL_LOAD_ERRORS e STL_LOADERROR_DETAIL para exibir os detalhes dos erros que ocorreram durante o carregamento mais recente na atual sessão.

```
select d.query, substring(d.filename,14,20),
d.line_number as line,
substring(d.value,1,16) as value,
substring(le.err_reason,1,48) as err_reason
from stl_loaderror_detail d, stl_load_errors le
where d.query = le.query
and d.query = pg_last_copy_id();
```

```
 query |    substring      | line |  value   |                  err_reason
-------+-------------------+------+----------
+-----------------------------------------------
    558| allusers_pipe.txt |  251 | 251      | String contains invalid or unsupported
 UTF8 code
    558| allusers_pipe.txt |  251 | ZRU29FGR | String contains invalid or unsupported
 UTF8 code
    558| allusers_pipe.txt |  251 | Kaitlin  | String contains invalid or unsupported
 UTF8 code
    558| allusers_pipe.txt |  251 | Walter   | String contains invalid or unsupported
 UTF8 code
```

PG_LAST_UNLOAD_ID

Retorna o ID de consulta do comando UNLOAD executado mais recentemente na sessão atual. Se nenhum comando UNLOAD tiver sido executado na sessão atual, o PG_LAST_UNLOAD_ID retornará -1.

O valor para PG_LAST_UNLOAD_ID é atualizado quando o comando UNLOAD começa o processo de carregamento. Se o UNLOAD falhar devido a dados de carga inválidos, o ID de UNLOAD estará atualizado e, portanto, você poderá usar o ID de UNLOAD para uma investigação adicional. Se a transação de UNLOAD for revertida, o ID de UNLOAD não será atualizado.

O ID de UNLOAD não será atualizado se o comando UNLOAD falhar devido a um erro que ocorra antes do começo do processo de carregamento, como um erro de sintaxe, erro de acesso, credenciais inválidas ou privilégios insuficientes.

Sintaxe

```
PG_LAST_UNLOAD_ID()
```

Tipo de retorno

Retorna um número inteiro.

Exemplo

A seguinte consulta retorna o ID de consulta do comando UNLOAD mais recente na atual sessão.

```
select PG_LAST_UNLOAD_ID();

PG_LAST_UNLOAD_ID
```

```
---------------
          5437
(1 row)
```

PG_LAST_QUERY_ID

Retorna o ID de consulta da consulta executada mais recentemente na sessão atual. Se nenhuma consulta tiver sido executada na sessão atual, PG_LAST_QUERY_ID retornará -1. PG_LAST_QUERY_ID não retorna o ID de consulta para consultas que executam exclusivamente no nó líder. Para obter mais informações, consulte Função de apenas nó líder.

Sintaxe

```
pg_last_query_id()
```

Tipo de retorno

Retorna um número inteiro.

Exemplo

A consulta a seguir retorna o ID da última consulta concluída na sessão atual.

```
select pg_last_query_id();
```

A seguir estão os resultados.

```
pg_last_query_id
----------------
           5437
(1 row)
```

A seguinte consulta retorna o ID de consulta e o texto da consulta concluída mais recentemente na sessão atual.

```
select query, trim(querytxt) as sqlquery
from stl_query
where query = pg_last_query_id();
```

A seguir estão os resultados.

```
query | sqlquery
------+--------------------------------------------------
 5437 | select name, loadtime from stl_file_scan where loadtime > 1000000;
(1 rows)
```

PG_LAST_UNLOAD_COUNT

Retorna o número de linhas que foram descarregadas pelo último comando UNLOAD executado na sessão atual. PG_LAST_UNLOAD_COUNT é atualizado com o ID de consulta do último UNLOAD, mesmo se a operação falhou. O ID de consulta é atualizado quando UNLOAD é concluído. Se UNLOAD falhar devido a um erro de sintaxe ou devido a privilégios insuficientes, PG_LAST_UNLOAD_COUNT retorna a contagem para o UNLOAD anterior. Se nenhum comando COPY foi concluído na sessão atual ou se o último UNLOAD falhou durante a operação de descarregamento, PG_LAST_UNLOAD_COUNT retorna 0.

Sintaxe

```
pg_last_unload_count()
```

Tipo de retorno

Retorna BIGINT.

Exemplo

A seguinte consulta retorna o número de linhas que foram descarregadas pelo comando UNLOAD mais recente na atual sessão.

```
select pg_last_unload_count();

pg_last_unload_count
--------------------
             192497
(1 row)
```

Função SLICE_NUM

Retorna um número inteiro correspondente ao número de fatia no cluster onde os dados para uma linha estão localizados. SLICE_NUM não aceita parâmetros.

Sintaxe

```
SLICE_NUM()
```

Tipo de retorno

A função SLICE_NUM retorna um número inteiro.

Exemplos

O seguinte exemplo mostra quais fatias contêm dados para as primeiras dez linhas de EVENT na tabela EVENTS:

```
select distinct eventid, slice_num() from event order by eventid limit 10;

 eventid | slice_num
---------+-----------
       1 |         1
       2 |         2
       3 |         3
       4 |         0
       5 |         1
       6 |         2
       7 |         3
       8 |         0
       9 |         1
      10 |         2
(10 rows)
```

O exemplo a seguir retorna um código (10000) para mostrar que uma consulta sem a instrução FROM executa no nó líder:

```
select slice_num();
slice_num
-----------
10000
(1 row)
```

USER

Sinônimo de CURRENT_USER. Consulte CURRENT_USER.

ROLE_IS_MEMBER_OF

Retorna true (verdadeiro) se a função for membro de outra função. Os superusuários podem verificar a associação de todas as funções. Usuários regulares que têm a permissão ACCESS SYSTEM TABLE podem verificar a associação de todos os usuários. Caso contrário, os usuários comuns podem verificar somente as funções às quais têm acesso. O Amazon Redshift gera erro se as funções fornecidas não existirem ou se o usuário atual não tiver acesso à função.

Considerações sobre compartilhamento de dados

Quando um cluster de consumidor consulta um objeto compartilhado que faz referência a essa função, como uma visualização, uma política de RLS ou uma política de DDM, a função avalia usando o contexto de segurança do cluster de consumidor. Os usuários locais, os perfis e as associações de grupos do consumidor determinam o resultado, não aqueles definidos no cluster de produtor. Se você pretende aplicar o mesmo contexto de permissões implementado no produtor, verifique se os nomes de perfis correspondentes, os nomes de grupos e as associações de usuários existem no cluster consumidor e correspondem aos do produtor.

Sintaxe

```
role_is_member_of( role_name,  granted_role_name)
```

Argumentos

role_name

O nome da função.

granted_role_name

O nome da função concedida.

Tipo de retorno

Retorna um BOOLEAN.

Exemplo

A consulta a seguir confirma que a função não é membro de role1 nem de role2.

```
SELECT role_is_member_of('role1', 'role2');
```

```
 role_is_member_of
-------------------
             False
```

USER_IS_MEMBER_OF

> **Important**
>
> A partir de 16 de fevereiro de 2026, o Amazon Redshift não permitirá mais o uso de `user_is_member_of` e de funções relacionadas que acessem informações do consumidor, como usuário, perfil ou grupo, por meio da unidade de compartilhamento de dados.

Retorna true (verdadeiro) se o usuário for um membro de uma função ou grupo. Os superusuários podem verificar a associação de todos os usuários. Usuários regulares que sejam membros da função sys:secadmin ou sys:superuser podem verificar a associação de todos os usuários. Caso contrário, usuários regulares podem verificar somente a si mesmos. O Amazon Redshift emite um erro se as identidades fornecidas não existirem ou se o usuário atual não tiver acesso à função.

Considerações sobre compartilhamento de dados

Quando um cluster de consumidor consulta um objeto compartilhado que faz referência a essa função, como uma visualização, uma política de RLS ou uma política de DDM, a função avalia usando o contexto de segurança do cluster de consumidor. Os usuários locais, os perfis e as associações de grupos do consumidor determinam o resultado, não aqueles definidos no cluster de produtor. Se você pretende aplicar o mesmo contexto de permissões implementado no produtor, verifique se os nomes de perfis correspondentes, os nomes de grupos e as associações de usuários existem no cluster consumidor e correspondem aos do produtor.

Sintaxe

```
user_is_member_of( user_name,  role_name | group_name)
```

Argumentos

user_name

O nome do usuário.

role_name

O nome da função.

nome_grupo

O nome do grupo.

Tipo de retorno

Retorna um BOOLEAN.

Exemplo

A consulta a seguir confirma que o usuário não é membro de role1.

```
SELECT user_is_member_of('reguser', 'role1');

 user_is_member_of
-------------------
           False
```

VERSION

A função VERSION retorna detalhes sobre a versão instalada atualmente, com informações específicas da versão do Amazon Redshift no final.

> **Note**
>
> Essa é uma função do nó de liderança. Essa função retorna um erro se fizer referência a uma tabela criada por usuário, a uma tabela de sistema STL ou STV ou a uma exibição de sistema SVV ou SVL.

Sintaxe

```
VERSION()
```

Tipo de retorno

Retorna uma string CHAR ou VARCHAR.

Exemplos

O exemplo a seguir mostra as informações de versão do cluster atual:

```
select version();
```

```
version
----------------------------------------------------------------------------------------------------------

PostgreSQL 8.0.2 on i686-pc-linux-gnu, compiled by GCC gcc (GCC) 3.4.2 20041017 (Red
Hat 3.4.2-6.fc3), Redshift 1.0.12103
```

Onde `1.0.12103` é o número da versão do cluster.

> **Note**
>
> Para forçar a atualização do seu cluster para uma versão mais recente, ajuste sua janela de manutenção.

Palavras reservadas

A seguir, uma lista de palavras reservadas do Amazon Redshift. Você pode usar as palavras reservadas com identificadores delimitados (aspas duplas).

> **Note**
>
> Embora START e CONNECT não sejam palavras reservadas, use identificadores delimitados ou AS se estiver usando START e CONNECT como aliases de tabela em sua consulta para evitar falhas no runtime.

Para ter mais informações, consulte Nomes e identificadores.

```
AES128
AES256
ALL
ALLOWOVERWRITE
ANALYSE
ANALYZE
```

```
AND
ANY
ARRAY
AS
ASC
AUTHORIZATION
AZ64
BACKUP
BETWEEN
BINARY
BLANKSASNULL
BOTH
BYTEDICT
BZIP2
CASE
CAST
CHECK
COLLATE
COLUMN
CONSTRAINT
CREATE
CREDENTIALS
CROSS
CURRENT_DATE
CURRENT_TIME
CURRENT_TIMESTAMP
CURRENT_USER
CURRENT_USER_ID
DEFAULT
DEFERRABLE
DEFLATE
DEFRAG
DELTA
DELTA32K
DESC
DISABLE
DISTINCT
DO
ELSE
EMPTYASNULL
ENABLE
ENCODE
ENCRYPT
ENCRYPTION
```

```
END
EXCEPT
EXPLICIT
FALSE
FOR
FOREIGN
FREEZE
FROM
FULL
GLOBALDICT256
GLOBALDICT64K
GRANT
GROUP
GZIP
HAVING
IDENTITY
IGNORE
ILIKE
IN
INITIALLY
INNER
INTERSECT
INTERVAL
INTO
IS
ISNULL
JOIN
LEADING
LEFT
LIKE
LIMIT
LOCALTIME
LOCALTIMESTAMP
LUN
LUNS
LZO
LZOP
MINUS
MOSTLY16
MOSTLY32
MOSTLY8
NATURAL
NEW
NOT
```

```
NOTNULL
NULL
NULLS
OFF
OFFLINE
OFFSET
OID
OLD
ON
ONLY
OPEN
OR
ORDER
OUTER
OVERLAPS
PARALLEL
PARTITION
PERCENT
PERMISSIONS
PIVOT
PLACING
PRIMARY
RAW
READRATIO
RECOVER
REFERENCES
REJECTLOG
RESORT
RESPECT
RESTORE
RIGHT
SELECT
SESSION_USER
SIMILAR
SNAPSHOT
SOME
SYSDATE
SYSTEM
TABLE
TAG
TDES
TEXT255
TEXT32K
THEN
```

```
TIMESTAMP
TO
TOP
TRAILING
TRUE
TRUNCATECOLUMNS
UNION
UNIQUE
UNNEST
UNPIVOT
USER
USING
VERBOSE
WALLET
WHEN
WHERE
WITH
WITHOUT
```

Referência de visualizações e tabelas do sistema

O Amazon Redshift tem muitas tabelas de sistema e visualizações que contêm informações sobre como o sistema está funcionando. Você pode consultar essas tabelas de sistema e visualizações da mesma maneira como consultaria qualquer outra tabela de banco de dados. Esta seção mostra algumas consultas de tabela de sistema de exemplo e explica:

- Como tipos diferentes de tabelas de sistema e visualizações são gerados
- Quais tipos de informações você pode obter dessas tabelas
- Como unir tabelas de sistema do Amazon Redshift a tabelas de catálogo
- Como gerenciar o crescimento de arquivos de log da tabela de sistema

Algumas tabelas de sistema somente podem ser usadas pela equipe da AWS para fins de diagnóstico. As seções a seguir abordam as tabelas de sistema que podem ser consultadas em busca de informações úteis por administradores de sistema ou outros usuários do banco de dados.

> **Note**
>
> As tabelas de sistema não estão incluídas em backups de cluster automatizados ou manuais (snapshots). As visualizações do sistema STL retêm sete dias do histórico de log. Reter os logs não requer nenhuma ação do cliente, mas se você quiser reter os dados de log por mais de sete dias, será necessário copiá-los periodicamente para outras tabelas ou descarregá-los no Amazon S3.

Tópicos

- Tipos de tabelas e visualizações de sistema
- Visibilidade de dados em tabelas e visualizações de sistema
- Migrar consultas somente provisionadas para consultas de visualização de monitoramento de SYS
- Melhoria do rastreamento do identificador de consultas usando as exbições de monitoramento SYS
- Consulta de tabela do sistema, processo e IDs de sessão
- Visualizações SVV de metadados

- Visualizações de monitoramento de SYS
- Mapeamento de visualizações do sistema para migrar para visualizações de monitoramento de SYS
- Monitoramento do sistema (somente provisionado)
- Tabelas de catálogo do sistema

Tipos de tabelas e visualizações de sistema

Há vários tipos de tabelas e visões do sistema:

- As visualizações SVV contêm informações sobre objetos de banco de dados com referências a tabelas STV transitórias.
- As visualizações SYS são usadas para monitorar o uso de consultas e workload para clusters e grupos de trabalho sem servidor.
- As tabelas STL são geradas a partir de logs que foram mantidos em disco para apresentar um histórico do sistema.
- Essas tabelas de STV são tabelas virtuais do sistema que contêm snapshots dos dados atuais do sistema. Elas se baseiam em dados transitórios na memória e não são mantidas em logs baseados em disco ou em tabelas regulares.
- As visões SVCS fornecem detalhes sobre consultas nos clusters principal e de escalabilidade da simultaneidade.
- As visualizações SVL fornecem detalhes sobre consultas nos clusters principais.

As tabelas de sistema e as visualizações não usam o mesmo modelo de consistência de tabelas regulares. É importante estar ciente desse problema ao consultá-las, especialmente para tabelas STV e visualizações SVV. Por exemplo, dada uma tabela t1 regular com uma coluna c1, você esperaria que a seguinte consulta não retornasse linhas:

```
select * from t1
where c1 > (select max(c1) from t1)
```

Porém, a seguinte consulta em relação a uma tabela de sistema também pode retornar linhas:

```
select * from stv_exec_state
```

```
where currenttime > (select max(currenttime) from stv_exec_state)
```

O motivo dessa consulta poder retornar linhas é que currenttime é temporário e as duas referências na consulta talvez não retornem o mesmo valor quando avaliadas.

Por outro lado, a seguinte consulta também pode não retornar linhas:

```
select * from stv_exec_state
where currenttime = (select max(currenttime) from stv_exec_state)
```

Visibilidade de dados em tabelas e visualizações de sistema

> **Note**
>
> O Amazon Redshift mascara automaticamente determinadas colunas das tabelas do sistema ao registrar informações sobre consultas feitas às visualizações do catálogo de dados para evitar a exposição de metadados confidenciais. Para obter mais informações, consulte Registros em log seguros no Guia de gerenciamento do Amazon Redshift.

Existem duas classes de visibilidade para dados em tabelas e exibições do sistema: visível para usuários e visível para superusuários.

Somente usuários com privilégios de superusuário podem ver os dados dessas tabelas que estejam na categoria visível para superusuários. Os usuários regulares podem ver dados em tabelas visíveis para usuários. Para oferecer a um usuário regular acesso às tabelas visíveis aos superusuários, conceda o privilégio SELECT nessa tabela ao usuário regular. Para obter mais informações, consulte GRANT.

Por padrão, na maioria das tabelas visíveis para usuários, as linhas geradas por um outro usuário são invisíveis para um usuário regular. Se um usuário regular receber SYSLOG ACCESS UNRESTRICTED, ele poderá ver todas as linhas em tabelas visíveis aos usuários, incluindo as linhas geradas por outros usuários. Para ter mais informações, consulte ALTER USER ou CREATE USER. Todas as linhas em SVV_TRANSACTIONS são visíveis a todos os usuários. Para obter informações sobre visibilidade de dados, consulte o artigo AWS re:Post Como posso permitir que usuários regulares do banco de dados Amazon Redshift visualizem dados em tabelas do sistema de outros usuários do cluster? da base de conhecimento.

Para visualizações de metadados, o Amazon Redshift não permite visibilidade para usuários que recebem SYSLOG ACCESS UNRESTRICTED.

> **Note**
>
> Fornecer acesso ilimitado para um usuário às tabelas de sistema é o mesmo que dar ao usuário visibilidade para os dados gerados por outros usuários. Por exemplo, STL_QUERY e STL_QUERY_TEXT contêm texto completo de instruções INSERT, UPDATE e DELETE, e podem conter dados confidenciais gerados pelos usuários.

Um superusuário pode ver todas as linhas em todas as tabelas. Para oferecer acesso a um usuário regular às tabelas visíveis para superusuários, GRANTselecione o privilégio nessa tabela para o usuário regular.

Filtrar consultas geradas pelo sistema

As tabelas e visualizações do sistema relacionadas à consulta, como SVL_QUERY_SUMMARY, SVL_QLOG e outras, geralmente contêm um grande número de instruções geradas automaticamente que o Amazon Redshift usa para monitorar o status do banco de dados. Essas consultas geradas pelo sistema são visíveis para um superusuário, mas raramente são úteis. Para filtrá-las ao selecionar uma tabela ou visualização de sistema que use a coluna `userid`, adicione a condição `userid > 1` à cláusula WHERE. Por exemplo:

```
select * from svl_query_summary where userid > 1
```

Migrar consultas somente provisionadas para consultas de visualização de monitoramento de SYS

Migrar clusters provisionados para o Amazon Redshift sem servidor

Se você estiver migrando um cluster provisionado para o Amazon Redshift sem servidor, poderá ter consultas usando as visualizações do sistema a seguir, que só armazenam dados de clusters provisionados.

- Todas as visualizações STL
- Todas as visualizações STV

- Todas as visualizações SVCS
- Todas as visualizações SVL
- Algumas visualizações SVV
  - Para ter uma lista completa de visualizações SVV não aceitas no Amazon Redshift sem servidor, consulte a lista na parte inferior de Monitorar consultas e workloads com o Amazon Redshift Serverless no Guia de gerenciamento do Amazon Redshift.

Para continuar usando suas consultas, redefina-as para utilizar colunas definidas nas visualizações de monitoramento de SYS que correspondam às colunas em suas visualizações somente provisionadas. Para ver a relação de mapeamento entre as visualizações somente provisionadas e as visualizações de monitoramento de SYS, acesse Mapeamento de visualizações do sistema para migrar para visualizações de monitoramento de SYS.

Atualizar consultas enquanto estiver em um cluster provisionado

Se você não estiver migrando para o Amazon Redshift sem servidor, ainda assim deveria atualizar as consultas existentes. As visualizações de monitoramento de SYS foram projetadas para serem fáceis de usar e diminuir a complexidade, fornecendo um conjunto completo de métricas para contribuir para a eficácia de monitoramento e solução de problemas. Usando visualizações SYS como SYS_QUERY_HISTORY e SYS_QUERY_DETAIL, que consolidam as informações de várias visualizações somente provisionadas, você pode simplificar suas consultas.

Melhoria do rastreamento do identificador de consultas usando as exbições de monitoramento SYS

As exibições de monitoramento SYS, como SYS_QUERY_HISTORY e SYS_QUERY_DETAIL, contêm a coluna query_id, que contém o identificador das consultas dos usuários. Da mesma forma, as exibições somente provisionadas, como STL_QUERY e SVL_QLOG, contêm a coluna de consulta, que também contém os identificadores de consulta. No entanto, os identificadores de consulta registrados nas exibições de sistema SYS são diferentes daqueles registrados nas exibições somente provisionadas.

A diferença entre os valores da coluna query_id das exibições SYS e os valores da coluna de consulta das exibições somente provisionadas é a seguinte:

- Em exibições SYS, a coluna query_id registra as consultas enviadas pelo usuário no formato original. O otimizador do Amazon Redshift pode dividi-las em consultas secundárias tendo em

vista um desempenho melhor, mas uma única consulta executada por você ainda terá apenas uma única linha em SYS_QUERY_HISTORY. Se quiser ver as consultas filho individuais, você poderá encontrá-las em SYS_QUERY_DETAIL.

- Em exibições somente provisionadas, a coluna de consulta registra consultas no nível da consulta filho. Se o otimizador do Amazon Redshift reescrever a consulta original em várias consultas filho, haverá várias linhas em STL_QUERY com valores identificadores de consulta diferentes para uma única consulta executada por você.

Ao migrar as consultas de monitoramento e diagnóstico de exibições somente provisionadas para exibições SYS, considere essa diferença e edite as consultas de acordo. Para obter mais informações sobre como o Amazon Redshift processa as consultas, consulte Planejamento de consulta e fluxo de trabalho de execução.

Exemplo

Para ver um exemplo de como o Amazon Redshift registra consultas de maneira diferente em visualizações somente provisionadas e de monitoramento de SYS, consulte o exemplo de consulta a seguir. Esta é a consulta escrita como você a executaria no Amazon Redshift.

```
SELECT
  s_name
  , COUNT(*) AS numwait
FROM
  supplier,
  lineitem l1,
  orders,
  nation
WHERE    s_suppkey = l1.l_suppkey
         AND o_orderkey = l1.l_orderkey
         AND o_orderstatus = 'F'
         AND l1.l_receiptdate > l1.l_commitdate
         AND EXISTS (SELECT
                       *
                     FROM
                       lineitem l2
                     WHERE  l2.l_orderkey = l1.l_orderkey
                            AND l2.l_suppkey <> l1.l_suppkey )
         AND NOT EXISTS (SELECT
                           *
                         FROM
```

```
                            lineitem l3
                           WHERE  l3.l_orderkey = l1.l_orderkey
                                  AND l3.l_suppkey <> l1.l_suppkey
                                  AND l3.l_receiptdate > l3.l_commitdate )
         AND s_nationkey = n_nationkey
         AND n_name = 'UNITED STATES'
GROUP BY
  s_name
ORDER BY
  numwait DESC
  , s_name LIMIT 100;
```

Nos bastidores, o otimizador de consultas do Amazon Redshift reescreve a consulta enviada pelo usuário acima em cinco consultas filho.

A primeira consulta filho cria uma tabela temporária para materializar uma subconsulta.

```
CREATE TEMP TABLE volt_tt_606590308b512(l_orderkey
                                       , l_suppkey
                                       , s_name   ) AS SELECT
                                                     l1.l_orderkey
                                                     , l1.l_suppkey
                                                     , public.supplier.s_name
                                                   FROM
                                                     public.lineitem AS l1,
                                                     public.nation,
                                                     public.orders,
                                                     public.supplier
                                                   WHERE  l1.l_commitdate <
 l1.l_receiptdate
                                                          AND l1.l_orderkey =
 public.orders.o_orderkey
                                                          AND l1.l_suppkey =
 public.supplier.s_suppkey
                                                          AND public.nation.n_name
 = 'UNITED STATES'::CHAR(8)
                                                          AND
 public.nation.n_nationkey = public.supplier.s_nationkey
                                                          AND
 public.orders.o_orderstatus = 'F'::CHAR(1);
```

A segunda consulta filho coleta estatísticas da tabela temporária.

```
padb_fetch_sample: select count(*) from volt_tt_606590308b512;
```

A terceira consulta filho cria outra tabela temporária para materializar outra subconsulta, referenciando a tabela temporária criada acima.

```
CREATE TEMP TABLE volt_tt_606590308c2ef(l_orderkey
                                       , l_suppkey) AS (SELECT

 volt_tt_606590308b512.l_orderkey
                                                          ,
 volt_tt_606590308b512.l_suppkey
                                                        FROM
                                                          public.lineitem AS l2,
                                                          volt_tt_606590308b512
                                                        WHERE  l2.l_suppkey <>
 volt_tt_606590308b512.l_suppkey
                                                               AND l2.l_orderkey =
 volt_tt_606590308b512.l_orderkey)
                                                               EXCEPT distinct (SELECT
 volt_tt_606590308b512.l_orderkey, volt_tt_606590308b512.l_suppkey
                                                               FROM public.lineitem AS
 l3, volt_tt_606590308b512
                                                               WHERE l3.l_commitdate <
 l3.l_receiptdate
                                                                 AND l3.l_suppkey <>
 volt_tt_606590308b512.l_suppkey
                                                                 AND l3.l_orderkey =
 volt_tt_606590308b512.l_orderkey);
```

A quarta consulta filho coleta novamente estatísticas da tabela temporária.

```
padb_fetch_sample: select count(*) from volt_tt_606590308c2ef
```

A última consulta filho usa as tabelas temporárias criadas acima para gerar a saída.

```
SELECT
  volt_tt_606590308b512.s_name AS s_name
  , COUNT(*) AS numwait
FROM
  volt_tt_606590308b512,
  volt_tt_606590308c2ef
WHERE    volt_tt_606590308b512.l_orderkey = volt_tt_606590308c2ef.l_orderkey
```

```
         AND volt_tt_606590308b512.l_suppkey = volt_tt_606590308c2ef.l_suppkey
GROUP BY
  1
ORDER BY
  2 DESC
  , 1 ASC LIMIT 100;
```

Na exibição do sistema somente provisionada STL_QUERY, o Amazon Redshift registra cinco linhas no nível da consulta filho, da seguinte maneira:

```
SELECT userid, xid, pid, query, querytxt::varchar(100);
FROM stl_query
WHERE xid = 48237350
ORDER BY xid, starttime;

 userid |   xid    |    pid     |  query   |
    querytxt
--------+----------+------------+----------
+-----------------------------------------------------------------------------------------------------
    101 | 48237350 | 1073840810 | 12058151 | CREATE TEMP TABLE
 volt_tt_606590308b512(l_orderkey, l_suppkey, s_name) AS SELECT l1.l_orderkey, l1.l
    101 | 48237350 | 1073840810 | 12058152 | padb_fetch_sample: select count(*) from
 volt_tt_606590308b512
    101 | 48237350 | 1073840810 | 12058156 | CREATE TEMP TABLE
 volt_tt_606590308c2ef(l_orderkey, l_suppkey) AS (SELECT volt_tt_606590308b512.l_or
    101 | 48237350 | 1073840810 | 12058168 | padb_fetch_sample: select count(*) from
 volt_tt_606590308c2ef
    101 | 48237350 | 1073840810 | 12058170 | SELECT s_name , COUNT(*) AS numwait FROM
 supplier, lineitem l1, orders, nation WHERE s_suppkey = l1.
(5 rows)
```

Na exibição de monitoramento SYS SYS_QUERY_HISTORY, o Amazon Redshift registra a consulta da seguinte maneira:

```
SELECT user_id, transaction_id, session_id, query_id, query_text::varchar(100)
FROM sys_query_history
WHERE transaction_id = 48237350
ORDER BY start_time;

 user_id | transaction_id | session_id | query_id |
          query_text
---------+----------------+------------+----------
+-----------------------------------------------------------------------------------------------------
```

```
     101 |       48237350 | 1073840810 | 12058149 | SELECT s_name , COUNT(*) AS numwait
 FROM supplier, lineitem l1, orders, nation WHERE s_suppkey = l1.
```

Em SYS_QUERY_DETAIL, você pode encontrar detalhes no nível da consulta filho usando o valor query_id de SYS_QUERY_HISTORY. A coluna child_query_sequence mostra a ordem na qual as consultas filho são executadas. Para obter mais informações sobre as colunas em SYS_QUERY_DETAIL, consulte SYS_QUERY_DETAIL.

```
select user_id,
       query_id,
       child_query_sequence,
       stream_id,
       segment_id,
       step_id,
       start_time,
       end_time,
       duration,
       blocks_read,
       blocks_write,
       local_read_io,
       remote_read_io,
       data_skewness,
       time_skewness,
       is_active,
       spilled_block_local_disk,
       spilled_block_remote_disk
from sys_query_detail
where query_id = 12058149
      and step_id = -1
order by query_id,
         child_query_sequence,
         stream_id,
         segment_id,
         step_id;

 user_id | query_id | child_query_sequence | stream_id | segment_id | step_id |
       start_time         |          end_time          | duration | blocks_read |
 blocks_write | local_read_io | remote_read_io | data_skewness | time_skewness |
 is_active | spilled_block_local_disk | spilled_block_remote_disk
---------+----------+----------------------+-----------+------------+---------
+---------------------------+---------------------------+----------+-------------
+--------------+--------------+---------------+--------------+--------------
+-----------+--------------------------+--------------------------
```

```
    101 | 12058149 |                    1 |         0 |          0 |      -1 |
2023-09-27 15:40:38.512415 | 2023-09-27 15:40:38.533333 |    20918 |           0 |
        0 |             0 |              0 |             0 |           44 | f
|                         0 |                           0
    101 | 12058149 |                    1 |         1 |          1 |      -1 |
2023-09-27 15:40:39.931437 | 2023-09-27 15:40:39.972826 |    41389 |          12 |
        0 |            12 |              0 |             0 |           77 | f
|                         0 |                           0
    101 | 12058149 |                    1 |         2 |          2 |      -1 |
2023-09-27 15:40:40.584412 | 2023-09-27 15:40:40.613982 |    29570 |          32 |
        0 |            32 |              0 |             0 |           25 | f
|                         0 |                           0
    101 | 12058149 |                    1 |         2 |          3 |      -1 |
2023-09-27 15:40:40.582038 | 2023-09-27 15:40:40.615758 |    33720 |           0 |
        0 |             0 |              0 |             0 |            1 | f
|                         0 |                           0
    101 | 12058149 |                    1 |         3 |          4 |      -1 |
2023-09-27 15:40:46.668766 | 2023-09-27 15:40:46.705456 |    36690 |          24 |
        0 |            15 |              0 |             0 |           17 | f
|                         0 |                           0
    101 | 12058149 |                    1 |         4 |          5 |      -1 |
2023-09-27 15:40:46.707209 | 2023-09-27 15:40:46.709176 |     1967 |           0 |
        0 |             0 |              0 |             0 |           18 | f
|                         0 |                           0
    101 | 12058149 |                    1 |         4 |          6 |      -1 |
2023-09-27 15:40:46.70656  | 2023-09-27 15:40:46.71289  |     6330 |           0 |
        0 |             0 |              0 |             0 |            0 | f
|                         0 |                           0
    101 | 12058149 |                    1 |         5 |          7 |      -1 |
2023-09-27 15:40:46.71405  | 2023-09-27 15:40:46.714343 |      293 |           0 |
        0 |             0 |              0 |             0 |            0 | f
|                         0 |                           0
    101 | 12058149 |                    2 |         0 |          0 |      -1 |
2023-09-27 15:40:52.083907 | 2023-09-27 15:40:52.087854 |     3947 |           0 |
        0 |             0 |              0 |             0 |           35 | f
|                         0 |                           0
    101 | 12058149 |                    2 |         1 |          1 |      -1 |
2023-09-27 15:40:52.089632 | 2023-09-27 15:40:52.091129 |     1497 |           0 |
        0 |             0 |              0 |             0 |           11 | f
|                         0 |                           0
    101 | 12058149 |                    2 |         1 |          2 |      -1 |
2023-09-27 15:40:52.089008 | 2023-09-27 15:40:52.091306 |     2298 |           0 |
        0 |             0 |              0 |             0 |            0 | f
|                         0 |                           0
```

```
    101 | 12058149 |                    3 |         0 |          0 |      -1 |
2023-09-27 15:40:56.882013 | 2023-09-27 15:40:56.897282 |    15269 |           0 |
        0 |             0 |              0 |             0 |           29 | f
|                         0 |                           0
    101 | 12058149 |                    3 |         1 |          1 |      -1 |
2023-09-27 15:40:59.718554 | 2023-09-27 15:40:59.722789 |     4235 |           0 |
        0 |             0 |              0 |             0 |           13 | f
|                         0 |                           0
    101 | 12058149 |                    3 |         2 |          2 |      -1 |
2023-09-27 15:40:59.800382 | 2023-09-27 15:40:59.807388 |     7006 |           0 |
        0 |             0 |              0 |             0 |           58 | f
|                         0 |                           0
    101 | 12058149 |                    3 |         3 |          3 |      -1 |
2023-09-27 15:41:06.488685 | 2023-09-27 15:41:06.493825 |     5140 |           0 |
        0 |             0 |              0 |             0 |           56 | f
|                         0 |                           0
    101 | 12058149 |                    3 |         3 |          4 |      -1 |
2023-09-27 15:41:06.486206 | 2023-09-27 15:41:06.497756 |    11550 |           0 |
        0 |             0 |              0 |             0 |            2 | f
|                         0 |                           0
    101 | 12058149 |                    3 |         4 |          5 |      -1 |
2023-09-27 15:41:06.499201 | 2023-09-27 15:41:06.500851 |     1650 |           0 |
        0 |             0 |              0 |             0 |           15 | f
|                         0 |                           0
    101 | 12058149 |                    3 |         4 |          6 |      -1 |
2023-09-27 15:41:06.498609 | 2023-09-27 15:41:06.500949 |     2340 |           0 |
        0 |             0 |              0 |             0 |            0 | f
|                         0 |                           0
    101 | 12058149 |                    3 |         5 |          7 |      -1 |
2023-09-27 15:41:06.502945 | 2023-09-27 15:41:06.503282 |      337 |           0 |
        0 |             0 |              0 |             0 |            0 | f
|                         0 |                           0
    101 | 12058149 |                    4 |         0 |          0 |      -1 |
2023-09-27 15:41:06.62899  | 2023-09-27 15:41:06.631452 |     2462 |           0 |
        0 |             0 |              0 |             0 |           22 | f
|                         0 |                           0
    101 | 12058149 |                    4 |         1 |          1 |      -1 |
2023-09-27 15:41:06.632313 | 2023-09-27 15:41:06.63391  |     1597 |           0 |
        0 |             0 |              0 |             0 |           20 | f
|                         0 |                           0
    101 | 12058149 |                    4 |         1 |          2 |      -1 |
2023-09-27 15:41:06.631726 | 2023-09-27 15:41:06.633813 |     2087 |           0 |
        0 |             0 |              0 |             0 |            0 | f
|                         0 |                           0
```

```
    101 | 12058149 |                    5 |         0 |          0 |      -1 |
 2023-09-27 15:41:12.571974 | 2023-09-27 15:41:12.584234 |    12260 |           0 |
         0 |             0 |              0 |             0 |           39 | f
 |                         0 |                           0
    101 | 12058149 |                    5 |         0 |          1 |      -1 |
 2023-09-27 15:41:12.569815 | 2023-09-27 15:41:12.585391 |    15576 |           0 |
         0 |             0 |              0 |             0 |            4 | f
 |                         0 |                           0
    101 | 12058149 |                    5 |         1 |          2 |      -1 |
 2023-09-27 15:41:13.758513 | 2023-09-27 15:41:13.76401  |     5497 |           0 |
         0 |             0 |              0 |             0 |           39 | f
 |                         0 |                           0
    101 | 12058149 |                    5 |         1 |          3 |      -1 |
 2023-09-27 15:41:13.749    | 2023-09-27 15:41:13.772987 |    23987 |           0 |
         0 |             0 |              0 |             0 |           32 | f
 |                         0 |                           0
    101 | 12058149 |                    5 |         2 |          4 |      -1 |
 2023-09-27 15:41:13.799526 | 2023-09-27 15:41:13.813506 |    13980 |           0 |
         0 |             0 |              0 |             0 |           62 | f
 |                         0 |                           0
    101 | 12058149 |                    5 |         2 |          5 |      -1 |
 2023-09-27 15:41:13.798823 | 2023-09-27 15:41:13.813651 |    14828 |           0 |
         0 |             0 |              0 |             0 |            0 | f
 |                         0 |                           0
(28 rows)
```

Consulta de tabela do sistema, processo e IDs de sessão

Ao analisar os IDs de consulta, de processo e de sessão que aparecem nas tabelas do sistema, esteja ciente do seguinte:

- O valor do ID de consulta (em colunas, como `query_id` e `query`) pode ser reutilizado ao longo do tempo.
- O valor do ID de processo ou de sessão (em colunas, como `process_id`, `pid` e `session_id`) pode ser reutilizado ao longo do tempo.
- O valor do ID de transação (em colunas, como `transaction_id` e `xid`) é exclusivo.

Visualizações SVV de metadados

As visualizações SVV são visualizações do sistema no Amazon Redshift que contêm informações sobre objetos do banco de dados. Embora armazenem informações como permissões de usuário ou nomes de tabelas, não se destinam a unir-se a relações criadas pelo usuário.

> **Note**
>
> O Amazon Redshift reporta um WARNING, não um ERROR, se uma resposta do banco de dados falha por qualquer motivo. O Amazon Redshift não envia mensagens de ERROR quando você está consultando objetos em uma unidade de compartilhamento de dados.

Tópicos

- SVV_ACTIVE_CURSORS
- SVV_ALL_COLUMNS
- SVV_ALL_SCHEMAS
- SVV_ALL_TABLES
- SVV_ALTER_TABLE_RECOMMENDATIONS
- SVV_ATTACHED_MASKING_POLICY
- SVV_COLUMNS
- SVV_COLUMN_PRIVILEGES
- SVV_COPY_JOB_INTEGRATIONS
- SVV_DATABASE_PRIVILEGES
- SVV_DATASHARE_PRIVILEGES
- SVV_DATASHARES
- SVV_DATASHARE_CONSUMERS
- SVV_DATASHARE_OBJECTS
- SVV_DEFAULT_PRIVILEGES
- SVV_DISKUSAGE
- SVV_EXTERNAL_COLUMNS
- SVV_EXTERNAL_DATABASES
- SVV_EXTERNAL_PARTITIONS

- SVV_EXTERNAL_SCHEMAS
- SVV_EXTERNAL_TABLES
- SVV_FUNCTION_PRIVILEGES
- SVV_GEOGRAPHY_COLUMNS
- SVV_GEOMETRY_COLUMNS
- SVV_IAM_PRIVILEGES
- SVV_IDENTITY_PROVIDERS
- SVV_INTEGRATION
- SVV_INTEGRATION_TABLE_MAPPING
- SVV_INTEGRATION_TABLE_STATE
- SVV_INTERLEAVED_COLUMNS
- SVV_LANGUAGE_PRIVILEGES
- SVV_MASKING_POLICY
- SVV_ML_MODEL_INFO
- SVV_ML_MODEL_PRIVILEGES
- SVV_MV_DEPENDENCY
- SVV_MV_INFO
- SVV_QUERY_INFLIGHT
- SVV_QUERY_STATE
- SVV_REDSHIFT_COLUMNS
- SVV_REDSHIFT_DATABASES
- SVV_REDSHIFT_FUNCTIONS
- SVV_REDSHIFT_SCHEMA_QUOTA
- SVV_REDSHIFT_SCHEMAS
- SVV_REDSHIFT_TABLES
- SVV_RELATION_PRIVILEGES
- SVV_RLS_APPLIED_POLICY
- SVV_RLS_ATTACHED_POLICY
- SVV_RLS_POLICY
- SVV_RLS_RELATION

- SVV_ROLE_GRANTS
- SVV_ROLES
- SVV_SCHEMA_PRIVILEGES
- SVV_SCHEMA_QUOTA_STATE
- SVV_SYSTEM_PRIVILEGES
- SVV_TABLE_INFO
- SVV_TABLES
- SVV_TRANSACTIONS
- SVV_USER_GRANTS
- SVV_USER_INFO
- SVV_VACUUM_PROGRESS
- SVV_VACUUM_SUMMARY

SVV_ACTIVE_CURSORS

A tabela SVV_ACTIVE_CURSORS exibe os detalhes dos cursores que estão abertos. Para obter mais informações, consulte DECLARE.

SVV_ACTIVE_CURSORS permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema. Um usuário só pode visualizar cursores abertos por ele. Os superusuários podem visualizar todos os cursores.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	integer	O ID do usuário que criou o cursor.
nome_cursor	varchar(128)	O nome do cursor.
transaction_id	bigint(128)	O ID da transação.
session_id	integer	O ID do processo com o cursor ativo.

Nome da coluna	Tipo de dados	Descrição
declare_time	timestamp	O horário em que o cursor foi declarado.
total_bytes	bigint	O tamanho do conjunto de resultados do cursor, em bytes.
total_rows	bigint	O número de linhas no conjunto de resultados do cursor.
fetched_rows	bigint	O número de linhas que foram obtidas do conjunto de resultados do cursor.
cursor_storage_limit_used_percent	integer	A porcentagem de espaço em disco em uso no momento pelo cursor.

SVV_ALL_COLUMNS

Use SVV_ALL_COLUMNS para exibir uma união de colunas das tabelas do Amazon Redshift, conforme mostrado em SVV_REDSHIFT_COLUMNS e na lista consolidada de todas as colunas externas de todas as tabelas externas. Para obter informações sobre as colunas do Amazon Redshift, consulte SVV_REDSHIFT_COLUMNS.

SVV_ALL_COLUMNS é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Para obter mais informações sobre as práticas recomendadas ao consultar tabelas e visualizações do sistema, consulte Consulta de tabelas do sistema.

> **Note**
>
> Use o comando SHOW COLUMNS para descobrir colunas. SHOW COLUMNS funciona de forma consistente em contextos locais, de unidade de compartilhamento de dados e de

catálogos externos e é atualizado à medida que novos recursos são lançados. Para obter mais informações, consulte Práticas recomendadas para descobrir metadados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
database_name	varchar(128)	O nome do banco de dados.
schema_name	varchar(128)	O nome do esquema.
table_name	varchar(128)	O nome da tabela.
column_name	varchar(128)	O nome da coluna.
ordinal_position	integer	A posição da coluna na tabela.
column_default	varchar(4000)	O valor padrão da coluna.
is_nullable	varchar(3)	Um valor que indica se a coluna é anulável. Os valores possíveis são sim e não.
data_type	varchar(128)	O tipo de dados da coluna.
character_maximum_length	integer	O número máximo de caracteres na coluna.
numeric_precision	integer	A precisão numérica.
numeric_scale	integer	A escala numérica.
remarks	varchar(256)	Observações.

Consultas de exemplo

O exemplo a seguir retorna a saída de SVV_ALL_COLUMNS.

```
SELECT *
```

```
FROM svv_all_columns
WHERE database_name = 'tickit_db'
    AND TABLE_NAME = 'tickit_sales_redshift'
ORDER BY COLUMN_NAME,
    SCHEMA_NAME
LIMIT 5;

 database_name | schema_name |     table_name        | column_name | ordinal_position
 | column_default | is_nullable | data_type | character_maximum_length |
 numeric_precision | numeric_scale | remarks
 -------------+------------+----------------------+-------------+-----------------
+---------------+-------------+-----------+--------------------------
+------------------+---------------+---------
   tickit_db   |    public   | tickit_sales_redshift |    buyerid  |        4         |
               |      NO     |  integer  |                          |         32
  |       0       |
   tickit_db   |    public   | tickit_sales_redshift | commission  |        9         |
               |     YES     |  numeric  |                          |          8
  | 2       |
   tickit_db   |    public   | tickit_sales_redshift |    dateid   |        7         |
               |      NO     |  smallint |                          |         16
  |       0       |
   tickit_db   |    public   | tickit_sales_redshift |   eventid   |        5         |
               |      NO     |  integer  |                          |         32
  |       0       |
   tickit_db   |    public   | tickit_sales_redshift |    listid   |        2         |
               |      NO     |  integer  |                          |         32
  |       0       |
```

SVV_ALL_SCHEMAS

Use SVV_ALL_SCHEMAS para exibir uma união de esquemas do Amazon Redshift, conforme mostrado em SVV_REDSHIFT_SCHEMAS e na lista consolidada de todos os esquemas externos de todos os bancos de dados. Para obter mais informações sobre os esquemas do Amazon Redshift, consulte SVV_REDSHIFT_SCHEMAS.

SVV_ALL_SCHEMAS é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Para obter mais informações sobre as práticas recomendadas ao consultar tabelas e visualizações do sistema, consulte Consulta de tabelas do sistema.

> **Note**
>
> Use o comando SHOW SCHEMAS para descobrir esquemas. SHOW SCHEMAS funciona de forma consistente em contextos locais, de unidade de compartilhamento de dados e de catálogos externos e é atualizado à medida que novos recursos são lançados. Para obter mais informações, consulte Práticas recomendadas para descobrir metadados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
database_name	varchar(128)	O nome do banco de dados no qual o esquema existe.
schema_name	varchar(128)	O nome do esquema.
schema_owner	integer	O ID do usuário do proprietá rio do esquema. Para obter informações sobre IDs de usuário, consulte PG_USER_I NFO.
schema_type	varchar(128)	O tipo do esquema. Os valores possíveis são esquemas externos, locais e compartilhados.
schema_acl	varchar(128)	A string que define as permissões para o usuário ou o grupo de usuários especific ado para o esquema.
source_database	varchar(128)	O nome do banco de dados de fonte do esquema externo.

Nome da coluna	Tipo de dados	Descrição
schema_option	varchar(256)	As opções do esquema. Este é um atributo de esquema externo.

Consulta de exemplo

O exemplo a seguir retorna a saída de SVV_ALL_SCHEMAS.

```
SELECT *
FROM svv_all_schemas
WHERE database_name = 'tickit_db'
ORDER BY database_name,
    SCHEMA_NAME;


 database_name |    schema_name     | schema_owner | schema_type | schema_acl |
 source_database | schema_option
---------------+--------------------+--------------+-------------+------------
+-----------------+---------------
   tickit_db   |       public       |       1      |   shared    |            |
         |
```

SVV_ALL_TABLES

Use SVV_ALL_TABLES para exibir uma união de tabelas do Amazon Redshift como mostrado em SVV_REDSHIFT_TABLES e na lista consolidada de todas as tabelas externas de todos os esquemas externos. Para obter informações sobre tabelas do Amazon Redshift, consulte SVV_REDSHIFT_TABLES.

SVV_TABLES é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Para obter mais informações sobre as práticas recomendadas ao consultar tabelas e visualizações do sistema, consulte Consulta de tabelas do sistema.

> **Note**
>
> Use o comando SHOW TABLES para descobrir tabelas. SHOW TABLES funciona de forma consistente em contextos locais, de unidade de compartilhamento de dados e de catálogos externos e é atualizado à medida que novos recursos são lançados. Para obter mais informações, consulte Práticas recomendadas para descobrir metadados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
database_name	varchar(128)	O nome do banco de dados onde a tabela existe.
schema_name	varchar(128)	O nome do esquema para a tabela.
table_name	varchar(128)	O nome da tabela.
table_acl	varchar(128)	A string que define a permissão para o usuário ou o grupo de usuários especific ado para a tabela.
table_type	varchar(128)	O tipo da tabela. Os valores possíveis são exibições, tabelas base, tabelas externas e tabelas compartilhadas.
remarks	varchar(256)	Observações.

Consultas de exemplo

O exemplo a seguir retorna a saída de SVV_ALL_TABLES.

```
SELECT *
FROM svv_all_tables
```

```
WHERE database_name = 'tickit_db'
ORDER BY TABLE_NAME,
    SCHEMA_NAME
LIMIT 5;

 database_name | schema_name |        table_name        | table_type | table_acl |
 remarks
---------------+-------------+--------------------------+------------+-----------
+---------
   tickit_db   |    public   | tickit_category_redshift |    TABLE   |           |
   tickit_db   |    public   |   tickit_date_redshift   |    TABLE   |           |
   tickit_db   |    public   |   tickit_event_redshift  |    TABLE   |           |
   tickit_db   |    public   | tickit_listing_redshift  |    TABLE   |           |
   tickit_db   |    public   |   tickit_sales_redshift  |    TABLE   |           |
```

Se o valor table_acl for nulo, nenhum privilégio de acesso foi concedido explicitamente à tabela correspondente.

SVV_ALTER_TABLE_RECOMMENDATIONS

Regista as recomendações atuais do Amazon Redshift Advisor para tabelas. Esta visualização mostra recomendações para todas as tabelas, estejam elas definidas para otimização automática ou não. Para verificar se uma tabela está definida para otimização automática, consulte SVV_TABLE_INFO. As entradas aparecem somente para tabelas visíveis no banco de dados da sessão atual. Depois que uma recomendação for aplicada (pelo Amazon Redshift ou por você), ela não aparecerá mais na visualização.

SVV_ALTER_TABLE_RECOMMENDATIONS é visível apenas para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
type	character (30)	O tipo de recomendação. Os valores possíveis são distkey e sortkey.
banco de dados	character (128)	O nome do banco de dados.

Nome da coluna	Tipo de dados	Descrição
table_id	integer	O identificador da tabela.
group_id	integer	O número do grupo de um conjunto de recomendações. Todas as recomendações em um grupo devem ser aplicadas para ver o máximo benefício. Os valores possíveis são -1 para uma recomendação de chave de classificação e um número maior que zero para uma recomendação de chave de distribuição.
ddl	character (1024)	A instrução de SQL que deve ser executada para aplicar a recomendação.
auto_eligible	character (1)	O valor indica se a recomendação é elegível para que o Amazon Redshift seja executado automaticamente. Se este valor for `t`, então a indicação é true (verdadeira), se `f`, então é false (falsa).

Consultas de exemplo

No exemplo a seguir, as linhas no resultado mostram recomendações para chave de distribuição e chave de classificação. As linhas também mostram se as recomendações estão qualificadas para que o Amazon Redshift as aplique automaticamente.

```
select type, database, table_id, group_id, ddl, auto_eligible
from svv_alter_table_recommendations;
```

```
 type      | database | table_id | group_id | ddl

                   | auto_eligible
 diststyle | db0      | 117884   | 2        | ALTER TABLE "sch"."dp21235_tbl_1" ALTER
 DISTSTYLE KEY DISTKEY "c0"
                     | f
 diststyle | db0      | 117892   | 2        | ALTER TABLE "sch"."dp21235_tbl_1" ALTER
 DISTSTYLE KEY DISTKEY "c0"
                     | f
```

```
diststyle | db0      | 117885   | 1        | ALTER TABLE "sch"."catalog_returns"
ALTER DISTSTYLE KEY DISTKEY "cr_sold_date_sk", ALTER COMPOUND SORTKEY
("cr_sold_date_sk","cr_returned_time_sk") | t
sortkey   | db0      | 117890   | -1       | ALTER TABLE "sch"."customer_addresses"
ALTER COMPOUND SORTKEY ("ca_address_sk")
                      | t
```

SVV_ATTACHED_MASKING_POLICY

Use SVV_ATTACHED_MASKING_POLICY para visualizar todas as relações e perfis/usuários com políticas anexadas no banco de dados conectado.

Somente superusuários e usuários com a função `sys:secadmin` podem exibir SVV_ATTACHED_MASKING_POLICY. Usuários regulares verão 0 linha.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
policy_name	texto	O nome da política de mascaramento anexada à tabela.
schema_name	texto	O esquema da tabela à qual a política está anexada.
table_name	texto	O nome da tabela à qual a política está anexada.
table_type	texto	O tipo da tabela à qual a política está anexada.
grantor	texto	O nome do usuário que anexou a política.
grantee	texto	O nome do usuário/perfil ao qual a política está anexada.
grantee_type	texto	O tipo de favorecido. Pode ser role, user ou public.

Nome da coluna	Tipo de dados	Descrição
priority	int	A prioridade da política anexada.
input_columns	texto	Os atributos da coluna de entrada da política anexada.
output_columns	texto	Os atributos da coluna de saída da política anexada.
is_masking_datashare_on	booleano	Se a tabela à qual a política está anexada for protegida por DDM por meio de unidades de compartilhamento de dados.

Funções internas

SVV_ATTACHED_MASKING_POLICY é compatível com as seguintes funções internas:

mask_get_policy_for_role_on_column

Obtenha a política de maior prioridade que se aplica a determinado par de coluna/perfil.

Sintaxe

```
mask_get_policy_for_role_on_column
                        (relschema,
                        relname,
                        colname,
                        rolename);
```

Parâmetros

relschema

O nome do esquema no qual a política está.

relname

O nome da tabela na qual a política está.

colname

O nome da coluna à qual a política está anexada.

rolename

O nome do perfil ao qual a política está anexada.

mask_get_policy_for_user_on_column

Obtenha a política de maior prioridade que se aplica a determinado par de coluna/usuário.

Sintaxe

```
mask_get_policy_for_user_on_column
                        (relschema,
                        relname,
                        colname,
                        username);
```

Parâmetros

relschema

O nome do esquema no qual a política está.

relname

O nome da tabela na qual a política está.

colname

O nome da coluna à qual a política está anexada.

rolename

O nome do usuário ao qual a política está anexada.

SVV_COLUMNS

Use a exibição SVV_COLUMNS para visualizar as informações do catálogo sobre as colunas das tabelas e exibições locais e externas, incluindo as exibições de vinculação tardia.

SVV_COLUMNS é visível a todos os usuários por padrão. Para controlar o acesso aos metadados do banco de dados, habilite a segurança de metadados para seu cluster provisionado ou grupo

de trabalho sem servidor. A segurança de metadados permite que você separe as permissões de visualização dos metadados do objeto por usuários e perfis. Para obter mais informações, consulte Segurança de metadados.

A visualização SVV_COLUMNS une os metadados das Tabelas de catálogo do sistema (tabelas com um prefixo PG) e da visualização do sistema SVV_EXTERNAL_COLUMNS. As tabelas do catálogo do sistema descrevem as tabelas do banco de dados Amazon Redshift. SVV_EXTERNAL_COLUMNS descreve tabelas externas que são usadas com o Amazon Redshift Spectrum.

Todos os usuários podem visualizar todas as linhas das tabelas de catálogo do sistema. Os usuários normais podem ver as definições de colunas na visualização SVV_EXTERNAL_COLUMNS somente para tabelas externas às quais foi atribuído acesso. Embora possam ver metadados de tabela nas tabelas de catálogo do sistema, usuários regulares só poderão selecionar dados de tabelas definidas pelo usuário se forem os proprietários da tabela ou se tiverem recebido acesso.

Para obter mais informações sobre as práticas recomendadas ao consultar tabelas e visualizações do sistema, consulte Consulta de tabelas do sistema.

> **Note**
>
> Use o comando SHOW COLUMNS para descobrir colunas. SHOW COLUMNS funciona de forma consistente em contextos locais, de unidade de compartilhamento de dados e de catálogos externos e é atualizado à medida que novos recursos são lançados. Para obter mais informações, consulte Práticas recomendadas para descobrir metadados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
table_catalog	texto	O nome do catálogo onde a tabela se encontra.
table_schema	texto	O nome do esquema para a tabela.
table_name	texto	O nome da tabela.

Nome da coluna	Tipo de dados	Descrição
column_name	texto	O nome da coluna.
ordinal_position	int	A posição da coluna na tabela.
column_default	texto	O valor padrão da coluna.
is_nullable	texto	Um valor que indica se a coluna é anulável.
data_type	texto	O tipo de dados da coluna.
character_maximum_length	int	O número máximo de caracteres na coluna.
numeric_precision	int	A precisão numérica. Se a coluna data_type for numérica, ela retornará o número de dígitos significativos no valor inteiro.
numeric_precision_radix	int	A raiz de precisão numérica. Se a coluna data_type for numérica, ela retornará a base das colunas numeric_precision e numeric_scale.
numeric_scale	int	A escala numérica. Se a coluna data_type for numérica, ela retornará o número de dígitos significativos no valor decimal.
datetime_precision	int	A precisão da data e hora.
interval_type	texto	O tipo do intervalo.
interval_precision	texto	A precisão do intervalo.

Nome da coluna	Tipo de dados	Descrição
character_set_catalog	texto	O catálogo do conjunto de caracteres.
character_set_schema	texto	O esquema do conjunto de caracteres.
character_set_name	texto	O nome do conjunto de caracteres.
collation_catalog	texto	O catálogo de agrupamentos.
collation_schema	texto	O esquema do agrupamento.
collation_name	texto	O nome do agrupamento.
domain_name	texto	O nome do domínio.
remarks	texto	Observações.

SVV_COLUMN_PRIVILEGES

Use SVV_COLUMN_PRIVILEGES para exibir as permissões de coluna que são explicitamente concedidas a usuários, funções e grupos no banco de dados atual.

SVV_COLUMN_PRIVILEGES permanece visível para os seguintes usuários:

- Superusuários
- Usuários com a permissão ACCESS SYSTEM TABLE

Os outros usuários só podem ver identidades às quais tenham acesso ou sejam proprietários.

Para obter mais informações sobre as práticas recomendadas ao consultar tabelas e visualizações do sistema, consulte Consulta de tabelas do sistema.

> **Note**
>
> Use o comando SHOW COLUMN GRANTS para descobrir permissões. SHOW COLUMN GRANTS funciona de forma consistente em contextos locais, de unidade de compartilhamento de dados e de catálogos externos e é atualizado à medida que novos recursos são lançados. Para obter mais informações, consulte Práticas recomendadas para descobrir metadados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
namespace_name	texto	O nome do namespace no qual uma relação especificada existe.
relation_name	texto	O nome da relação.
column_name	texto	O nome da coluna.
privilege_type	texto	O tipo de permissão. Os valores possíveis são SELECT ou UPDATE.
identity_id	integer	O ID da identidade. Os valores possíveis são ID do usuário, ID da função ou ID do grupo.
identity_name	texto	O nome da identidade.
identity_type	texto	O tipo da identidade. Os valores possíveis são usuário, função, grupo ou público.

Consulta de exemplo

O exemplo a seguir exibe o resultado de SVV_COLUMN_PRIVILEGIES.

```
SELECT
 namespace_name,relation_name,COLUMN_NAME,privilege_type,identity_name,identity_type
FROM svv_column_privileges WHERE relation_name = 'lineitem';

 namespace_name | relation_name | column_name | privilege_type | identity_name |
 identity_type
----------------+---------------+-------------+----------------+---------------
+----------------
    public      |   lineitem    | l_orderkey  |     SELECT     |    reguser    |
 user
    public      |   lineitem    | l_orderkey  |     SELECT     |     role1     |
 role
    public      |   lineitem    | l_partkey   |     SELECT     |    reguser    |
 user
    public      |   lineitem    | l_partkey   |     SELECT     |     role1     |
 role
```

SVV_COPY_JOB_INTEGRATIONS

Use SVV_COPY_JOB_INTEGRATIONS para visualizar detalhes das integrações de eventos do S3.

Esta visualização contém as integrações de eventos do S3 que foram criadas.

SVV_COPY_JOB_INTEGRATIONS é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
job_owner	integer	O identificador do proprietário do trabalho.
channel_arn	name	O identificador da integração.
bucket	texto	O nome do bucket do Amazon S3 associado à integração.

Nome da coluna	Tipo de dados	Descrição
channel state	name	O estado da integração. Valores válidos: `Pending`, `Established` e `Inactive`.
db_name	name	O nome do banco de dados do objeto dependente.
job_name	texto	O nome do trabalho.
job_state	integer	O estado do trabalho. Valores válidos: `0` para ativo; 1 para pendente.

O exemplo a seguir retorna integrações do S3 para o banco de dados atual.

```
SELECT * FROM SVV_COPY_JOB_INTEGRATIONS WHERE db_name = pg_catalog.current_database();
```

SVV_DATABASE_PRIVILEGES

Use SVV_DATABASE_PRIVILEGES para exibir as permissões de banco de dados que são explicitamente concedidas a usuários, funções e grupos em seu cluster do Amazon Redshift.

SVV_DATABASE_PRIVILEGES permanece visível para os seguintes usuários:

- Superusuários
- Usuários com a permissão ACCESS SYSTEM TABLE

Os outros usuários só podem ver identidades às quais tenham acesso ou sejam proprietários.

Para obter mais informações sobre as práticas recomendadas ao consultar tabelas e visualizações do sistema, consulte Consulta de tabelas do sistema.

> **Note**
>
> Use o comando SHOW GRANTS para descobrir permissões. SHOW GRANTS funciona de forma consistente em contextos locais, de unidade de compartilhamento de dados e de

catálogos externos e é atualizado à medida que novos recursos são lançados. Para obter mais informações, consulte Práticas recomendadas para descobrir metadados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
database_name	texto	O nome do banco de dados.
privilege_type	texto	O tipo de permissão. Com relação a permissões com um privilege_scope de banco de dados, os valores possíveis são USAGE, CREATE, TEMPORARY, TEMP e ALTER. Quanto a valores de privilege_scope que não são de banco de dados, os valores possíveis incluem qualquer tipo de permissão disponível no escopo da permissão.
identity_id	integer	O ID da identidade. Os valores possíveis são ID do usuário, ID da função ou ID do grupo.
identity_name	texto	O nome da identidade.
identity_type	texto	O tipo da identidade. Os valores possíveis são usuário, função, grupo ou público.
admin_option	booleano	Um valor que indica se o usuário pode conceder permissão a outros usuários e funções. É sempre falso para o tipo de identidade de função e grupo.
privilege_scope	texto	O escopo da permissão especificada em privilege_type. Os valores possíveis são: • DATABASE • SCHEMAS • TABLES

Nome da coluna	Tipo de dados	Descrição
		• FUNCTIONS • LANGUAGES Para ter informações sobre as permissões com escopo, acesse Permissões em escopo.

Consulta de exemplo

O exemplo a seguir exibe o resultado de SVV_DATABASE_PRIVILEGIES.

```
SELECT database_name,privilege_type,identity_name,identity_type,admin_option FROM
 svv_database_privileges
WHERE database_name = 'test_db';

 database_name | privilege_type | identity_name | identity_type | admin_option
---------------+----------------+---------------+---------------+--------------
     test_db   |     CREATE     |     reguser   |      user     |     False
     test_db   |     CREATE     |      role1    |      role     |     False
     test_db   |     TEMP       |      public   |      public   |     False
     test_db   |     TEMP       |      role1    |      role     |     False
```

SVV_DATASHARE_PRIVILEGES

Use SVV_DATASHARE_PRIVILEGES para exibir as permissões de unidade de compartilhamento de dados que são explicitamente concedidas a usuários, funções e grupos em seu cluster do Amazon Redshift.

SVV_DATASHARE_PRIVILEGES permanece visível para os seguintes usuários:

- Superusuários
- Usuários com a permissão ACCESS SYSTEM TABLE

Os outros usuários só podem ver identidades às quais tenham acesso ou sejam proprietários.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
datashare_name	texto	O nome do datashare.
privilege_type	texto	O tipo de permissão. Os valores possíveis são ALTER ou SHARE.
identity_id	integer	O ID da identidade. Os valores possíveis são ID do usuário, ID da função ou ID do grupo.
identity_name	texto	O nome da identidade.
identity_type	texto	O tipo da identidade. Os valores possíveis são usuário, função, grupo ou público.
admin_option	booleano	Um valor que indica se o usuário pode conceder permissão a outros usuários e funções. É sempre falso para o tipo de identidade de função e grupo.

Consulta de exemplo

O exemplo a seguir exibe o resultado de SVV_DATASHARE_PRIVILEGIES.

```
SELECT datashare_name,privilege_type,identity_name,identity_type,admin_option FROM
 svv_datashare_privileges
WHERE datashare_name = 'demo_share';

 datashare_name | privilege_type |  identity_name | identity_type | admin_option
----------------+----------------+----------------+---------------+--------------
   demo_share   |     ALTER      |    superuser   |     user      |   False
   demo_share   |     ALTER      |    reguser     |     user      |   False
```

SVV_DATASHARES

Use SVV_DATASHARES para visualizar uma lista de unidades de compartilhamento de dados criadas no cluster e unidades de compartilhamento de dados compartilhadas com o cluster.

SVV_DATASHARES permanece visível para os seguintes usuários:

- Superusuários
- Proprietários da unidade de compartilhamento de dados
- Usuários com permissões ALTER ou USAGE em uma unidade de compartilhamento de dados

Outros usuários não podem ver nenhuma linha. Para obter informações sobre as permissões ALTER e USAGE, consulte o GRANT.

Para obter mais informações sobre as práticas recomendadas ao consultar tabelas e visualizações do sistema, consulte Consulta de tabelas do sistema.

> **Note**
>
> Use o comando SHOW DATASHARES para descobrir unidades de compartilhamento de dados. SHOW DATASHARES é atualizado à medida que novos recursos são lançados. Para obter mais informações, consulte Práticas recomendadas para descobrir metadados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
share_name	varchar(128)	O nome de um datashare.
share_id	integer	O ID do datashare.
share_owner	integer	O proprietário do datashare.
source_database	varchar(128)	O banco de dados de fonte para este datashare.
consumer_database	varchar(128)	O banco de dados do consumidor que é criado a partir deste datashare.

Nome da coluna	Tipo de dados	Descrição
share_type	varchar(8)	O tipo do datashare. Valores possíveis são INBOUND e OUTBOUND.
createdate	time stamp sem fuso horário	Data em que o datashare foi criado.
is_publicaccessible	booleano	A propriedade que especifica se um datashare pode ser compartilhado para um cluster acessível publicamente.
share_acl	varchar(256)	A string que define as permissões para o usuário ou o grupo de usuários especificado para a unidade de compartilhamento de dados.
producer_account	varchar(16)	O ID da conta de produtor de datashare.
producer_namespace	varchar(64)	O identificador de cluster exclusivo para o cluster do produtor de datashare.
managed_by	varchar(64)	A propriedade que especifica o produto da AWS que gerencia a unidade de compartilhamento de dados.

Observações de uso

Recuperação de metadados adicionais: usando o número inteiro exibido na coluna `share_owner`, é possível unir com `usesysid` em SVL_USER_INFO para ter dados sobre o proprietário da unidade de compartilhamento de dados. Isso inclui o nome e as propriedades adicionais.

Consulta de exemplo

O seguinte exemplo retorna a saída para SVV_DATASHARES.

```
SELECT share_owner, source_database, share_type, is_publicaccessible
FROM svv_datashares
WHERE share_name LIKE 'tickit_datashare%'
AND source_database = 'dev';

  share_owner | source_database | share_type  | is_publicaccessible
--------------+-----------------+-------------+----------------------
     100      |      dev        |   OUTBOUND  |        True
(1 rows)
```

O exemplo a seguir retorna a saída para SVV_DATASHARES para unidades de compartilhamento de dados de saída.

```
SELECT share_name, share_owner, btrim(source_database), btrim(consumer_database),
 share_type, is_publicaccessible, share_acl, btrim(producer_account),
 btrim(producer_namespace), btrim(managed_by) FROM svv_datashares WHERE share_type =
 'OUTBOUND';

   share_name   | share_owner | source_database | consumer_database | share_type |
 is_publicaccessible | share_acl | producer_account|         producer_namespace
   | managed_by
----------------+-------------+-----------------+-------------------
+------------+---------------------+-----------+-----------------
+--------------------------------------+------------
    salesshare  |      1      |       dev       |                   |  OUTBOUND  |
   True         |           |   123456789012  | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d |

 marketingshare |      1      |       dev       |                   |  OUTBOUND  |
   True         |           |   123456789012  | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d |
```

O exemplo a seguir retorna a saída para SVV_DATASHARES para unidades de compartilhamento de dados de entrada.

```
SELECT share_name, share_owner, btrim(source_database), btrim(consumer_database),
 share_type, is_publicaccessible, share_acl, btrim(producer_account),
 btrim(producer_namespace), btrim(managed_by) FROM svv_datashares WHERE share_type =
 'INBOUND';
```

```
 share_name    | share_owner | source_database | consumer_database | share_type |
is_publicaccessible | share_acl | producer_account |         producer_namespace
   | managed_by
---------------+-------------+-----------------+-------------------
+-----------+--------------------+----------+-----------------
+--------------------------------------+-----------
 salesshare    |             |                 |                   |  INBOUND   |
  False         |           |  123456789012    | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d
|
marketingshare |             |                 |                   |  INBOUND   |
 False         |           |  123456789012    | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d |
ADX
```

SVV_DATASHARE_CONSUMERS

Use SVV_DATASHARE_CONSUMERS para visualizar uma lista de consumidores para um datashare criado em um cluster.

SVV_DATASHARE_CONSUMERS permanece visível para os seguintes usuários:

- Superusuários
- Proprietários da unidade de compartilhamento de dados
- Usuários com permissões ALTER ou USAGE em uma unidade de compartilhamento de dados

Outros usuários não podem ver nenhuma linha. Para obter informações sobre as permissões ALTER e USAGE, consulte o GRANT.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
share_name	varchar(128)	O nome do datashare.
consumer_account	varchar(16)	O ID da conta para o consumidor de datashare.
consumer_namespace	varchar(64)	O identificador de cluster exclusivo do cluster do consumidor de datashare.

Nome da coluna	Tipo de dados	Descrição
share_date	time stamp sem fuso horário	A data em que o datashare foi compartilhado.

Consulta de exemplo

O exemplo a seguir retorna a saída para SVV_DATASHARE_CONSUMERS.

```
SELECT count(*)
FROM svv_datashare_consumers
WHERE share_name LIKE 'tickit_datashare%';

1
```

SVV_DATASHARE_OBJECTS

Use SVV_DATASHARE_OBJECTS para visualizar uma lista de objetos em todos os datashares criados no cluster ou compartilhados com o cluster.

SVV_DATASHARE_OBJECTS permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Para obter informações sobre como exibir uma lista de unidades de compartilhamento de dados, consulte SVV_DATASHARES.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
share_type	varchar(8)	O tipo do datashare especific ado. Valores possíveis são OUTBOUND e INBOUND.
share_name	varchar(128)	O nome do datashare.
object_type	varchar(64)	O tipo de um objeto especific ado. Os valores possíveis

Nome da coluna	Tipo de dados	Descrição
		são esquemas, tabelas, visualizações, visualizações de vinculação tardia, visões materializadas e funções.
nome_objeto	varchar(512)	O nome do objeto. O nome do objeto se estende para incluir o nome do esquema, como schema1.t1.
producer_account	varchar(16)	O ID da conta de produtor de datashare.
producer_namespace	varchar(64)	O identificador de cluster exclusivo para o cluster do produtor de datashare.
include_new	booleano	A propriedade que especific a se deseja adicionar futuras tabelas, visualizações ou funções definidas pelo usuário (UDFs) SQL criadas no esquema especificado para o datashare. Este parâmetro é relevante apenas para datashares OUTBOUND e apenas para tipos de esquema no datashare.

Consulta de exemplo

O exemplo a seguir retorna a saída para SVV_DATASHARE_OBJECTS.

```
SELECT share_type,
    btrim(share_name)::varchar(16) AS share_name,
    object_type,
```

```
    object_name
FROM svv_datashare_objects
WHERE share_name LIKE 'tickit_datashare%'
AND object_name LIKE '%tickit%'
ORDER BY object_name
LIMIT 5;

 share_type |     share_name     | object_type |          object_name
------------+--------------------+-------------+--------------------------------
 OUTBOUND   |  tickit_datashare  |    table    |  public.tickit_category_redshift
 OUTBOUND   |  tickit_datashare  |    table    |  public.tickit_date_redshift
 OUTBOUND   |  tickit_datashare  |    table    |  public.tickit_event_redshift
 OUTBOUND   |  tickit_datashare  |    table    |  public.tickit_listing_redshift
 OUTBOUND   |  tickit_datashare  |    table    |  public.tickit_sales_redshift
```

```
SELECT * FROM SVV_DATASHARE_OBJECTS WHERE share_name like 'sales%';

share_type | share_name | object_type | object_name  | producer_account |
 producer_namespace          | include_new
-----------+------------+-------------+--------------+-----------------
+--------------------------------------+-------------
 OUTBOUND  | salesshare | schema      | public       | 123456789012     |
 13b8833d-17c6-4f16-8fe4-1a018f5ed00d |      t
 OUTBOUND  | salesshare | table       | public.sales | 123456789012     |
 13b8833d-17c6-4f16-8fe4-1a018f5ed00d |
```

SVV_DEFAULT_PRIVILEGES

Use SVV_DEFAULT_PRIVILEGES para visualizar os privilégios padrão aos quais um usuário tem acesso em um cluster do Amazon Redshift.

SVV_DEFAULT_PRIVILEGES permanece visível para os seguintes usuários:

- Superusuários
- Usuários com a permissão ACCESS SYSTEM TABLE

Os outros usuários só podem ver permissões padrão concedidas a eles.

Para obter mais informações sobre as práticas recomendadas ao consultar tabelas e visualizações do sistema, consulte Consulta de tabelas do sistema.

> **Note**
>
> Use o comando SHOW GRANTS para descobrir permissões. SHOW GRANTS funciona de forma consistente em contextos locais, de unidade de compartilhamento de dados e de catálogos externos e é atualizado à medida que novos recursos são lançados. Para obter mais informações, consulte Práticas recomendadas para descobrir metadados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
schema_name	texto	O nome do esquema.
object_type	texto	O tipo do objeto. Os valores possíveis são RELATION, FUNCTION ou PROCEDURE.
owner_id	integer	O ID do proprietário. O valor possível é o ID do usuário.
owner_name	texto	O nome do proprietário.
owner_type	texto	Tipo do proprietário. O valor possível é usuário.
privilege_type	texto	O tipo de privilégio. Os valores possíveis são INSERT, SELECT, UPDATE, DELETE, RULE, REFERENCES TRIGGER, DROP e EXECUTE.
grantee_id	integer	O ID da entidade autorizada. Os valores possíveis são ID do usuário, ID da função ou ID do grupo.
grantee_type	texto	O tipo da entidade autorizada. Os valores possíveis são usuário, função e público.
admin_option	booleano	O valor que indica se o usuário pode conceder permissões a outros usuários e funções. É sempre falso para o tipo de função e grupo.

Consulta de exemplo

O exemplo a seguir retorna a saída para SVV_DEFAULT_PRIVILEGES.

```
SELECT * from svv_default_privileges;

 schema_name |    object_type    | owner_id | owner_name | owner_type | privilege_type
 | grantee_id | grantee_name | grantee_type | admin_option
-------------+------------------+--------- +------------+------------+----------------
+------------+--------------+--------------+-------------+
   public    |     RELATION      |    106   |     u1     |    user    |     UPDATE
 |     107    |      u2      |     user     |      f      |
   public    |     RELATION      |    106   |     u1     |    user    |     SELECT
 |     107    |      u2      |     user     |      f      |
```

SVV_DISKUSAGE

O Amazon Redshift cria a exibição do sistema SVV_DISKUSAGE unindo as tabelas STV_TBL_PERM e STV_BLOCKLIST. A exibição SVV_DISKUSAGE contém informações sobre a alocação de dados para as tabelas de um banco de dados.

Use consultas agregadas com o SVV_DISKUSAGE, como mostram os exemplos a seguir, para determinar o número de blocos de disco de 1 MB alocados por banco de dados, tabela, fatia ou coluna. Cada bloco de dados usa 1 MB. Você também pode usar a STV_PARTITIONS para visualizar informações resumidas sobre a utilização do disco.

SVV_DISKUSAGE é visível somente para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> Essa visualização só está disponível ao consultar clusters provisionados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
db_id	integer	ID do banco de dados.
name	character (72)	O nome da tabela.
slice	integer	A fatia dos dados alocada para a tabela.
col	integer	Um índice com base zero para a coluna. Toda tabela que é criada tem três colunas ocultas anexadas a ela: INSERT_XID, DELETE_XID e ROW_ID (OID). Uma tabela com 3 colunas definidas pelo usuário contém na realidade 6 colunas, e as colunas definidas pelo usuário são numeradas internamente com 0, 1 e 2. As colunas INSERT_XID, DELETE_XID e ROW_ID são numeradas com 3, 4 e 5, respectivamente, neste exemplo.
tbl	integer	ID da tabela.
blocknum	integer	O ID do bloco de dados.
num_values	integer	O número de valores contidos no bloco.
minvalue	bigint	O valor mínimo contido no bloco.
maxvalue	bigint	O valor máximo contido no bloco.
sb_pos	integer	O identificador interno para a posição do superbloco no disco.
pinned	integer	Se o bloco é fixado ou não na memória como parte do pré-carregamento. 0 = false; 1 = true. O padrão é falso.
on_disk	integer	Se o bloco é ou não armazenado automaticamente no disco. 0 = false; 1 = true. O padrão é falso.
modified	integer	Se o bloco foi modificado ou não. 0 = false; 1 = true. O padrão é falso.

Nome da coluna	Tipo de dados	Descrição
hdr_modified	integer	Se o cabeçalho do bloco foi modificado ou não. 0 = false; 1 = true. O padrão é falso.
unsorted	integer	Se um bloco está ou não desordenado. 0 = false; 1 = true. O padrão é true (verdadeiro).
tombstone	integer	Para uso interno.
preferred_diskno	integer	O número do disco onde o bloco se encontra, a menos que o disco esteja com uma falha. Depois que o disco for corrigido, o bloco voltará para esse disco.
temporary	integer	Se o bloco contém ou não dados temporários, como de uma tabela temporária ou resultados de consulta intermediários. 0 = false; 1 = true. O padrão é falso.
newblock	integer	Indica se um bloco é ou não novo (true) ou nunca foi confirmado no disco (false). 0 = false; 1 = true.

Consultas de exemplo

A exibição SVV_DISKUSAGE contém uma linha por bloco de disco alocado, de maneira que uma consulta que selecione todas as linhas pode retornar um número muito grande de linhas. Recomendamos somente o uso de consultas agregadas com a SVV_DISKUSAGE.

Retorna o número mais alto de blocos atribuídos à coluna 6 na tabela USERS (a coluna EMAIL):

```
select db_id, trim(name) as tablename, max(blocknum)
from svv_diskusage
where name='users' and col=6
group by db_id, name;

db_id  | tablename | max
--------+-----------+-----
175857 | users     |   2
(1 row)
```

A consulta a seguir retorna resultados semelhantes para todas as colunas de uma tabela grande de 10 colunas chamada SALESNEW. (As últimas três linhas, nas colunas de 10 a 12, são reservadas para as colunas ocultas de metadados).

```
select db_id, trim(name) as tablename, col, tbl, max(blocknum)
from svv_diskusage
where name='salesnew'
group by db_id, name, col, tbl
order by db_id, name, col, tbl;

db_id  | tablename  | col |  tbl   | max
--------+------------+-----+--------+-----
175857 | salesnew   |   0 | 187605 | 154
175857 | salesnew   |   1 | 187605 | 154
175857 | salesnew   |   2 | 187605 | 154
175857 | salesnew   |   3 | 187605 | 154
175857 | salesnew   |   4 | 187605 | 154
175857 | salesnew   |   5 | 187605 |  79
175857 | salesnew   |   6 | 187605 |  79
175857 | salesnew   |   7 | 187605 | 302
175857 | salesnew   |   8 | 187605 | 302
175857 | salesnew   |   9 | 187605 | 302
175857 | salesnew   |  10 | 187605 |   3
175857 | salesnew   |  11 | 187605 |   2
175857 | salesnew   |  12 | 187605 | 296
(13 rows)
```

SVV_EXTERNAL_COLUMNS

Use a exibição SVV_EXTERNAL_COLUMNS para visualizar os detalhes das colunas em tabelas externas. Use SVV_EXTERNAL_COLUMNS também para consultas entre bancos de dados para exibir detalhes em todas as colunas da tabela em bancos de dados não conectados aos quais os usuários têm acesso.

SVV_EXTERNAL_COLUMNS é visível por todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Para obter mais informações sobre as práticas recomendadas ao consultar tabelas e visualizações do sistema, consulte Consulta de tabelas do sistema.

> **Note**
>
> Use o comando SHOW COLUMNS para descobrir colunas. SHOW COLUMNS funciona de forma consistente em contextos locais, de unidade de compartilhamento de dados e de catálogos externos e é atualizado à medida que novos recursos são lançados. Para obter mais informações, consulte Práticas recomendadas para descobrir metadados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
redshift_database_name	texto	O nome do banco de dados local do Amazon Redshift.
schemaname	texto	O nome do esquema externo do Amazon Redshift para a tabela externa.
tablename	texto	O nome da tabela externa.
columnname	texto	O nome da coluna.
external_type	texto	O tipo de dados da coluna.
columnnum	integer	O número da coluna externa, a partir de 1.
part_key	integer	Se a coluna é uma chave de partição, indica a ordem de chave. Se a coluna não for uma partição, o valor será **0**.
is_nullable	texto	Define se uma coluna é anulável ou não. Alguns valores são `true`, `false`, ou uma string vazia " " que

Nome da coluna	Tipo de dados	Descrição
		não representa nenhuma informação.

SVV_EXTERNAL_DATABASES

Use a exibição SVV_EXTERNAL_DATABASES para visualizar os detalhes dos bancos de dados externos.

SVV_EXTERNAL_DATABASES é visível por todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
eskind	integer	O tipo do catálogo externo do banco de dados; **1** indica um catálogo de dados, **2** indica uma metastore do Hive.
esoptions	texto	Os detalhes do catálogo onde o banco de dados reside.
databasename	texto	O nome do banco de dados no catálogo externo.
location	texto	A localização do banco de dados.
parameters	texto	Os parâmetros do banco de dados.

SVV_EXTERNAL_PARTITIONS

Use a exibição SVV_EXTERNAL_PARTITIONS para visualizar os detalhes das partições em tabelas externas.

SVV_EXTERNAL_PARTITIONS é visível por todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para ter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema..

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
schemaname	texto	O nome do esquema externo do Amazon Redshift para a tabela externa com as partições especificadas.
tablename	texto	O nome da tabela externa.
values	texto	Os valores para a partição.
location	texto	A localização da partição. O tamanho da coluna é limitado a 128 caracteres. Valores maiores são truncados.
input_format	texto	O formato da entrada.
output_format	texto	O formato da saída.
serialization_lib	texto	A biblioteca de serialização.
serde_parameters	texto	Os parâmetros de SerDe.
compressed	integer	Um valor que indica se a partição é compactada; **1** indica que é compactada, **0** indica que não é compactada.
parameters	texto	As propriedades da partição.

SVV_EXTERNAL_SCHEMAS

Use a exibição SVV_EXTERNAL_SCHEMAS para visualizar informações sobre os esquemas externos. Para obter mais informações, consulte CREATE EXTERNAL SCHEMA.

SVV_EXTERNAL_SCHEMAS é visível por todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Para obter mais informações sobre as práticas recomendadas ao consultar tabelas e visualizações do sistema, consulte Consulta de tabelas do sistema.

> **Note**
>
> Use o comando SHOW SCHEMAS para descobrir esquemas. SHOW SCHEMAS funciona de forma consistente em contextos locais, de unidade de compartilhamento de dados e de catálogos externos e é atualizado à medida que novos recursos são lançados. Para obter mais informações, consulte Práticas recomendadas para descobrir metadados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
esoid	oid	ID do esquema externo.
eskind	smallint	O tipo de catálogo externo para o esquema externo: 1 indica um catálogo de dados, 2 indica um metastore do Hive, 3 indica uma consulta federada para o Aurora PostgreSQL ou o Amazon RDS PostgreSQL, 4 indica um esquema para um banco de dados do Amazon Redshift local, 5 indica um esquema para um banco de dados remoto do Amazon Redshift, 6 indica um esquema para uma tabela de sistema, 8 indica um esquema para bancos de dados MySQL remotos, 9 indica um esquema para um fluxo de dados do Amazon Kinesis e 10 indica um fluxo de dados do Amazon Managed Streaming for Apache Kafka.

Nome da coluna	Tipo de dados	Descrição
schemaname	name	O nome do esquema externo.
esowner	integer	O ID de usuário do proprietário do esquema externo.
databasename	texto	O nome do banco de dados externo.
esoptions	texto	As opções do esquema externo.

Exemplo

O exemplo a seguir mostra os detalhes dos esquemas externos.

```
select * from svv_external_schemas;

esoid  | eskind | schemaname | esowner | databasename | esoptions

-------+--------+------------+---------+--------------
+-------------------------------------------------------------
100133 |      1 | spectrum   |     100 | redshift     |
 {"IAM_ROLE":"arn:aws:iam::123456789012:role/mySpectrumRole"}
```

SVV_EXTERNAL_TABLES

Use SVV_EXTERNAL_TABLES para ver os detalhes das tabelas externas. Para obter mais informações, consulte CREATE EXTERNAL SCHEMA. Use SVV_EXTERNAL_TABLES também para consultas entre bancos de dados para exibir metadados em todas as tabelas em bancos de dados não conectados às quais os usuários têm acesso.

SVV_EXTERNAL_TABLES é visível por todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Para obter mais informações sobre as práticas recomendadas ao consultar tabelas e visualizações do sistema, consulte Consulta de tabelas do sistema.

> **Note**
>
> Use o comando SHOW TABLES para descobrir tabelas. SHOW TABLES funciona de forma consistente em contextos locais, de unidade de compartilhamento de dados e de catálogos externos e é atualizado à medida que novos recursos são lançados. Para obter mais informações, consulte Práticas recomendadas para descobrir metadados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
redshift_database_name	texto	O nome do banco de dados local do Amazon Redshift.
schemaname	texto	O nome do esquema externo do Amazon Redshift para a tabela externa.
tablename	texto	O nome da tabela externa.
tabletype	texto	O tipo da tabela. Os valores são: TABLE, VIEW, MATERIALIZED VIEW, EXTERNAL TABLE ou uma string vazia que não represent a nenhuma informação. As visualizações materiali zadas do Iceberg criadas com USING ICEBERG aparecem com `tabletype = 'EXTERNAL TABLE'`. As visualizações materializadas do Iceberg são enumerada s somente para esquemas externos; aquelas acessadas por meio de `awsdataca`

Nome da coluna	Tipo de dados	Descrição
		`talog` ou dos catálogos de Tabelas do Amazon S3 não são enumeradas por SVV_EXTERNAL_TABLES. Para descobri-las, use o comando SHOW TABLES.
location	texto	A localização da tabela.
input_format	texto	O formato da entrada
output_format	texto	O formato da saída.
serialization_lib	texto	A biblioteca de serialização.
serde_parameters	texto	Os parâmetros de SerDe.
compressed	integer	Um valor que indica se a tabela é compactada; **1** indica que é compactada, **0** indica que não é compactada.
parameters	texto	As propriedades da tabela.

Exemplo

O exemplo a seguir mostra os detalhes de svv_external_tables com um predicado no esquema externo usado por uma consulta federada.

```
select schemaname, tablename from svv_external_tables where schemaname = 'apg_tpch';
schemaname  | tablename
------------+-----------
apg_tpch    | customer
apg_tpch    | lineitem
apg_tpch    | nation
apg_tpch    | orders
apg_tpch    | part
apg_tpch    | partsupp
```

```
apg_tpch    | region
apg_tpch    | supplier
(8 rows)
```

O seguinte exemplo lista as visualizações materializadas do Iceberg em esquemas externos, que aparecem com `tabletype = 'EXTERNAL TABLE'`:

```
SELECT schemaname, tablename, location
FROM svv_external_tables
WHERE tabletype = 'EXTERNAL TABLE';
```

SVV_FUNCTION_PRIVILEGES

Use SVV_FUNCTION_PRIVILEGES para exibir as permissões de função que são explicitamente concedidas a usuários, funções e grupos no banco de dados atual.

SVV_FUNCTION_PRIVILEGES permanece visível para os seguintes usuários:

- Superusuários
- Usuários com a permissão ACCESS SYSTEM TABLE

Os outros usuários só podem ver identidades às quais tenham acesso ou sejam proprietários.

Para obter mais informações sobre as práticas recomendadas ao consultar tabelas e visualizações do sistema, consulte Consulta de tabelas do sistema.

> **Note**
>
> Use o comando SHOW GRANTS para descobrir permissões. SHOW GRANTS funciona de forma consistente em contextos locais, de unidade de compartilhamento de dados e de catálogos externos e é atualizado à medida que novos recursos são lançados. Para obter mais informações, consulte Práticas recomendadas para descobrir metadados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
namespace_name	texto	O nome do namespace no qual uma função especificada existe.
function_name	texto	Nome da função.
argument_types	texto	Uma string que representa o tipo de argumento de entrada de uma função.
privilege_type	texto	O tipo de permissão. O valor possível é EXECUTE.
identity_id	integer	O ID da identidade. Os valores possíveis são ID do usuário, ID da função ou ID do grupo.
identity_name	texto	O nome da identidade.
identity_type	texto	O tipo da identidade. Os valores possíveis são usuário, função, grupo ou público.
admin_option	booleano	Um valor que indica se o usuário pode conceder permissão a outros usuários e funções. É sempre falso para o tipo de identidad e de função e grupo.

Consulta de exemplo

O exemplo a seguir exibe o resultado de SVV_FUNCTION_PRIVILEGIES.

```
SELECT
 namespace_name,function_name,argument_types,privilege_type,identity_name,identity_type,admin_o
 FROM svv_function_privileges
WHERE identity_name IN ('role1', 'reguser');
```

```
 namespace_name | function_name |       argument_types       | privilege_type |
 identity_name | identity_type | admin_option
----------------+---------------+----------------------------+----------------
+----------------+---------------+-------------
    public      | test_func1    | integer                    |    EXECUTE     |
 role1     |     role      |  False
    public      | test_func2    | integer, character varying |    EXECUTE     |
 reguser    |     user      |  False
```

SVV_GEOGRAPHY_COLUMNS

Use SVV_GEOGRAPHY_COLUMNS para ver a lista de colunas GEOGRAPHY em seu data warehouse. Essa lista de colunas inclui colunas de unidades de compartilhamento de dados.

SVV_GEOGRAPHY_COLUMNS é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
f_table_catalog	varchar(128)	O nome do banco de dados que contém a tabela com a coluna GEOGRAPHY.
f_table_schema	varchar(128)	O nome do esquema que contém a tabela com a coluna GEOGRAPHY.
f_table_name	varchar(128)	O nome da tabela com a coluna GEOGRAPHY.
f_geography_column	varchar(128)	O nome da coluna GEOGRAPHY.
coord_dimension	integer	O número de dimensões dos dados de GEOGRAPHY.
srid	integer	O identificador do sistema de referência espacial (SRID) dos dados de GEOGRAPHY.

Nome da coluna	Tipo de dados	Descrição
type	varchar(128)	O nome do tipo de dado de geografia espacial.

Consulta de exemplo

O exemplo a seguir exibe o resultado de SVV_GEOGRAPHY_COLUMNS.

```
SELECT * FROM svv_geography_columns;

f_table_catalog  | f_table_schema  | f_table_name  | f_geography_column  |
 coord_dimension | srid |  type
-----------------+-----------------+--------------+--------------------
+-----------------+------+-------------
dev              | public          | spatial_test  | test_geography      | 2
    | 0    | GEOGRAPHY
```

SVV_GEOMETRY_COLUMNS

Use SVV_GEOMETRY_COLUMNS para ver a lista de colunas GEOMETRY em seu data warehouse. Essa lista de colunas inclui colunas de unidades de compartilhamento de dados.

SVV_GEOMETRY_COLUMNS é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
f_table_catalog	varchar(128)	O nome do banco de dados que contém a tabela com a coluna GEOMETRY.
f_table_schema	varchar(128)	O nome do esquema que contém a tabela com a coluna GEOMETRY.

Nome da coluna	Tipo de dados	Descrição
f_table_name	varchar(128)	O nome da tabela com a coluna GEOMETRY.
f_geometry_column	varchar(128)	O nome da coluna GEOMETRY.
coord_dimension	integer	O número de dimensões dos dados de GEOMETRY.
srid	integer	O identificador do sistema de referência espacial (SRID) da coluna GEOMETRY.
type	varchar(128)	O nome do tipo de geometria espacial.

Consulta de exemplo

O exemplo a seguir exibe o resultado de SVV_GEOMETRY_COLUMNS.

```
SELECT * FROM svv_geometry_columns;

f_table_catalog  | f_table_schema  | f_table_name  | f_geometry_column   |
 coord_dimension | srid |  type
-----------------+-----------------+---------------+--------------------
+-----------------+------+--------------
dev              | public          | accomodations | shape               | 2
    | 0    | GEOMETRY
dev              | public          | zipcode       | wkb_geometry        | 2
    | 0    | GEOMETRY
```

SVV_IAM_PRIVILEGES

Use SVV_IAM_PRIVILEGES para ver os privilégios do IAM concedidos explicitamente a usuários, perfis e grupos.

SVV_IAM_PRIVILEGES permanece visível para os seguintes usuários:

- Superusuários
- Usuários com a permissão ACCESS SYSTEM TABLE

Outros usuários só podem ver entradas às quais têm acesso.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
iam_arn	texto	Nome do namespace.
commanc ype	texto	Tipos de privilégios. Os valores possíveis são COPY, UNLOAD, CREATE MODEL ou EXTERNAL FUNCTION.
identity_ id	integer	ID da identidade. Os valores possíveis são ID do usuário, ID da função ou ID do grupo.
identity_ name	texto	Nome da identidade.
identity_ type	texto	Tipo da identidade. Os valores possíveis são usuário, função, grupo ou público.

Consultas de exemplo

O exemplo a seguir mostra os resultados de SVV_IAM_PRIVILEGES.

```
SELECT * from SVV_IAM_PRIVILEGES ORDER BY IDENTITY_ID;
       iam_arn        | command_type | identity_id | identity_name | identity_type
----------------------+--------------+-------------+---------------+---------------
 default-aws-iam-role | COPY         |           0 | public        | public
 default-aws-iam-role | UNLOAD       |           0 | public        | public
 default-aws-iam-role | CREATE MODEL |           0 | public        | public
 default-aws-iam-role | EXFUNC       |           0 | public        | public
 default-aws-iam-role | COPY         |         106 | u1            | user
 default-aws-iam-role | UNLOAD       |         106 | u1            | user
```

```
 default-aws-iam-role | CREATE MODEL |         106 | u1            | user
 default-aws-iam-role | EXFUNC       |         106 | u1            | user
 default-aws-iam-role | COPY         |      118413 | r1            | role
 default-aws-iam-role | UNLOAD       |      118413 | r1            | role
 default-aws-iam-role | CREATE MODEL |      118413 | r1            | role
 default-aws-iam-role | EXFUNC       |      118413 | r1            | role
(12 rows)
```

SVV_IDENTITY_PROVIDERS

A visualização SVV_IDENTITY_PROVIDERS retorna o nome e outras propriedades dos provedores de identidade. Para obter mais informações sobre como um provedor de identidade, consulte CREATE IDENTITY PROVIDER.

SVV_IDENTITY_PROVIDERS é visível somente para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
uid	integer	O ID exclusivo do provedor de identidades inscrito.
name	texto	O nome do provedor de identidade.
type	texto	O tipo de provedor de identidade.
instanceid	texto	O diferenciador exclusivo entre instâncias do mesmo tipo.
namespc	texto	O prefixo do namespace do provedor de identidade.
params	texto	O objeto JSON com parâmetros para o provedor de identidade.

Nome da coluna	Tipo de dados	Descrição
habilitado	bool	Indica se o provedor de identidadesestá habilitado.

Consultas de exemplo

Para visualizar propriedades do provedor de identidades, execute uma consulta como a apresentada a seguir depois de criar provedores de identidade.

```
SELECT name, type, instanceid, namespc, params, enabled
FROM svv_identity_providers
ORDER BY 1;
```

O exemplo de saída inclui descrições de parâmetros para o Microsoft Entra ID (antigo Azure AD):

```
       name       | type  |              instanceid              | namespc |

                                        params

                      | enabled
------------------+-------+--------------------------------------+---------
+-----------------------------------------------------------------------------------------------------
+---------
 rs5517_azure_idp | azure | e40d4bb2-7670-44ae-bfb8-5db013221d73 | abc     |
 {"issuer":"https://login.microsoftonline.com/e40d4bb2-7670-44ae-bfb8-5db013221d73/
v2.0", "client_id":"871c010f-5e61-4fb1-83ac-98610a7e9110", "client_secret":,
 "audience":["https://analysis.windows.net/powerbi/connector/AmazonRedshift", "https://
analysis.windows.net/powerbi/connector/AWSRDS"]} | t
(1 row)
```

A saída de exemplo inclui descrições de parâmetro para Centro de Identidade do AWS IAM:

```
   name   |  type  |                                  instanceid
                |     namespc      |
                        params
                  | enabled
----------+--------
+-------------------------------------------------------------------------------
+------------------
```

```
+-------------------------------------------------------------------------------------------------------------
+---------
 idc_name | awsidc | arn:aws:sso::123456789012:application/ssoins-12345f67fe123d4/
apl-a0b0a12dc123b1a4 |   idc_namespc   | {"iam_role":"arn:aws:iam::123456789012:role/
MyRedshiftRole","instance_arn":"arn:aws:sso:::instance/
ssoins-12345f67fe123d4","is_lakehouse_app":"true"}        | t
(1 row)
```

SVV_INTEGRATION

SVV_INTEGRATION exibe detalhes sobre a configuração das integrações.

SVV_INTEGRATION só permanece visível para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Para ter informações sobre Integrações ETL zero, consulte Integrações ETL zero.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
integration_id	character (128)	O identificador associado à integração.
target_database	character (128)	O banco de dados no Amazon Redshift que recebe os dados da integração.
origem	character (128)	Os dados de origem para a integração. Os tipos possíveis incluem `MySQL`, `PostgreSQL` e `S3_EVENT_NOTIFICATIONS` .
estado	character (128)	O estado da integração. Os possíveis valores incluem: `PendingDbConnectState` , `SchemaDiscoveryState` , `CdcRefreshState` e `ErrorState` .
current_lag	bigint	O tempo de atraso atual (milissegundos) entre a origem e o destino da integração.

Nome da coluna	Tipo de dados	Descrição
last_replicated_checkpoint	character (128)	O último ponto de verificação replicado.
total_tables_replicated	integer	O número total de tabelas atualmente no estado replicado.
total_tables_failed	integer	O número total de tabelas atualmente no estado com falha.
creation_time	timestamp	A hora (UTC) em que a integração foi criada. É definido como o momento em que o banco de dados de destino é criado a partir da integração.
refresh_interval	integer	O intervalo de tempo aproximado, em segundos, para atualizar os dados da origem de ETL zero para o banco de dados de destino.
source_database	character (128)	O nome do banco de dados de origem.
is_history_mode	booleano	Um valor de TRUE indica que o modo histórico está ativado. Um FALSE indica que o modo histórico está desativado.
latest_detected_change_time	timestamp	A hora (UTC) em que a alteração mais recente da origem desse banco de dados de integração foi preparada na fila de replicação.
latest_applied_change_time	timestamp	A hora (UTC) da última ingestão bem-sucedida no Amazon Redshift para esse banco de dados de destino.
auto_remediation	booleano	Um valor TRUE indica que a correção automática de linhas duplicadas está habilitada. Um valor FALSE indica que ela está desabilitada.

Nome da coluna	Tipo de dados	Descrição
latest_shipped_source_commit_time	timestamp	A hora (UTC), no relógio do banco de dados de origem, até a qual as transações da origem foram enviadas para essa integração.

Consultas de exemplo

O comando SQL a seguir exibe as integrações definidas atualmente.

```
select * from svv_integration;

           integration_id              | target_database | source |      state
   | current_lag |      last_replicated_checkpoint     | total_tables_replicated |
 total_tables_failed |       creation_time       |  refresh_interval  | source_database
 | is_history_mode
--------------------------------------+-----------------+--------+----------------
+-------------+--------------------------------------+-------------------------
+---------------------+---------------------------+--------------------
+-----------------+-----------------
  99108e72-1cfd-414f-8cc0-0216acefac77 |     perfdb      |  MySQL | CdcRefreshState |
  56606106  | {"txn_seq":9834,"txn_id":126597515} |            152          |
 0         | 2023-09-19 21:05:27.520299|      720           + mysourceetl     | f
```

SVV_INTEGRATION_TABLE_MAPPING

SVV_INTEGRATION_TABLE_MAPPING exibe o mapeamento de banco de dados, esquema, tabela, coluna e tipo de dados da origem para o destino quando o valor do identificador desses campos é diferente.

> **Note**
>
> Essa visão é preenchida somente para os seguintes tipos de Integração ETL zero:
>
> - AWS GlueAplicações de terceiros do para o Amazon SageMaker Lakehouse
> - Amazon DynamoDB para Amazon SageMaker Lakehouse

Para ter mais informações, consulte Integrações ETL zero no Guia do desenvolvedor do AWS Glue.

A transformação dos valores do identificador da origem para o destino segue as regras abaixo:

- Uma letra maiúscula é convertida em minúscula.
- Um caractere que não seja uma letra minúscula, um dígito ou um sublinhado (_) é convertido em sublinhado (_).
- Se houver conflito com um valor de identificador existente, um identificador exclusivo universal (UUID) será anexado ao novo identificador.
- Se o valor do identificador de origem for uma palavra-chave do Amazon Redshift, o sufixo `_redshift` será anexado ao novo identificador.

Após a transformação, um caractere deve ser uma letra minúscula, um dígito ou um sublinhado (_) e corresponder ao padrão regex `[a-z0-9_]`. Os seguintes exemplos demonstram as regras de conversão:

Origem	Target	Observações
foo	foo	Sem transformação
Barra	barra	
fooBar	foobar	
foo1	foo1	Sem transformação
foo_1	foo_1	Sem transformação
Bar@1	bar_1	
foo_bar@	foo_bar_	
case	case_redshift	

SVV_INTEGRATION_TABLE_MAPPING fica visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Para ter informações sobre Integrações ETL zero, consulte Integrações ETL zero no Guia de gerenciamento do Amazon Redshift.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
integration_id	character (128)	O identificador associado à integração.
source_database	character (128)	O nome do banco de dados na origem.
target_database	character (128)	O banco de dados no Amazon Redshift que recebe os dados da integração.
source_schema_name	character (128)	O nome do esquema na origem.
target_schema_name	character (128)	O esquema no Amazon Redshift que recebe os dados da integração.
source_table_name	character (128)	O nome da tabela na origem.
nome_tabela_destino	character (128)	A tabela no Amazon Redshift que recebe os dados da integração.
source_column_name	character (128)	O nome da coluna na origem.
target_column_name	character (128)	A coluna no Amazon Redshift que recebe os dados da integração.

Nome da coluna	Tipo de dados	Descrição
source_data_type	character (128)	O tipo de dados da coluna na origem.
target_data_type	character (128)	O tipo de dados da coluna no Amazon Redshift que recebe os dados da integração.

Consultas de exemplo

O comando SQL a seguir exibe o mapeamento de valores de metadados da origem para o destino.

```
select * from svv_integration_table_mapping;

           integration_id              | source_database | target_database |
 source_schema_name | target_schema_name | source_table_name | target_table_name |
--------------------------------------+-----------------+-----------------
+--------------------+--------------------+-------------------------------------+
  99108e72-1cfd-414f-8cc0-0216acefac77 |     mydatabase  |  mydatabase     |  myschema
         | myschema           | Mytable            | mytable           |


                                       | source_column_name | target_column_name |
 source_data_type | target_data_type |
                                      +--------------------+--------------------
+------------------+------------------+
                                       | Mycolumnname       | mycolumnname       |
 Mydatatype       | mydatatype       |
```

SVV_INTEGRATION_TABLE_STATE

SVV_INTEGRATION_TABLE_STATE exibe detalhes sobre as informações da integração no nível da tabela.

SVV_INTEGRATION_TABLE_STATE só permanece visível para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Para obter mais informações, consulte Integrações ETL zero.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
integration_id	character(128)	O identificador associado à integração.
target_database	character(128)	O nome do banco de dados do Amazon Redshift.
schema_name	character(128)	O nome do esquema do Amazon Redshift.
table_name	character(128)	O nome da tabela.
table_state	character(128)	O estado da tabela. Os valores possíveis são `Synced`, `Failed`, `Deleted`, `ResyncRequired`, `ResyncInitiated` e `DroppedSource`. O estado `DroppedSource` indica que a fonte de uma tabela do modo histórico foi descartada na origem.
table_last_replicated_checkpoint	character(128)	As coordenadas de log sincronizadas atuais.
reason	character(256)	O motivo da última transição de estado. Os motivos comuns podem ser tipos de dados incompatíveis com as tabelas, pois elas não têm chaves primárias. Para saber mais sobre como solucionar problemas comuns, consulte Solução de problemas de Integrações ETL zero no Amazon Redshift.
last_updated_timestamp	time stamp sem fuso horário	A hora (UTC) em que a tabela foi atualizada pela última vez.
table_rows	bigint	O número total de linhas na tabela.
table_size	bigint	O tamanho da tabela em megabytes (MB).

Nome da coluna	Tipo de dados	Descrição
is_history_mode	booleano	Um valor de TRUE indica que o modo histórico está ativado. Um FALSE indica que o modo histórico está desativado.

Consultas de exemplo

O comando SQL a seguir exibe as colunas do log de integrações.

```
select * from svv_integration_table_state;

          integration_id              | target_database | schema_name |     table_name
   | table_state  |table_last_replicated_checkpoint | reason | last_updated_timestamp
   |table_rows  | table_size | is_history_mode
--------------------------------------+-----------------+-------------
+------------------+-------------+---------------------------------+--------
+---------------------------+-----------+-----------+-----------------
 4798e675-8f9f-4686-b05f-92c538e19629 |  sample_test2   |    sample   |
 SampleTestChannel |    Synced    |   {"txn_seq":3,"txn_id":3122}   |        |
 2023-05-12 12:40:30.656625 | 2          |   16       | f
```

SVV_INTERLEAVED_COLUMNS

Use a exibição SVV_INTERLEAVED_COLUMNS para ajudar a determinar se uma tabela que usa as chaves de classificação intercaladas deve ser reindexada usando o VACUUM REINDEX. Para obter mais informações sobre como determinar a frequência de execução do VACUUM e quando executar o VACUUM REINDEX, consulte Minimizar tempos de limpeza.

SVV_INTERLEAVED_COLUMNS é visível somente para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
tbl	integer	ID da tabela.

Nome da coluna	Tipo de dados	Descrição
col	integer	Um índice com base zero para a coluna.
interleaved_skew	numeric 9,2)	A taxa que indica o grau de distorção presente nas colunas com chave de classificação intercalada para uma tabela. O valor 1.00 indica que não há distorção, e valores maiores indicam graus maiores de distorção. As tabelas com uma grande distorção devem ser reindexadas com o comando VACUUM REINDEX.
last_reindex	timestar	O horário em que o último VACUUM REINDEX foi executado para a tabela especificada. Esse valor será NULL se uma tabela nunca tiver sido reindexada ou se a tabela de log do sistema subjacente, STL_VACUUM, tiver sido rotacionada desde a última reindexação.

Consultas de exemplo

Para identificar as tabelas que podem precisar de reindexação, execute a consulta a seguir.

```
select tbl as tbl_id, stv_tbl_perm.name as table_name,
col, interleaved_skew, last_reindex
from svv_interleaved_columns, stv_tbl_perm
where svv_interleaved_columns.tbl = stv_tbl_perm.id
and interleaved_skew is not null;

 tbl_id | table_name | col | interleaved_skew | last_reindex
--------+------------+-----+------------------+---------------------
 100068 | lineorder  |   0 |             3.65 | 2015-04-22 22:05:45
 100068 | lineorder  |   1 |             2.65 | 2015-04-22 22:05:45
 100072 | customer   |   0 |             1.65 | 2015-04-22 22:05:45
 100072 | lineorder  |   1 |             1.00 | 2015-04-22 22:05:45
(4 rows)
```

SVV_LANGUAGE_PRIVILEGES

Use SVV_LANGUAGE_PRIVILEGES para exibir as permissões de linguagem que são explicitamente concedidas a usuários, funções e grupos no banco de dados atual.

SVV_LANGUAGE_PRIVILEGES permanece visível para os seguintes usuários:

- Superusuários
- Usuários com a permissão ACCESS SYSTEM TABLE

Os outros usuários só podem ver identidades às quais tenham acesso ou sejam proprietários.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
language_name	texto	O nome da linguagem.
privilege_type	texto	O tipo de permissão. O valor possível é USAGE.
identity_id	integer	O ID da identidade. Os valores possíveis são ID do usuário, ID da função ou ID do grupo.
identity_name	texto	O nome da identidade.
identity_type	texto	O tipo da identidade. Os valores possíveis são usuário, função, grupo ou público.
admin_option	booleano	Um valor que indica se o usuário pode conceder permissão a outros usuários e funções. É sempre falso para o tipo de identidade de função e grupo.

Consulta de exemplo

O exemplo a seguir exibe o resultado do SVV_LANGUAGE_PRIVILEGES.

```
SELECT language_name,privilege_type,identity_name,identity_type,admin_option FROM
 svv_language_privileges
WHERE identity_name IN ('role1', 'reguser');

 language_name | privilege_type | identity_name | identity_type | admin_option
---------------+----------------+---------------+---------------+--------------
```

```
exfunc      |     USAGE      |    reguser    |     user      |    False
exfunc      |     USAGE      |     role1     |     role      |    False
plpythonu   |     USAGE      |    reguser    |     user      |    False
```

SVV_MASKING_POLICY

Use SVV_MASKING_POLICY para visualizar todas as políticas de mascaramento criadas no cluster.

Somente superusuários e usuários com a função `sys:secadmin` podem exibir SVV_MASKING_POLICY. Usuários regulares verão 0 linha.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
policy_database	texto	O nome do banco de dados no qual a política de mascaramento foi criada.
policy_name	texto	O nome da política de mascaramento.
input_columns	texto	Os atributos fornecidos na cláusula WITH da instrução CREATE POLICY.
policy_expression	texto	A expressão de mascarame nto usada na política.
policy_modified_by	texto	O nome do usuário que modificou a política pela última vez.
policy_modified_time	timestamp	O time stamp de quando a política foi criada ou modificad a pela última vez.

SVV_ML_MODEL_INFO

O estado das informações sobre o estado atual do modelo de Machine Learning.

SVV_ML_MODEL_INFO é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
database_name	char(128)	O banco de dados do modelo.
schema_name	char(128)	O esquema do modelo.
user_name	char(128)	O proprietário do modelo.
model_name	char(128)	O nome do modelo.
life_cycle	char(20)	O status do ciclo de vida do modelo.
is_refreshable	integer	O estado do modelo se ele é atualizável se as tabelas e colunas originais na consulta de treinamento ainda existirem e o usuário ainda tiver as permissões para elas. Os valores possíveis são: 1 (atualizável) e 0 (não atualizáv el).
model_state	char(128)	O estado atual do modelo.

Consulta de exemplo

A consulta a seguir exibe o estado atual dos modelos de Machine Learning.

```
SELECT schema_name, model_name, model_state
FROM svv_ml_model_info;

 schema_name |        model_name            |             model_state
```

```
-------------+------------------------------+--------------------------------------
 public      | customer_churn_auto_model    | Train Model On SageMaker In Progress
 public      | customer_churn_xgboost_model | Model is Ready
(2 row)
```

SVV_ML_MODEL_PRIVILEGES

Use SVV_ML_MODEL_PRIVILEGES para exibir as permissões do modelo de machine learning que são explicitamente concedidas a usuários, funções e grupos no cluster.

SVV_ML_MODEL_PRIVILEGES permanece visível para os seguintes usuários:

- Superusuários
- Usuários com a permissão ACCESS SYSTEM TABLE

Os outros usuários só podem ver identidades às quais tenham acesso ou sejam proprietários.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
namespace_name	texto	O nome do namespace no qual um modelo de machine learning especificado existe.
model_name	texto	O nome do modelo de machine learning.
model_version	integer	O número da versão do modelo.
privilege_type	texto	O tipo de permissão. O valor possível é EXECUTE.
identity_id	integer	O ID da identidade. Os valores possíveis são ID do usuário, ID da função ou ID do grupo.
identity_name	texto	O nome da identidade.

Nome da coluna	Tipo de dados	Descrição
identity_type	texto	O tipo da identidade. Os valores possíveis são usuário, função, grupo ou público.
admin_option	booleano	Um valor que indica se o usuário pode conceder permissão a outros usuários e funções. É sempre falso para o tipo de identidade de função e grupo.

Consulta de exemplo

O exemplo a seguir exibe o resultado do SVV_ML_MODEL_PRIVILEGES.

```
SELECT
 namespace_name,model_name,model_version,privilege_type,identity_name,identity_type,admin_optio
 FROM svv_ml_model_privileges
WHERE model_name = 'test_model';

 namespace_name | model_name | model_version | privilege_type |  identity_name |
 identity_type | admin_option
----------------+------------+---------------+----------------+----------------
+---------------+--------------
      public    | test_model |       1       |    EXECUTE     |     reguser    |
 user      |    False
      public    | test_model |       1       |    EXECUTE     |     role1      |
 role      |    False
```

SVV_MV_DEPENDENCY

A tabela SVV_MV_DEPENDENCY mostra as dependências de visões materializadas em outras visões materializadas no Amazon Redshift.

Para obter mais informações sobre visões materializadas, consulte Visões materializadas no Amazon Redshift.

SVV_MV_DEPENDENCY é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
database_name	char(128)	O banco de dados que contém a visualização materializada especificada.
schema_name	char(128)	O esquema da visão materializada.
name	char(128)	O nome da visualização materializada.
dependent_database_name	char(128)	O banco de dados de visão materializada do qual essa visão materializada depende.
dependent_schema_name	char(128)	O esquema de visualização materializada do qual essa visualização materializada depende.
dependent_name	char(128)	O nome da visualização materializada da qual essa visualização materializada depende.

Consulta de exemplo

A consulta a seguir retorna uma linha de saída que indica que a visualização materializada `mv_over_foo` usa a visualização materializada `mv_foo` em sua definição como uma dependência.

```
CREATE SCHEMA test_ivm_setup;
CREATE TABLE test_ivm_setup.foo(a INT);
CREATE MATERIALIZED VIEW test_ivm_setup.mv_foo AS SELECT * FROM test_ivm_setup.foo;
CREATE MATERIALIZED VIEW test_ivm_setup.mv_over_foo AS SELECT * FROM
 test_ivm_setup.mv_foo;

SELECT * FROM svv_mv_dependency;

 database_name | schema_name          | name        | dependent_database_name |
 dependent_schema_name     | dependent_name
```

```
--------------+----------------------+-------------+-------------------------
+---------------------------+----------
 dev           | test_ivm_setup       | mv_over_foo |                      dev |
 test_ivm_setup            | mv_foo
```

SVV_MV_INFO

A tabela SVV_MV_INFO contém uma linha para cada visão materializada, se os dados estão ou não obsoletos e informações de estado.

Para obter mais informações sobre visões materializadas, consulte Visões materializadas no Amazon Redshift.

SVV_MV_INFO é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
database_name	char(128)	O banco de dados que contém a visão materiali zada.
schema_name	char(128)	O esquema do banco de dados.
user_name	char(128)	O usuário que possui a visualização materiali zada.
name	char(128)	O nome da visão materializada.
is_stale	char(1)	Um `t` indica que a visualização materializada está obsoleta. Uma visualização materializada obsoleta é aquela que não foi atualizada nas tabelas de base atualizadas. Um `f` indica que a visão materializada está atualizada com as alterações da tabela base. Um valor u indica uma desatualização desconhecida, por exemplo, no caso de MVs em tabelas externas simples. É possível que essas informações não sejam

Nome da coluna	Tipo de dados	Descrição
		precisas se uma atualização não tiver sido executada desde a última reinicialização.
estado	integer	O estado da visualização materializada da seguinte forma: • 0 - A visão materializada é totalmente recomputada quando atualizada. • 1 - A visão materializada é incremental. • 101 - A visão materializada não pode ser atualizada devido a uma coluna eliminada. Essa restrição é aplicável mesmo que a coluna não seja usada na visualização materializada. • 102 - A visão materializada não pode ser atualizada devido a um tipo de coluna alterado. Essa restrição é aplicável mesmo que a coluna não seja usada na visualização materializada. • 103 - A visão materializada não pode ser atualizada devido a uma tabela renomeada. • 104 - A visão materializada não pode ser atualizada devido a uma coluna renomeada. Essa restrição é aplicável mesmo que a coluna não seja usada na visualização materializada. • 105 - A visão materializada não pode ser atualizada devido a um esquema renomeado.
autorewrite	char(1)	Um `t` indica que a visualização materializada é elegível para reescrita automática das consultas.
autorefresh	char(1)	Um `t` indica que a visualização materializada pode ser atualizada automaticamente.

Consulta de exemplo

Para exibir o estado de todas as visões materializadas, execute a consulta a seguir.

```
select * from svv_mv_info;
```

Essa consulta retorna os seguintes dados de saída de exemplo.

```
database_name |       schema_name       | user_name |   name   |  is_stale | state |
 autorefresh | autorewrite
--------------+-------------------------+-----------+---------+-----------+-------
+-------------+----------------
 dev          | test_ivm_setup          | catch-22  | mv       |   f       |     1 |
       1 |           0
 dev          | test_ivm_setup          | lotr      | old_mv   |   t       |     1 |
       0 |           1
```

SVV_QUERY_INFLIGHT

Use a exibição SVV_QUERY_INFLIGHT para determinar quais consultas estão sendo executadas no banco de dados no momento. Esta exibição une a STV_INFLIGHT e a STL_QUERYTEXT. A exibição SVV_QUERY_INFLIGHT não mostra as consultas executadas apenas nos nós de liderança. Para obter mais informações, consulte Função de apenas nó líder.

SVV_QUERY_INFLIGHT é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> Essa visualização só está disponível ao consultar clusters provisionados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
slice	integer	A fatia onde a consulta está sendo executada.
consultar	integer	ID da consulta. Ele pode ser usado para unir várias outras tabelas e exibições do sistema.
pid	integer	ID do processo. Todas as consultas em uma sessão são executadas no mesmo processo, portanto esse valor permanece constante se você executa uma série de consultas na mesma sessão. Use esta coluna para unir a tabela STL_ERROR.
starttime	timestamp	O horário do início da consulta.
suspended	integer	Indica se a consulta está suspensa: **0** = falso; **1** = verdadeiro.
texto	character(200)	O texto da consulta, em incrementos de 200 caracteres.
sequence	integer	O número de sequência para os segmentos das instruções da consulta.

Consultas de exemplo

Os dados de saída do exemplo abaixo mostram duas consultas em execução no momento, a própria consulta SVV_QUERY_INFLIGHT e a consulta 428, que é dividida em três linhas na tabela. (As colunas starttime e statement estão truncadas neste exemplo).

```
select slice, query, pid, starttime, suspended, trim(text) as statement, sequence
from svv_query_inflight
order by query, sequence;

slice|query| pid  |      starttime       |suspended| statement | sequence
```

```
-----+-----+------+---------------------+---------+-----------+---------
1012 | 428 | 1658 | 2012-04-10 13:53:... |       0 | select ...|    0
1012 | 428 | 1658 | 2012-04-10 13:53:... |       0 | enueid ...|    1
1012 | 428 | 1658 | 2012-04-10 13:53:... |       0 | atname,...|    2
1012 | 429 | 1608 | 2012-04-10 13:53:... |       0 | select ...|    0
(4 rows)
```

SVV_QUERY_STATE

Use a exibição SVV_QUERY_STATE para visualizar as informações sobre o tempo de execução de consultas em execução no momento.

A exibição SVV_QUERY_STATE contém um subconjunto de dados da tabela STV_EXEC_STATE.

SVV_QUERY_STATE é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

> **Note**
>
> Essa visualização só está disponível ao consultar clusters provisionados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. Ele pode ser usado para unir várias outras tabelas e exibições do sistema.

Nome da coluna	Tipo de dados	Descrição
seg	integer	O número do segmento de consulta que está em execução. Uma consulta consiste em vários segmentos e cada segmento consiste em uma ou mais etapas. Os segmentos de uma consulta podem ser executados em paralelo. Cada segmento executa em um único processo.
etapa	integer	O número da etapa de consulta que está em execução. Uma etapa é a menor unidade do tempo de execução de uma consulta. Cada etapa representa uma unidade de trabalho distinta, como fazer a varredura de uma tabela, retornar resultados ou classificar dados.
maxtime	interval	O tempo máximo (em microssegundos) para executar esta etapa.
avgtime	interval	O tempo médio (em microssegundos) para executar esta etapa.
rows	bigint	O número de linhas produzidas pela etapa que está em execução.
bytes	bigint	O número de bytes produzidos pela etapa que está em execução.
cpu	bigint	Para uso interno.
memory	bigint	Para uso interno.
rate_row	double precision	A taxa de linhas por segundo desde o início da consulta, calculada pela soma das linhas e divisão pelo número de segundos, desde quando a consulta começou até a hora atual.
rate_byte	double precision	A taxa de bytes por segundo desde o início da consulta, calculada pela soma dos bytes e divisão pelo número de segundos, desde quando a consulta começou até a hora atual.
rótulo	character(25)	O rótulo da consulta: um nome para a etapa, como `scan` ou `sort`.

Nome da coluna	Tipo de dados	Descrição
is_diskbased	character(1)	Indica se esta etapa da consulta é executada como uma operação em disco: true (**t**, verdadeiro) ou false (**f**, falso). Somente algumas etapas, como hash, classificação e etapas de agregação podem ir para o disco. Muitos tipos de etapas são sempre realizados na memória.
workmem	bigint	A quantidade de memória de trabalho (em bytes) atribuída à etapa da consulta.
num_parts	integer	O número de partições em que a tabela hash é particionada durante uma etapa de hash. Um número positivo nessa coluna não significa que a etapa de hash é executada como uma operação em disco. Verifique o valor na coluna IS_DISKBASED para ver se a etapa de hash foi executada em disco.
is_rrscan	character(1)	O valor true (**t**) indica que a varredura restrita ao intervalo foi utilizada na etapa. O padrão é falso (**f**).
is_delayed_scan	character(1)	O valor true (**t**) indica que a varredura com atraso foi utilizada na etapa. O padrão é falso (**f**).

Consultas de exemplo

Determinação do tempo de processamento de uma consulta por etapa

A consulta a seguir mostra quanto tempo cada etapa da consulta com a ID de consulta 279 levou para ser executada e quantas linhas de dados o Amazon Redshift processou:

```
select query, seg, step, maxtime, avgtime, rows, label
from svv_query_state
where query = 279
order by query, seg, step;
```

Esta consulta recupera as informações do processamento da consulta 279, como mostram os seguintes dados de saída de exemplo:

```
query |   seg   | step | maxtime | avgtime |  rows   | label
------+---------+------+---------+---------+---------+------------------
  279 |       3 |    0 | 1658054 | 1645711 | 1405360 | scan
  279 |       3 |    1 | 1658072 | 1645809 |       0 | project
  279 |       3 |    2 | 1658074 | 1645812 | 1405434 | insert
  279 |       3 |    3 | 1658080 | 1645816 | 1405437 | distribute
  279 |       4 |    0 | 1677443 | 1666189 | 1268431 | scan
  279 |       4 |    1 | 1677446 | 1666192 | 1268434 | insert
  279 |       4 |    2 | 1677451 | 1666195 |       0 | aggr
(7 rows)
```

Determinar se há consultas ativas em execução no disco

A consulta a seguir mostra se há consultas ativas em execução no disco no momento:

```
select query, label, is_diskbased from svv_query_state
where is_diskbased = 't';
```

Esta saída de exemplo mostra as consultas ativas em execução no disco no momento:

```
 query | label        | is_diskbased
-------+--------------+--------------
1025   | hash tbl=142 |      t
(1 row)
```

SVV_REDSHIFT_COLUMNS

Use SVV_REDSHIFT_COLUMNS para exibir uma lista de todas as colunas às quais um usuário tem acesso. Este conjunto de colunas inclui as colunas no cluster e as colunas de datashares fornecidos por clusters remotos.

SVV_REDSHIFT_COLUMNS é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Para obter mais informações sobre as práticas recomendadas ao consultar tabelas e visualizações do sistema, consulte Consulta de tabelas do sistema.

> **Note**
>
> Use o comando SHOW COLUMNS para descobrir colunas. SHOW COLUMNS funciona de forma consistente em contextos locais, de unidade de compartilhamento de dados e de catálogos externos e é atualizado à medida que novos recursos são lançados. Para obter mais informações, consulte Práticas recomendadas para descobrir metadados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
database_name	varchar(128)	O nome do banco de dados no qual a tabela contendo as colunas existe.
schema_name	varchar(128)	O nome do esquema para a tabela.
table_name	varchar(128)	O nome da tabela.
column_name	varchar(128)	O nome da coluna.
ordinal_position	integer	A posição da coluna na tabela.
data_type	varchar(32)	O tipo de dados da coluna.
column_default	varchar(4000)	O valor padrão da coluna.
is_nullable	varchar(3)	Um valor que define se uma coluna é anulável. Os valores possíveis são `yes`, `no` e " " (uma string vazia que não representa nenhuma informação).
encoding	varchar(128)	O tipo de codificação da coluna.

Nome da coluna	Tipo de dados	Descrição
distkey	booleano	Um valor que é true se essa coluna for a chave de distribui ção da tabela e false caso contrário.
sortkey	integer	Um valor que especifica a ordem da coluna na chave de classificação. Se a tabela usar uma chave de classificação composta, todas as colunas que fizerem parte da chave de classific ação terão um valor positivo que indicará a posição da coluna na chave de classific ação. Se a tabela usar uma chave de classificação intercalada, cada coluna que fizer parte da chave de classificação terá um valor alternadamente positivo ou negativo. Aqui, o valor absoluto indica a posição da coluna na chave de classific ação. Se `sortkey` for 0, a coluna não é parte de uma chave de classificação.
column_acl	varchar(128)	Uma string que define as permissões para o usuário ou grupo de usuários especific ado para a coluna.

Nome da coluna	Tipo de dados	Descrição
remarks	varchar(256)	Observações.

Consulta de exemplo

O exemplo a seguir retorna a saída de SVV_REDSHIFT_COLUMNS.

```
SELECT *
FROM svv_redshift_columns
WHERE database_name = 'tickit_db'
    AND TABLE_NAME = 'tickit_sales_redshift'
ORDER BY COLUMN_NAME,
    TABLE_NAME,
    database_name
LIMIT 5;

database_name | schema_name |       table_name      | column_name | ordinal_position |
 data_type | column_default | is_nullable | encoding | distkey | sortkey | column_acl
 | remarks
--------------+-------------+-----------------------+-------------+------------------
+-----------+----------------+-------------+----------+---------+---------
+-------------+--------
   tickit_db  |   public    | tickit_sales_redshift |   buyerid   |        4         |
 integer  |                |      NO     |   az64   |  False  |    0    |             |
   tickit_db  |   public    | tickit_sales_redshift |  commission |        9         |
 numeric  |      (8,2)     |     YES     |   az64   |  False  |    0    |             |
   tickit_db  |   public    | tickit_sales_redshift |    dateid   |        6         |
 smallint |                |      NO     |   none   |  False  |    1    |             |
   tickit_db  |   public    | tickit_sales_redshift |   eventid   |        5         |
 integer  |                |      NO     |   az64   |  False  |    0    |       |
   tickit_db  |   public    | tickit_sales_redshift |   listid    |        2         |
 integer  |                |      NO     |   az64   |  True   |    0    |             |
```

SVV_REDSHIFT_DATABASES

Use SVV_REDSHIFT_DATABASES para exibir uma lista de todos os bancos de dados aos quais um usuário tem acesso. Isso inclui os bancos de dados no cluster e os bancos de dados criados a partir de datashares fornecidos por clusters remotos.

SVV_REDSHIFT_DATABASES é visível a todos os usuários por padrão. Para controlar o acesso aos metadados do banco de dados, habilite a segurança de metadados para seu cluster provisionado ou grupo de trabalho sem servidor. A segurança de metadados permite que você separe as permissões de visualização dos metadados do objeto por usuários e perfis. Para obter mais informações, consulte Segurança de metadados.

Para obter mais informações sobre as práticas recomendadas ao consultar tabelas e visualizações do sistema, consulte Consulta de tabelas do sistema.

> **Note**
>
> Use o comando SHOW DATABASES para descobrir bancos de dados. SHOW DATABASES funciona de forma consistente em contextos locais, de unidade de compartilhamento de dados e de catálogos externos e é atualizado à medida que novos recursos são lançados. Para obter mais informações, consulte Práticas recomendadas para descobrir metadados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
database_name	varchar(128)	O nome do banco de dados.
database_owner	integer	O ID do usuário proprietário do banco de dados.
database_type	varchar(32)	O tipo do banco de dados. Os tipos possíveis são bancos de dados locais ou compartil hados.
database_acl	varchar(128)	Essas informações são somente para uso interno.
database_options	varchar(128)	As propriedades do banco de dados.
database_isolation_level	varchar(128)	O nível de isolamento do banco de dados. Os valores

Nome da coluna	Tipo de dados	Descrição
		possíveis incluem: `Snapshot Isolation` e `Serializable`.

Consulta de exemplo

O exemplo a seguir retorna a saída para SVV_REDSHIFT_DATABASES.

```
select database_name, database_owner, database_type, database_options,
 database_isolation_level
from  svv_redshift_databases;


database_name | database_owner | database_type | database_options |
 database_isolation_level
--------------+----------------+---------------+------------------+-----------------
   dev        |  1             | local         | NULL             | Serializable
```

SVV_REDSHIFT_FUNCTIONS

Use SVV_REDSHIFT_FUNCTIONS para exibir uma lista de todas as funções às quais um usuário tem acesso. Este conjunto de funções inclui as funções no cluster e as funções de datashares fornecidos por clusters remotos.

SVV_REDSHIFT_FUNCTIONS permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Para obter mais informações sobre as práticas recomendadas ao consultar tabelas e visualizações do sistema, consulte Consulta de tabelas do sistema.

> **Note**
>
> Use o comando SHOW FUNCTIONS para descobrir funções. SHOW FUNCTIONS funciona de forma consistente em contextos locais, de unidade de compartilhamento de dados e de catálogos externos e é atualizado à medida que novos recursos são lançados. Para obter mais informações, consulte Práticas recomendadas para descobrir metadados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
database_name	varchar(128)	O nome do banco de dados no qual o cluster que tem essas funções existe.
schema_name	varchar(128)	O nome do esquema que especifica uma determinada função.
function_name	varchar(128)	O nome de uma função especificada.
function_type	varchar(128)	O tipo de função. Os valores possíveis são funções regulares, funções agregadas e procedimentos armazenados.
argument_type	varchar(512)	Uma string que representa o tipo de argumento de entrada de uma função.
result_type	varchar(128)	O tipo de dados do valor de retorno de uma função.

Consulta de exemplo

O exemplo a seguir retorna a saída para SVV_REDSHIFT_FUNCTIONS.

```
SELECT *
FROM svv_redshift_functions
WHERE database_name = 'tickit_db'
    AND SCHEMA_NAME = 'public'
ORDER BY function_name
LIMIT 5;
```

```
database_name | schema_name |      function_name    |  function_type   |
 argument_type  | result_type
--------------+-------------+-----------------------+------------------
+------------------+-------------
   tickit_db  |    public   |     shared_function   | REGULAR FUNCTION | integer,
 integer |   integer
```

SVV_REDSHIFT_SCHEMA_QUOTA

Exibe a cota e o uso atual do disco para cada esquema em um banco de dados.

SVV_REDSHIFT_SCHEMA_QUOTA está visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Essa visualização está disponível ao consultar clusters provisionados ou grupos de trabalho do Redshift sem servidor.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
database_name	character(128)	O banco de dados que contém o esquema.
schema_name	character(128)	O nome do esquema.
schema_owner	integer	O ID de usuário interno do proprietá rio do esquema.
quota	integer	A quantidade de espaço em disco (em MB) que o esquema pode usar.
disk_usage	integer	O espaço em disco (em MB) atualmente usado pelo esquema.

Consulta de exemplo

O exemplo a seguir exibe a cota e o uso atual do disco para o esquema denominado `sales_schema`.

```
SELECT TRIM(SCHEMA_NAME) "schema_name", QUOTA, disk_usage FROM
 svv_redshift_schema_quota
WHERE SCHEMA_NAME = 'sales_schema';


schema_name   | quota | disk_usage
--------------+-------+-----------
sales_schema  | 2048  | 30
```

SVV_REDSHIFT_SCHEMAS

Use SVV_REDSHIFT_SCHEMAS para exibir uma lista de todos os esquemas aos quais um usuário tem acesso. Este conjunto de esquemas inclui os esquemas no cluster e os esquemas de datashares fornecidos por clusters remotos.

SVV_REDSHIFT_SCHEMAS permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Para obter mais informações sobre as práticas recomendadas ao consultar tabelas e visualizações do sistema, consulte Consulta de tabelas do sistema.

> **Note**
>
> Use o comando SHOW SCHEMAS para descobrir esquemas. SHOW SCHEMAS funciona de forma consistente em contextos locais, de unidade de compartilhamento de dados e de catálogos externos e é atualizado à medida que novos recursos são lançados. Para obter mais informações, consulte Práticas recomendadas para descobrir metadados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
database_name	varchar(128)	O nome do banco de dados no qual um esquema especific ado existe.

Nome da coluna	Tipo de dados	Descrição
schema_name	varchar(128)	O nome do esquema ou namespace.
schema_owner	integer	O ID de usuário interno do proprietário do esquema.
schema_type	varchar(16)	O tipo do esquema. Os valores possíveis são esquemas compartilhados e locais.
schema_acl	varchar(128)	A string que define as permissões para o usuário ou o grupo de usuários especific ado para o esquema.
schema_option	varchar(128)	As opções do esquema.

Consulta de exemplo

O exemplo a seguir retorna a saída de SVV_REDSHIFT_SCHEMAS.

```
SELECT *
FROM svv_redshift_schemas
WHERE database_name = 'tickit_db'
ORDER BY database_name,
    SCHEMA_NAME;

database_name |    schema_name     | schema_owner | schema_type | schema_acl |
 schema_option
--------------+--------------------+--------------+-------------+------------
+---------------
   tickit_db  |       public       |       1      |    shared   |            |
```

SVV_REDSHIFT_TABLES

Use SVV_REDSHIFT_TABLES para exibir uma lista de todas as tabelas às quais um usuário tem acesso. Esse conjunto de tabelas inclui as tabelas no cluster e as tabelas de datashares fornecidos por clusters remotos.

SVV_REDSHIFT_TABLES permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Para obter mais informações sobre as práticas recomendadas ao consultar tabelas e visualizações do sistema, consulte Consulta de tabelas do sistema.

> **Note**
>
> Use o comando SHOW TABLES para descobrir tabelas. SHOW TABLES funciona de forma consistente em contextos locais, de unidade de compartilhamento de dados e de catálogos externos e é atualizado à medida que novos recursos são lançados. Para obter mais informações, consulte Práticas recomendadas para descobrir metadados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
database_name	varchar(128)	O nome do banco de dados no qual uma tabela especific ada existe.
schema_name	varchar(128)	O nome do esquema para a tabela.
table_name	varchar(128)	O nome da tabela.
table_type	varchar(128)	O tipo da tabela. Os valores possíveis são visualizações e tabelas.

Nome da coluna	Tipo de dados	Descrição
table_acl	varchar(128)	A string que define as permissões para o usuário ou o grupo de usuários especific ado para a tabela.
remarks	varchar(128)	Observações.
table_owner	varchar(128)	O proprietário da tabela.

Consulta de exemplo

O exemplo a seguir retorna a saída de SVV_REDSHIFT_TABLES.

```
SELECT *
FROM svv_redshift_tables
WHERE database_name = 'tickit_db' AND TABLE_NAME LIKE 'tickit_%'
ORDER BY database_name,
TABLE_NAME;

database_name | schema_name |         table_name       | table_type | table_acl |
 remarks | table_owner
--------------+-------------+--------------------------+------------+-----------
+---------+-----------
   tickit_db  |    public   | tickit_category_redshift |    TABLE   |           |
   +
   tickit_db  |    public   |   tickit_date_redshift   |    TABLE   |           |
   +
   tickit_db  |    public   |   tickit_event_redshift  |    TABLE   |           |
   +
   tickit_db  |    public   |  tickit_listing_redshift |    TABLE   |           |
   +
   tickit_db  |    public   |   tickit_sales_redshift  |    TABLE   |           |
   +
   tickit_db  |    public   |   tickit_users_redshift  |    TABLE   |           |
   +
   tickit_db  |    public   |   tickit_venue_redshift  |    TABLE   |           |
```

Se o valor table_acl for nulo, nenhum privilégio de acesso foi concedido explicitamente à tabela correspondente.

SVV_RELATION_PRIVILEGES

Use SVV_RELATION_PRIVILEGES para exibir as permissões de relação (tabelas e exibições) que são explicitamente concedidas a usuários, funções e grupos no banco de dados atual.

SVV_RELATION_PRIVILEGES permanece visível para os seguintes usuários:

- Superusuários
- Usuários com a permissão SYSLOG ACCESS UNRESTRICTED

Os outros usuários só podem ver identidades às quais tenham acesso ou sejam proprietários. Para obter mais informações sobre visibilidade de dados, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Para obter mais informações sobre as práticas recomendadas ao consultar tabelas e visualizações do sistema, consulte Consulta de tabelas do sistema.

> **Note**
>
> Use o comando SHOW GRANTS para descobrir permissões. SHOW GRANTS funciona de forma consistente em contextos locais, de unidade de compartilhamento de dados e de catálogos externos e é atualizado à medida que novos recursos são lançados. Para obter mais informações, consulte Práticas recomendadas para descobrir metadados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
namespace_name	texto	O nome do namespace no qual uma relação especificada existe.
relation_name	texto	O nome da relação.
privilege_type	texto	O tipo de permissão. Os valores possíveis são INSERT, SELECT, UPDATE, DELETE, REFERENCES ou DROP.

Nome da coluna	Tipo de dados	Descrição
identity_id	integer	O ID da identidade. Os valores possíveis são ID do usuário, ID da função ou ID do grupo.
identity_name	texto	O nome da identidade.
identity_type	texto	O tipo da identidade. Os valores possíveis são usuário, função, grupo ou público.
admin_option	booleano	Um valor que indica se o usuário pode conceder permissão a outros usuários e funções. É sempre falso para o tipo de identidade de função e grupo.

Consulta de exemplo

O exemplo a seguir exibe o resultado do SVV_RELATION_PRIVILEGES.

```
SELECT
 namespace_name,relation_name,privilege_type,identity_name,identity_type,admin_option
 FROM svv_relation_privileges
WHERE relation_name = 'orders' AND privilege_type = 'SELECT';

 namespace_name | relation_name | privilege_type |  identity_name | identity_type |
 admin_option
----------------+---------------+----------------+----------------+---------------
+--------------
     public     |    orders     |     SELECT     |    reguser     |     user      |
 False
     public     |    orders     |     SELECT     |     role1      |     role      |
 False
```

SVV_RLS_APPLIED_POLICY

Use SVV_RLS_APPLIED_POLICY para monitorar a aplicação de políticas RLS em consultas que fazem referência a relações protegidas por RLS.

SVV_RLS_APPLIED_POLICY permanece visível para os seguintes usuários:

- Superusuários
- Usuários com a função `sys:operator`
- Usuários com a permissão ACCESS SYSTEM TABLE

Observe que sys:secadmin não recebe essa permissão do sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
username	texto	O nome do usuário que executou a consulta.
consultar	integer	A ID da consulta.
xid	long	O contexto da transação.
pid	integer	O processo líder que executa a consulta.
recordtime	horário	A hora quando a consulta foi gravada.
command	char(1)	O comando para o qual a política de RLS foi aplicada. Os valores possíveis são `k` para desconhecido, `s` para selecionar, `u` para atualizar, `i` para inserir, `y` para utilitário e `d` para excluir.
datname	texto	O nome do banco de dados da relação à qual a política de segurança no nível da linha está anexada.
relschema	texto	O nome do esquema da relação ao qual a política de segurança no nível da linha está anexada.
relname	texto	O nome da relação à qual a política de segurança no nível da linha está anexada.
polname	texto	O nome da relação da política de segurança no nível da linha que está anexada à relação.
poldefault	char(1)	A configuração padrão da política de segurança no nível da linha que está anexada à relação. Os valores possíveis são `f` para falso

Nome da coluna	Tipo de dados	Descrição
		se a política false padrão tiver sido aplicada e `t` para verdadeiro se a política true padrão tiver sido aplicada.

Consulta de exemplo

O exemplo a seguir exibe o resultado de SVV_RLS_APPLIED_POLICY. Para consultar a SVV_RLS_APPLIED_POLICY, você deve ter a permissão ACCESS SYSTEM TABLE.

```
-- Check what RLS policies were applied to the run query.
SELECT username, command, datname, relschema, relname, polname, poldefault
FROM svv_rls_applied_policy
WHERE datname = CURRENT_DATABASE() AND query = PG_LAST_QUERY_ID();

 username | command |  datname  | relschema |          relname         |      polname
  | poldefault
----------+---------+-----------+-----------+--------------------------
+-----------------+------------
   molly  |    s    | tickit_db |   public  | tickit_category_redshift |
 policy_concerts |
```

SVV_RLS_ATTACHED_POLICY

Use SVV_RLS_ATTACHED_POLICY para exibir uma lista de todas as relações e usuários que têm uma ou mais políticas de segurança no nível da linha anexadas ao banco de dados conectado no momento.

Somente usuários com a função sys:secadmin podem consultar essa exibição.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
relschema	texto	O nome do esquema da relação ao qual a política de segurança no nível da linha está anexada.

Nome da coluna	Tipo de dados	Descrição
relname	texto	O nome da relação à qual a política de segurança no nível da linha está anexada.
relkind	texto	O tipo de objeto, como tabela.
polname	texto	O nome da relação da política de segurança no nível da linha que está anexada à relação.
grantor	texto	O nome do usuário que anexou essa política.
grantee	texto	O nome do usuário ou função em que essa política foi anexada.
granteekind	texto	O tipo de favorecido. Os valores possíveis são usuário ou função.
is_pol_on	booleano	O parâmetro que indica se uma política de segurança no nível da linha está ativada ou desativada em uma tabela. Os valores possíveis são true e false.
is_rls_on	booleano	O parâmetro que indica se uma segurança no nível da linha está ativada ou desativada em uma tabela. Os valores possíveis são true e false.
rls_conjunction_type	character (3)	O parâmetro que indica se a relação combina políticas RLS com `and` ou `or`.

Consulta de exemplo

O exemplo a seguir exibe o resultado de SVV_RLS_ATTACHED_POLICY.

```
--Inspect the policy in SVV_RLS_ATTACHED_POLICY
SELECT * FROM svv_rls_attached_policy;

 relschema |        relname           | relkind |     polname     | grantor | grantee
 | granteekind | is_pol_on | is_rls_on | rls_conjuntion_type
-----------+--------------------------+---------+-----------------+---------+----------
+-------------+-----------+-----------+--------------------
```

```
public    | tickit_category_redshift |  table  | policy_concerts |   bob   |  analyst
|    role     |    True   |    True   |      and
public    | tickit_category_redshift |  table  | policy_concerts |   bob   |  dbadmin
|    role     |    True   |    True   |      and
```

SVV_RLS_POLICY

Use SVV_RLS_POLICY para visualizar uma lista de todas as políticas de segurança no nível da linha criadas no cluster do Amazon Redshift.

SVV_RLS_RLS_RLS_RLS_POLICY permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Para obter mais informações sobre as práticas recomendadas ao consultar tabelas e visualizações do sistema, consulte Consulta de tabelas do sistema.

> **Note**
>
> Use o comando SHOW POLICIES para descobrir políticas. SHOW POLICIES funciona de forma consistente em contextos locais, de unidade de compartilhamento de dados e de catálogos externos e é atualizado à medida que novos recursos são lançados. Para obter mais informações, consulte Práticas recomendadas para descobrir metadados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
poldb	texto	O nome do banco de dados no qual a política de segurança no nível da linha é criada.
polname	texto	O nome da política de segurança no nível da linha.
polalias	texto	O alias da tabela usado na definição da política.
polatts	texto	Os atributos fornecidos para a definição da política.

Nome da coluna	Tipo de dados	Descrição
polqual	texto	A condição da política fornecida na cláusula USING da instrução CREATE POLICY.
polenabled	booleano	Se a política for ativada globalmente.
polmodifiedby	texto	O nome do usuário que criou ou modificou a política mais recentemente.
polmodifiedtime	timestamp	O carimbo de data/hora de quando a política é criada ou modificada pela última vez.

Consulta de exemplo

O exemplo a seguir exibe o resultado de SVV_RLS_POLICY.

```
-- Create some policies.
CREATE RLS POLICY pol1 WITH (a int) AS t USING ( t.a IS NOT NULL );
CREATE RLS POLICY pol2 WITH (c varchar(10)) AS t USING ( c LIKE '%public%');

-- Inspect the policy in SVV_RLS_POLICY
SELECT * FROM svv_rls_policy;

 poldb | polname | polalias |                     polatts                      |
         polqual                | polenabled | polmodifiedby |   polmodifiedtime
-------+---------+----------+--------------------------------------------------
+--------------------------------------+------------+--------------
+---------------------
 my_db | pol1    | t        | [{"colname":"a","type":"integer"}]               |
 "t"."a" IS NOT NULL                   | t          | policy_admin  | 2022-02-11
 14:40:49
 my_db | pol2    | t        | [{"colname":"c","type":"character varying(10)"}] |
 "t"."c" LIKE CAST('%public%' AS TEXT) | t          | policy_admin  | 2022-02-11
 14:41:28
```

SVV_RLS_RELATION

Use SVV_RLS_RELATION para exibir uma lista de todas as relações protegidas por RLS.

SYSLOG ACCESS permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
datname	texto	O nome do banco de dados que contém a relação.
relschema	texto	O nome do esquema que contém a relação.
relname	texto	O nome da relação.
relkind	texto	O tipo da relação, como tabelas ou exibições.
is_rls_on	booleano	O parâmetro que indica se a relação é protegida por RLS.
is_rls_datashare_on	booleano	O parâmetro que indica se a relação é protegida por RLS sobre unidades de compartilhamento de dados.
rls_conjunction_type	character(3)	O parâmetro que indica se a relação combina políticas RLS com `and` ou`or`.
rls_datashare_conjunction_type	character(3)	O parâmetro que indica se a relação combina políticas RLS com `and` ou `or` por meio das unidades de compartilhamento de dados.

Consulta de exemplo

O exemplo a seguir exibe o resultado de SVV_RLS_RELATION.

```
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY ON FOR DATASHARES;


--Inspect RLS state on the relations using SVV_RLS_RELATION.
SELECT datname, relschema, relname, relkind, is_rls_on, is_rls_datashare_on FROM
 svv_rls_relation ORDER BY relname;
```

```
  datname  | relschema |        relname           | relkind | is_rls_on |
 is_rls_datashare_on | rls_conjunction_type | rls_datashare_conjunction_type
-----------+-----------+--------------------------+---------+-----------
+--------------------+----------------------+--------------------------------
 tickit_db |   public  | tickit_category_redshift |  table  |      t    |           t
       |          and         |              and
(1 row)
```

SVV_ROLE_GRANTS

Use SVV_ROLE_GRANTS para exibir uma lista de funções que são explicitamente concedidas no cluster.

SVV_ROLE_GRANTS permanece visível para os seguintes usuários:

- Superusuários
- Usuários com a permissão ACCESS SYSTEM TABLE

Os outros usuários só podem ver identidades às quais tenham acesso ou sejam proprietários.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
role_id	integer	O ID da função.
role_name	texto	O nome da função.
granted_role_id	integer	O ID da função concedida.
granted_role_name	texto	O nome da função concedida.

Consulta de exemplo

O exemplo a seguir retorna a saída de SVV_ROLE_GRANTS.

```
GRANT ROLE role1 TO ROLE role2;
GRANT ROLE role2 TO ROLE role3;
```

```
SELECT role_name, granted_role_name FROM svv_role_grants;

 role_name |  granted_role_name
-----------+--------------------
   role2   |      role1
   role3   |      role2
(2 rows)
```

SVV_ROLES

Use SVV_ROLES para visualizar informações de perfil.

Esta tabela é visível para todos os usuários.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
role_id	integer	O ID da função.
role_name	texto	O nome da função.
role_owner	texto	O nome do proprietário da função.
external_id	texto	O identificador exclusivo da função no provedor de identidades de terceiros.

Consulta de exemplo

O exemplo a seguir retorna a saída de SVV_ROLES.

```
SELECT role_name,role_owner FROM svv_roles WHERE role_name IN ('role1', 'role2');

 role_name | role_owner
-----------+------------
   role1   | superuser
   role2   | superuser
```

SVV_SCHEMA_PRIVILEGES

Use SVV_SCHEMA_PRIVILEGES para exibir as permissões de esquema que são explicitamente concedidas a usuários, funções e grupos no banco de dados atual.

SVV_SCHEMA_PRIVILEGES permanece visível para os seguintes usuários:

- Superusuários
- Usuários com a permissão ACCESS SYSTEM TABLE

Os outros usuários só podem ver identidades às quais tenham acesso ou sejam proprietários.

Para obter mais informações sobre as práticas recomendadas ao consultar tabelas e visualizações do sistema, consulte Consulta de tabelas do sistema.

> **Note**
>
> Use o comando SHOW GRANTS para descobrir permissões. SHOW GRANTS funciona de forma consistente em contextos locais, de unidade de compartilhamento de dados e de catálogos externos e é atualizado à medida que novos recursos são lançados. Para obter mais informações, consulte Práticas recomendadas para descobrir metadados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
namespace_name	texto	O nome do namespace no qual um esquema especificado existe.
privilege_type	texto	O tipo de permissão. Com relação a permissões com um privilege _scope de esquema, os valores possíveis são CREATE, USAGE e ALTER. Quanto a valores de privilege_scope que não são de esquema, os valores possíveis

Nome da coluna	Tipo de dados	Descrição
		incluem qualquer tipo de permissão disponível no escopo da permissão.
identity_id	integer	O ID da identidade. Os valores possíveis são ID do usuário, ID da função ou ID do grupo.
identity_name	texto	O nome da identidade.
identity_type	texto	O tipo da identidade. Os valores possíveis são usuário, função, grupo ou público.
admin_option	booleano	Um valor que indica se o usuário pode conceder permissão a outros usuários e funções. É sempre falso para o tipo de identidade de função e grupo.
privilege_scope	texto	O escopo da permissão especific ada em privilege_type. Os valores possíveis são: • SCHEMA • TABLES • FUNCTIONS Para ter informações sobre as permissões com escopo, acesse Permissões em escopo.

Consulta de exemplo

O exemplo a seguir exibe o resultado de SVV_SCHEMA_PRIVILEGES.

```
SELECT namespace_name,privilege_type,identity_name,identity_type,admin_option FROM
 svv_schema_privileges
WHERE namespace_name = 'test_schema1';

 namespace_name | privilege_type |  identity_name | identity_type | admin_option
----------------+----------------+----------------+---------------+--------------
 test_schema1   |    USAGE       |     reguser    |     user      |   False
 test_schema1   |    USAGE       |     role1      |     role      |   False
```

SVV_SCHEMA_QUOTA_STATE

Exibe a cota e o uso atual do disco para cada esquema.

Os usuários comuns podem ver informações de esquemas para os quais têm permissão USAGE. Os superusuários podem ver informações de todos os esquemas no banco de dados atual.

SVV_SCHEMA_QUOTA_STATE é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> Essa visualização só está disponível ao consultar clusters provisionados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
schema_id	integer	O ID do esquema ou namespace.
schema_name	character (128)	O nome do esquema ou namespace .
schema_owner	integer	O ID de usuário interno do proprietá rio do esquema.
quota	integer	A quantidade de espaço em disco (em MB) que o esquema pode usar.

Nome da coluna	Tipo de dados	Descrição
disk_usage	integer	O espaço em disco (em MB) atualmente usado pelo esquema.
disk_usage_pct	double precision	A porcentagem de espaço em disco usado atualmente pelo esquema fora da cota configurada.

Consulta de exemplo

O exemplo a seguir exibe a cota e o uso atual do disco para o esquema.

```
SELECT TRIM(SCHEMA_NAME) "schema_name", QUOTA, disk_usage, disk_usage_pct FROM
 svv_schema_quota_state
WHERE SCHEMA_NAME = 'sales_schema';
schema_name   | quota | disk_usage | disk_usage_pct
--------------+-------+------------+---------------
sales_schema  | 2048  | 30         | 1.46
(1 row)
```

SVV_SYSTEM_PRIVILEGES

SVV_SYSTEM_PRIVILEGES permanece visível para os seguintes usuários:

- Superusuários
- Usuários com a permissão ACCESS SYSTEM TABLE

Os outros usuários só podem ver identidades às quais tenham acesso ou sejam proprietários.

Para obter mais informações sobre as práticas recomendadas ao consultar tabelas e visualizações do sistema, consulte Consulta de tabelas do sistema.

> **Note**
>
> Use o comando SHOW GRANTS para descobrir permissões. SHOW GRANTS funciona de forma consistente em contextos locais, de unidade de compartilhamento de dados e de

catálogos externos e é atualizado à medida que novos recursos são lançados. Para obter mais informações, consulte Práticas recomendadas para descobrir metadados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
system_privilege	texto	O nome da permissão do sistema.
identity_id	integer	O ID da identidade. Os valores possíveis são ID do usuário ou ID da função.
identity_name	texto	O nome da identidade.
identity_type	texto	O tipo da identidade. Os valores possíveis são usuário ou função.

Consulta de exemplo

O exemplo a seguir exibe o resultado dos parâmetros especificados.

```
SELECT system_privilege,identity_name,identity_type FROM svv_system_privileges
WHERE system_privilege = 'ALTER TABLE' AND identity_name = 'sys:superuser';

 system_privilege | identity_name | identity_type
------------------+---------------+--------------
   ALTER TABLE    | sys:superuser |     role
```

SVV_TABLE_INFO

Mostra informações resumidas de tabelas e visões materializadas no banco de dados atualmente conectado. A visualização filtra as tabelas do sistema e mostra somente as tabelas definidas pelo usuário e as visões materializadas que contêm pelo menos uma linha de dados.

Você pode usar a exibição SVV_TABLE_INFO para diagnosticar e resolver problemas de design de tabelas que podem influenciar a performance de uma consulta. Isso inclui problemas com codificação de compactação, chaves de distribuição, estilo de classificação, distorção de distribuição de dados,

tamanho da tabela e estatísticas. A exibição SVV_TABLE_INFO não retorna nenhuma informações para tabelas vazias.

A visualização SVV_TABLE_INFO resume as informações das seguintes tabelas do sistema e tabelas de catálogo:

- STV_NODE_STORAGE_CAPACITY
- STV_SLICES
- STV_TBL_PERM
- PG_ATTRIBUTE
- PG_CLASS
- PG_DATABASE
- PG_NAMESPACE
- PG_STATISTIC_INDICATOR

SVV_TABLE_INFO é visível somente para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema. Para permitir que um usuário consulte a exibição, conceda ao usuário a permissão SELECT em SVV_TABLE_INFO.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
database	texto	Database name.
schema	texto	Nome do esquema.
table_id	oid	ID da tabela.
table	texto	O nome da tabela.
encoded	texto	O valor que indica se alguma coluna está com a codificação de compactação definida.
diststyle	texto	O estilo de distribuição ou coluna da chave de distribui

Nome da coluna	Tipo de dados	Descrição
		ção, se a chave de distribui ção estiver definida. Os valores possíveis incluem EVEN, KEY(*column*), ALL, AUTO(ALL) , AUTO(EVEN) e AUTO(KEY(*column*)).
sortkey1	texto	A primeira coluna na chave de classificação, se a chave de classificação estiver definida. Os valores possíveis incluem *column*, AUTO(SORTKEY) e AUTO(SORTKEY(*column*)).
max_varchar	integer	O tamanho da maior coluna que usa um tipo de dados VARCHAR.
sortkey1_enc	character(32)	A codificação da compactaç ão da primeira coluna na chave de classificação, se a chave de classificação estiver definida.
sortkey_num	integer	O número de colunas definidas como chaves de classificação.
size	bigint	O tamanho da tabela, em blocos de dados de 1 MB.
pct_used	numeric(10,4)	A porcentagem do espaço disponível que é usado pela tabela.

Nome da coluna	Tipo de dados	Descrição
empty	bigint	Para uso interno. Esta coluna não é mais usada e será removida em uma versão futura.
unsorted	numeric(5,2)	A porcentagem de linhas não classificadas na tabela.
stats_off	numeric(5,2)	O número que indica o nível de obsolescência das estatísticas de uma tabela; 0 para atuais, 100 para obsoletas.
tbl_rows	numeric(38,0)	O número total de linhas na tabela. Este valor inclui as linhas marcadas para exclusão, mas que ainda não foram limpas.
skew_sortkey1	numeric(19,2)	A razão entre o tamanho da maior coluna de chave sem classificação e o tamanho da primeira coluna da chave de classificação, se uma chave de classificação estiver definida. Use esse valor para avaliar a eficácia da chave de classificação.
skew_rows	numeric(19,2)	A razão entre o número de linhas na fatia com o maior número de linhas e o número de linhas na fatia com o menor número de linhas.

Nome da coluna	Tipo de dados	Descrição
estimated_visible_rows	numeric(38,0)	As linhas estimadas na tabela. Esse valor não inclui linhas marcadas para exclusão.

risk_event	texto	Informações de risco sobre uma tabela. O campo é separado em partes:
		`risk_type |xid|timestamp`
		• O `risk_type` , em que 1 indica que um `COPY command with the EXPLICIT_IDS option` foi executado. O Amazon Redshift não verifica mais a exclusividade de colunas IDENTITY na tabela. Para obter mais informações, consulte EXPLICIT_IDS.
		• O ID da transação, `xid`, que introduziu o risco.
		• O `timestamp` quando o comando COPY foi executado.
		O exemplo a seguir mostra os valores no campo.
		`1|1107|2019-06-22 07:16:11.292952`

Nome da coluna	Tipo de dados	Descrição
vacuum_sort_benefit	numérico (12,2)	A melhoria da porcentag em máxima estimada da performance da consulta de verificação quando você executa vacuum sort.
create_time	time stamp sem fuso horário	O timestamp do momento em que a tabela foi criada.

Consultas de exemplo

O exemplo a seguir mostra a codificação, o estilo de distribuição, a classificação e a distorção dos dados para todas as tabelas definidas por usuários no banco de dados. Aqui, "tabela" deve ser colocada entre aspas duplas porque é uma palavra reservada.

```
select "table", encoded, diststyle, sortkey1, skew_sortkey1, skew_rows
from svv_table_info
order by 1;

table          | encoded | diststyle       | sortkey1     | skew_sortkey1 | skew_rows
---------------+---------+-----------------+--------------+---------------+----------
category       | N       | EVEN            |              |               |
date           | N       | ALL             | dateid       |          1.00 |
event          | Y       | KEY(eventid)    | dateid       |          1.00 |      1.02
listing        | Y       | KEY(listid)     | dateid       |          1.00 |      1.01
sales          | Y       | KEY(listid)     | dateid       |          1.00 |      1.02
users          | Y       | KEY(userid)     | userid       |          1.00 |      1.01
venue          | N       | ALL             | venueid      |          1.00 |
(7 rows)
```

SVV_TABLES

Use a exibição SVV_TABLES para visualizar as tabelas em catálogos locais e externos.

SVV_TABLES é visível a todos os usuários por padrão. Para controlar o acesso aos metadados do banco de dados, habilite a segurança de metadados para seu cluster provisionado ou grupo de trabalho sem servidor. A segurança de metadados permite que você separe as permissões de

visualização dos metadados do objeto por usuários e perfis. Para obter mais informações, consulte Segurança de metadados.

Para obter mais informações sobre as práticas recomendadas ao consultar tabelas e visualizações do sistema, consulte Consulta de tabelas do sistema.

> **Note**
>
> Use o comando SHOW TABLES para descobrir tabelas. SHOW TABLES funciona de forma consistente em contextos locais, de unidade de compartilhamento de dados e de catálogos externos e é atualizado à medida que novos recursos são lançados. Para obter mais informações, consulte Práticas recomendadas para descobrir metadados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
table_catalog	texto	O nome do catálogo onde a tabela está localizada.
table_schema	texto	O nome do esquema para a tabela.
table_name	texto	O nome da tabela.
table_type	texto	O tipo da tabela. Os valores possíveis são visualizações, tabelas externas e tabelas base.
remarks	texto	Observações.

SVV_TRANSACTIONS

Registra informações sobre as transações que mantêm bloqueios em tabelas do banco de dados no momento. Use a exibição SVV_TRANSACTIONS para identificar as transações abertas e

os problemas de disputa de bloqueio. Para obter mais informações sobre bloqueios, consulte Gerenciamento de operações de gravação simultâneas e LOCK.

SVV_SVV_TRANSACTIONS permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
txn_owner	texto	O nome do proprietário da transação.
txn_db	texto	O nome do banco de dados associado à transação.
xid	bigint	ID da transação.
pid	integer	O ID do processo associado ao bloqueio.
txn_start	timestamp	O horário de início da transação.
lock_mode	texto	O nome do modo de bloqueio mantido ou solicitado por esse processo. Se `lock_mode` for `ExclusiveLock` e `granted` for true (`t`), esse ID de transação será uma transação aberta.
lockable_object_type	texto	O tipo do objeto que está solicitando ou mantendo o bloqueio, podendo ser `relation`, se for uma tabela ou `transactionid` , se for uma transação.

Nome da coluna	Tipo de dados	Descrição
relation	integer	O ID da tabela (relação) que está solicitando o bloqueio. Esse valor será NULL se `lockable_object_type` for `transactionid` .
granted	booleano	O valor que indica se o bloqueio foi concedido (`t`) ou se está pendente (`f`).

Consultas de exemplo

O comando a seguir mostra todas as transações ativas e os bloqueios solicitados por cada transação.

```
select * from svv_transactions;

 txn_
 lockable_
 owner | txn_db |  xid   |  pid  |         txn_start          |      lock_mode      |
 object_type    | relation | granted
-------+--------+--------+-------+----------------------------+---------------------
+----------------+----------+---------
 root  | dev    | 438484 | 22223 | 2016-03-02 18:42:18.862254 | AccessShareLock     |
 relation       |   100068 | t
 root  | dev    | 438484 | 22223 | 2016-03-02 18:42:18.862254 | ExclusiveLock       |
 transactionid  |          | t
 root  | tickit | 438490 | 22277 | 2016-03-02 18:42:48.084037 | AccessShareLock     |
 relation       |    50860 | t
 root  | tickit | 438490 | 22277 | 2016-03-02 18:42:48.084037 | AccessShareLock     |
 relation       |    52310 | t
 root  | tickit | 438490 | 22277 | 2016-03-02 18:42:48.084037 | ExclusiveLock       |
 transactionid  |          | t
 root  | dev    | 438505 | 22378 | 2016-03-02 18:43:27.611292 | AccessExclusiveLock |
 relation       |   100068 | f
 root  | dev    | 438505 | 22378 | 2016-03-02 18:43:27.611292 | RowExclusiveLock    |
 relation       |    16688 | t
```

```
 root  | dev    | 438505 | 22378 | 2016-03-02 18:43:27.611292 | AccessShareLock     |
 relation       |   100064 | t
 root  | dev    | 438505 | 22378 | 2016-03-02 18:43:27.611292 | AccessExclusiveLock |
 relation       |   100166 | t
 root  | dev    | 438505 | 22378 | 2016-03-02 18:43:27.611292 | AccessExclusiveLock |
 relation       |   100171 | t
 root  | dev    | 438505 | 22378 | 2016-03-02 18:43:27.611292 | AccessExclusiveLock |
 relation       |   100190 | t
 root  | dev    | 438505 | 22378 | 2016-03-02 18:43:27.611292 | ExclusiveLock       |
 transactionid  |          | t
(12 rows)

(12 rows)
```

SVV_USER_GRANTS

Use SVV_USER_GRANTS para exibir uma lista de usuários que têm funções explicitamente concedidas no cluster.

SVV_USER_GRANTS permanece visível para os seguintes usuários:

- Superusuários
- Usuários com a permissão ACCESS SYSTEM TABLE

Os outros usuários só podem ver funções explicitamente concedidas a eles.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	integer	O ID do usuário.
user_name	texto	O nome do usuário.
role_id	integer	O ID da função para a função concedida.
role_name	texto	O nome da função para a função concedida.

Nome da coluna	Tipo de dados	Descrição
admin_option	booleano	Um valor que indica se o usuário pode conceder a função a outros usuários e funções.

Consultas de exemplo

As consultas a seguir concedem funções aos usuários e mostram a lista de usuários que têm funções explicitamente concedidas.

```
GRANT ROLE role1 TO reguser;
GRANT ROLE role2 TO reguser;
GRANT ROLE role1 TO superuser;
GRANT ROLE role2 TO superuser;

SELECT user_name,role_name,admin_option FROM svv_user_grants;

 user_name | role_name | admin_option
-----------+-----------+--------------
 superuser |  role1    | False
 reguser   |  role1    | False
 superuser |  role2    | False
  reguser  |  role2    | False
```

SVV_USER_INFO

Você pode recuperar dados sobre os usuários do banco de dados do Amazon Redshift com a visualização SVV_USER_INFO.

SVV_USER_INFO permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_name	texto	O nome do usuário da função.
user_id	integer	O ID do usuário.
createdb	booleano	Um valor que indica se o usuário tem permissões para criar bancos de dados.
superusuário	booleano	Um valor que indica se o usuário é um superusuário.
catalog_update	booleano	Um valor que indica se o usuário pode atualizar catálogos do sistema.
connection_limit	texto	O número de conexões que o usuário pode abrir.
syslog_access	texto	Um valor que indica se o usuário tem acesso aos logs do sistema. Os dois valores possíveis são `RESTRICTED` e `UNRESTRICTED`. `RESTRICTED` significa que os usuários que não são superusuários podem ver seus próprios registros. `UNRESTRICTED` significa que o usuário que não é superusuário pode ver todos os registros nas exibições e tabelas do sistema para as quais ele tem privilégios `SELECT`.
last_ddl_timestamp	timestamp	O registro de data e hora da última linguagem de definição de dados (DDL) cria uma instrução executada pelo usuário.
session_timeout	integer	O tempo máximo em segundos em que uma sessão permanece inativa ou ociosa antes do tempo limite. 0 indica que nenhum tempo limite está definido. Para obter informações sobre a configuração de tempo limite ocioso ou inativo do cluster, consulte “Cotas e limites no Amazon Redshift” no Guia de gerenciamento de clusters do Amazon Redshift.
external_user_id	texto	O identificador exclusivo do usuário no provedor de identidades de terceiros.

Consultas de exemplo

O comando a seguir recupera informações de usuário de SVV_USER_INFO.

```
SELECT * FROM SVV_USER_INFO;
```

SVV_VACUUM_PROGRESS

Essa exibição retorna uma estimativa de quanto tempo levará para concluir uma operação de limpeza que está em andamento no momento.

SV_VACUUM_PROGRESS é visível somente para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_VACUUM_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Para obter informações sobre SVV_VACUUM_SUMMARY, consulte SVV_VACUUM_SUMMARY.

Para obter informações sobre SVL_VACUUM_PERCENTAGE, consulte SVL_VACUUM_PERCENTAGE.

> **Note**
>
> Essa visualização só está disponível ao consultar clusters provisionados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
table_name	texto	O nome da tabela que está sendo limpada ou da última tabela que foi limpada, se não houver uma operação em andamento.
status	texto	A descrição da atividade que está sendo processada no momento como parte da operação de limpeza:

Nome da coluna	Tipo de dados	Descrição
		• Inicializar • Sort • Merge • Excluir • Select • Failed • Completo • Ignorado • Criar ordem de INTERLEAVED SORTKEY
time_remaining_estimate	texto	O tempo restante estimado para a operação de limpeza atual ser concluída, em minutos e segundos: **`5m 10s`**, por exemplo. O tempo estimado não é retornado até que a limpeza tenha concluído sua primeira operação de classificação. Se não houver uma operação de limpeza em andamento, a última limpeza executada será exibida com **`Completed`** na coluna STATUS mostrando e a coluna TIME_REMAINING_ESTIMATE vazia. Normalmente, a estimativa se torna mais precisa à medida que a limpeza progride.

Consultas de exemplo

As consultas a seguir, executadas com alguns minutos de intervalo entre elas, mostram uma tabela grande chamada SALESNEW sendo limpada.

```
select * from svv_vacuum_progress;

table_name    |           status             | time_remaining_estimate
--------------+------------------------------+-------------------------
salesnew      |  Vacuum: initialize salesnew  |
(1 row)
...
select * from svv_vacuum_progress;
```

```
table_name   |         status         | time_remaining_estimate
-------------+------------------------+-------------------------
salesnew     |  Vacuum salesnew sort  | 33m 21s
(1 row)
```

A consulta a seguir mostra que não há operação de limpeza em andamento no momento. A última tabela a ser limpada foi a tabela SALES.

```
select * from svv_vacuum_progress;

table_name   |  status  | time_remaining_estimate
-------------+----------+-------------------------
  sales      | Complete |
(1 row)
```

SVV_VACUUM_SUMMARY

A exibição SVV_VACUUM_SUMMARY une as tabelas STL_VACUUM, STL_QUERY e STV_TBL_PERM para resumir as informações sobre as operações de limpeza registradas pelo sistema. A exibição retorna uma linha por tabela por transação de limpeza. Ela registra o tempo decorrido da operação, o número de partições de classificação criadas, o número de incrementos de mesclagem necessários e os deltas nas contagens de linha e de bloco, antes e depois da execução da operação.

SVV_VACUUM_SUMMARY é visível somente para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_VACUUM_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Para obter informações sobre SVV_VACUUM_PROGRESS, consulte SVV_VACUUM_PROGRESS.

Para obter informações sobre SVL_VACUUM_PERCENTAGE, consulte SVL_VACUUM_PERCENTAGE.

> **Note**
>
> Essa visualização só está disponível ao consultar clusters provisionados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
table_name	texto	O nome da tabela que foi limpada.
xid	bigint	O ID da transação da operação de VACUUM.
sort_partitions	bigint	O número de partições classificadas criadas durante a fase de classificação da operação de limpeza.
merge_increments	bigint	O número de incrementos de mesclagem necessários para concluir a fase de mesclagem da operação de limpeza.
elapsed_time	bigint	O tempo de execução decorrido na operação de limpeza (em microssegundos).
row_delta	bigint	A diferença no número total de linhas da tabela, antes e depois da limpeza.
sortedrow_delta	bigint	A diferença no número de linhas da tabela classificada, antes e depois da limpeza.
block_delta	integer	A diferença no número de blocos da tabela, antes e depois da limpeza.
max_merge_partitions	integer	Essa coluna é usada para a análise de performance e representa o número máximo de partições que a limpeza pode processar para a tabela por iteração de fase de mesclagem. (A limpeza classifica a região não classificada em uma ou mais partições classificadas. Dependendo do número de colunas na tabela e da configuração atual do Amazon Redshift, a fase de mesclagem pode processar um número máximo de partições em uma única iteração de mesclagem. A fase de mesclagem ainda pode prosseguir se o número de partições classificadas ultrapassar o número máximo de partições de mesclagem, mas serão necessárias mais iterações de mesclagem).

Consulta de exemplo

A consulta a seguir retorna estatísticas para as operações de limpeza em três tabelas diferentes. A tabela SALES foi limpada duas vezes.

```
select table_name, xid, sort_partitions as parts, merge_increments as merges,
elapsed_time, row_delta, sortedrow_delta as sorted_delta, block_delta
from svv_vacuum_summary
order by xid;

table_  | xid  |parts|merges| elapsed_ | row_    | sorted_ | block_
name    |      |     |      | time     | delta   | delta   | delta
--------+------+-----+------+----------+---------+---------+--------
users   | 2985 |   1 |    1 | 61919653 |       0 |   49990 |      20
category| 3982 |   1 |    1 | 24136484 |       0 |      11 |       0
sales   | 3992 |   2 |    1 | 71736163 |       0 | 1207192 |      32
sales   | 4000 |   1 |    1 | 15363010 | -851648 | -851648 |    -140
(4 rows)
```

Visualizações de monitoramento de SYS

As visualizações de monitoramento são visualizações do sistema no Amazon Redshift usadas para monitorar o uso de recursos de consulta e workload de clusters provisionados e grupos de trabalho sem servidor. Essas visualizações estão localizadas no esquema `pg_catalog`. Para exibir as informações fornecidas por essas exibições, execute instruções SQL SELECT.

Salvo indicação em contrário, essas visualizações estão disponíveis para clusters do Amazon Redshift e grupos de trabalho do Amazon Redshift sem servidor.

SYS_SERVERLESS_USAGE somente coleta dados de uso do Amazon Redshift Serverless.

Tópicos

- SYS_ANALYZE_COMPRESSION_HISTORY
- SYS_ANALYZE_HISTORY
- SYS_APPLIED_MASKING_POLICY_LOG
- SYS_AUTOMATIC_OPTIMIZATION
- SYS_AUTO_TABLE_OPTIMIZATION
- SYS_CHILD_QUERY_TEXT

- SYS_CONNECTION_LOG
- SYS_COPY_JOB
- SYS_COPY_JOB_DETAIL
- SYS_COPY_JOB_INFO
- SYS_COPY_REPLACEMENTS
- SYS_DATASHARE_CHANGE_LOG
- SYS_DATASHARE_CROSS_REGION_USAGE
- SYS_DATASHARE_USAGE_CONSUMER
- SYS_DATASHARE_USAGE_PRODUCER
- SYS_EXTERNAL_QUERY_DETAIL
- SYS_EXTERNAL_QUERY_ERROR
- SYS_EXTRA_COMPUTE_FOR_AUTOMATIC_OPTIMIZATION
- SYS_INTEGRATION_ACTIVITY
- SYS_INTEGRATION_TABLE_ACTIVITY
- SYS_INTEGRATION_TABLE_STATE_CHANGE
- SYS_LOAD_DETAIL
- SYS_LUDF_DETAIL
- SYS_LOAD_ERROR_DETAIL
- SYS_LOAD_HISTORY
- SYS_MV_REFRESH_HISTORY
- SYS_MV_STATE
- SYS_PROCEDURE_CALL
- SYS_PROCEDURE_MESSAGES
- SYS_QUERY_DETAIL
- SYS_QUERY_EXPLAIN
- SYS_QUERY_HISTORY
- SYS_QUERY_TEXT
- SYS_REDSHIFT_TEMPLATE
- SYS_RESTORE_LOG
- SYS_RESTORE_STATE

- SYS_SCHEMA_QUOTA_VIOLATIONS
- SYS_SERVERLESS_USAGE
- SYS_SESSION_HISTORY
- SYS_SPATIAL_SIMPLIFY
- SYS_STREAM_SCAN_ERRORS
- SYS_STREAM_SCAN_STATES
- SYS_TRANSACTION_HISTORY
- SYS_UDF_LOG
- SYS_UNLOAD_DETAIL
- SYS_UNLOAD_HISTORY
- SYS_USERLOG
- SYS_VACUUM_HISTORY

SYS_ANALYZE_COMPRESSION_HISTORY

Registra os detalhes das operações de análise de compactação durante os comandos COPY ou ANALYZE COMPRESSION.

SYS_ANALYZE_COMPRESSION_HISTORY permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	integer	O ID do usuário que gerou a entrada.
start_time	timestamp	O horário em que a operação de análise de compactação foi iniciada.
transaction_id	bigint	O ID da transação da operação de análise de compactação.

Nome da coluna	Tipo de dados	Descrição
table_id	integer	O ID da tabela da tabela que foi analisada.
table_name	character(128)	O nome da tabela da tabela que foi analisada.
column_position	integer	O índice da coluna na tabela que foi analisada para determinar a codificação de compactação.
old_encoding	character(15)	O tipo de codificação antes da análise de compactação.
new_encoding	character(15)	O tipo de codificação após a análise de compactação.
mode	character(14)	Os valores possíveis são: PRESET Especifica que `new_encoding` é determinado pelo comando COPY do Amazon Redshift com base no tipo de dados da coluna. Não é criada nenhuma amostra de dados. ON Especifica que `new_encoding` é determinado pelo comando COPY do Amazon Redshift com base em uma análise de dados de amostra. ANALYZE ONLY Especifica que `new_encoding` é determinado pelo comando ANALYZE COMPRESSION do Amazon Redshift com base em uma análise de dados de amostra. No entanto, o tipo de codificação da coluna analisada não é alterado.

Consultas de exemplo

O exemplo a seguir inspeciona os detalhes da análise de compactação na tabela `lineitem` pelo último comando COPY executado na mesma sessão.

```
select transaction_id, table_id, btrim(table_name) as table_name, column_position,
 old_encoding, new_encoding, mode
from sys_analyze_compression_history
where transaction_id = (select transaction_id from sys_query_history where query_id =
 pg_last_copy_id()) order by column_position;

 transaction_id  |  table_id   | table_name | column_position |  old_encoding   |
 new_encoding   |      mode
-----------------+-------------+------------+-----------------+-----------------
+-----------------+-------------
      8196       |   248126    | lineitem   |        0        | mostly32        |
 mostly32        | ON
      8196       |   248126    | lineitem   |        1        | mostly32        | lzo
           | ON
      8196       |   248126    | lineitem   |        2        | lzo             |
 delta32k        | ON
      8196       |   248126    | lineitem   |        3        | delta           | delta
           | ON
      8196       |   248126    | lineitem   |        4        | bytedict        |
 bytedict        | ON
      8196       |   248126    | lineitem   |        5        | mostly32        |
 mostly32        | ON
      8196       |   248126    | lineitem   |        6        | delta           | delta
           | ON
      8196       |   248126    | lineitem   |        7        | delta           | delta
           | ON
      8196       |   248126    | lineitem   |        8        | lzo             | zstd
           | ON
      8196       |   248126    | lineitem   |        9        | runlength       | zstd
           | ON
      8196       |   248126    | lineitem   |       10        | delta           | lzo
           | ON
      8196       |   248126    | lineitem   |       11        | delta           | delta
           | ON
      8196       |   248126    | lineitem   |       12        | delta           | delta
           | ON
      8196       |   248126    | lineitem   |       13        | bytedict        | zstd
           | ON
```

```
      8196       |   248126    | lineitem   |       14        | bytedict        | zstd
           | ON
      8196       |   248126    | lineitem   |       15        | text255         | zstd
           | ON
(16 rows)
```

SYS_ANALYZE_HISTORY

Registra detalhes de operações ANALYZE

SYS_ANALYZE_HISTORY é visível somente para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	integer	O ID do usuário que gerou a entrada.
transaction_id	long	O ID da transação.
database_name	char(30)	O nome do banco de dados.
table_name	char(30)	O nome da tabela.
table_id	integer	O ID da tabela.
is_automatic	char(1)	O valor será true (t) se a operação incluir uma operação ANALYZE do Amazon Redshift por padrão. O valor será false (f) se o comando ANALYZE tiver sido executado explicitamente.
status	char(15)	O resultado do comando de análise. Os valores possíveis são Full, Skipped e PredicateColumn.
start_time	timestamp	A hora de início (em UTC) da execução da operação ANALYZE.

Nome da coluna	Tipo de dados	Descrição
end_time	timestamp	A hora de término (em UTC) da execução da operação ANALYZE.
rows	double	O número total de linhas na tabela.
modified_rows	double	O número total de linhas que foram modificadas desde a última operação ANALYZE.
analyze_threshold_percent	integer	O valor do parâmetro analyze_threshold_percent.
last_analyze_time	timestamp	A hora (em UTC) em que a tabela foi analisada anteriormente.

Consultas de exemplo

```
 user_id | transaction_id | database_name | schema_name |      table_name     |
 table_id | is_automatic | Status |         start_time         |          end_time
     | rows | modified_rows | analyze_threshold_percent |  last_analyze_time
---------+----------------+---------------+------------+--------------------
+----------+--------------+--------+---------------------------+----------
+-----------------+------+--------------+-------------------------
+---------------------
     101 |           8006 |           dev |      public | test_table_562bf8dc
 |   110427 |            f |   Full | 2023-09-21 18:33:08.504646 | 2023-09-21
 18:33:24.296498 |    5 |             5 |                          0 | 2000-01-01
 00:00:00
```

SYS_APPLIED_MASKING_POLICY_LOG

Use SYS_APPLIED_MASKING_POLICY_LOG para monitorar a aplicação de políticas de mascaramento dinâmico de dados em consultas que fazem referência a relações protegidas por DDM.

SYS_APPLIED_MASKING_POLICY_LOG permanece visível para os seguintes usuários:

- Superusuários
- Usuários com a função `sys:operator`
- Usuários com a permissão ACCESS SYSTEM TABLE

Usuários regulares verão 0 linha.

Observe que SYS_APPLIED_MASKING_POLICY_LOG não está visível para os usuários com a função `sys:secadmin`.

Para obter mais informações sobre mascaramento dinâmico de dados, consulte Mascaramento dinâmico de dados.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
policy_name	texto	O nome da política de mascaramento.
user_id	texto	O ID do usuário que executou a consulta.
record_time	timestamp	A hora em que a entrada da visualização do sistema foi registrada.
session_id	int	O ID do processo.
transaction_id	long	O ID da transação.
query_id	int	O ID da consulta.
database_name	texto	O nome do banco de dados no qual a consulta foi executada.
relation_name	texto	O nome da tabela à qual a política de mascaramento é aplicada.

Nome da coluna	Tipo de dados	Descrição
schema_name	texto	O nome do esquema no qual a tabela está.
attachment_id	long	O ID da política de mascaramento anexada.
relation_kind	texto	O nome da tabela à qual a política de mascaramento é aplicada. Os valores possíveis são `TABLE`, `VIEW`, `LATE BINDING VIEW` e `MATERIALIZED VIEW` .
query_uuid	character(36)	O identificador globalmente exclusivo (UUID) da consulta.

Consultas de exemplo

O exemplo a seguir mostra que a política de mascaramento `mask_credit_card_full` está anexada à tabela `credit_db.public.credit_cards`.

```
select policy_name, database_name, relation_name, schema_name, relation_kind
from sys_applied_masking_policy_log;

policy_name           | database_name | relation_name | schema_name | relation_kind
----------------------+---------------+---------------+-------------+---------------
mask_credit_card_full | credit_db     | credit_cards  | public      | table

(1 row)
```

SYS_AUTOMATIC_OPTIMIZATION

Use SYS_AUTOMATIC_OPTIMIZATION para visualizar detalhes sobre as tarefas que o Amazon Redshift executa para otimização automática, também conhecidas como operações autônomas. Para ter mais informações sobre otimização automática, consulte Otimização automática de banco de dados.

SYS_AUTOMATIC_OPTIMIZATION está visível apenas para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
session_id	integer	ID do processo que está executando as consultas de tarefa. Inicialmente definido como 0 quando a tarefa é criada, esse valor se torna diferente de zero somente quando `event` está definido como Started.
database_name	character(128)	O nome do banco de dados no qual a tarefa é executada.
object_type	character(30)	O tipo de objeto no qual a tarefa de operações autônomas opera. Os valores possíveis são os seguintes: • table • mv • table_column_pairs
object_ids	character(512)	O identificador dos objetos nos quais a tarefa de operações autônomas está operando. Esse campo pode conter mais de um objeto quando a tarefa é executada em vários objetos do banco de dados. Nesse caso, os identificadores são separados por vírgulas.
task_type	character(100)	O tipo de tarefa de operações autônomas executada. As possíveis tarefas são as seguintes: • AutoAlterTableTaskSortkey • AutoAlterTableTaskDistkey • VacuumSort • VacuumDelete • Analyze
event	character(50)	O tipo de evento de transição de estado para a tarefa de operações autônomas. Os valores possíveis são os seguintes: • Queued

Nome da coluna	Tipo de dados	Descrição
		• Started • Suspended • Completed • Failed
event_time	timestamp	A hora em que a transição de estado ocorreu.
status	character(512)	O status de execução da tarefa de otimização. Vazio, se não houver tentativa de execução da tarefa.
compute_type	character(100)	Tipo de recurso de computação usado pela tarefa. Os valores possíveis para o modo provisionado são os seguintes: • primary • primary-scale O valor também pode estar vazio quando provision ado se a tarefa não usar nenhum cálculo. Para ter mais informações sobre os diferentes tipos de recurso de computação que é possível usar para operações autônomas, consulte Alocação de recursos extras de computação para otimização automática de banco de dados.
task_details	character(512)	Detalhes adicionais da tarefa, se houver. Esse campo pode estar vazio.

Observações de uso

A coluna compute_type ficará vazia para clusters sem servidor porque o Amazon Redshift não diferencia os recursos de computação primários ou de escalabilidade primária. Os recursos de computação de clusters sem servidor são medidos pelo uso de unidades de processamento do Redshift (RPUs). Para ter mais informações, consulte Capacidade computacional do Amazon Redshift sem servidor.

Exemplos

A consulta a seguir exibe as otimizações automáticas realizadas na tabela 155259.

```
SELECT pid, trim(task_type) as task_type,
  trim(database) as database,
  trim(status) as status,
  trim(event) as event,
  event_time
from SYS_AUTOMATIC_OPTIMIZATION
WHERE object_ids like '%155259%'
AND status = 'Task completed successfully';

 task_type  |    database    |           status            |   event   |
 event_time
------------+----------------+-----------------------------+-----------
+----------------------------
 VacuumSort | tpcds_100g_oob | Task completed successfully | Completed | 2025-12-22
 07:27:15.943018
```

A consulta a seguir mostra todas as otimizações automáticas “VacuumSort” executadas. Para ter mais informações sobre “VacuumSort”, consulte Classificação automática de tabela.

```
SELECT trim(task_type) as task_type,
  trim(database) as database,
  trim(object_type) as object_type,
  trim(object_ids) as object_ids,
  trim(status) as status,
  trim(event) as event,
  event_time
from SYS_AUTOMATIC_OPTIMIZATION
WHERE task_type like '%VacuumSort%'
AND status = 'Task completed successfully';

task_type  |    database    | object_type | object_ids |           status            |
  event   |         event_time
------------+----------------+-------------+------------+-----------------------------
+-----------+----------------------------
 VacuumSort | tpcds_100g_oob | table       | 155301     | Task completed successfully |
 Completed | 2025-12-22 07:14:00.065391
 VacuumSort | tpcds_100g_oob | table       | 155303     | Task completed successfully |
 Completed | 2025-12-22 07:14:09.158251
```

```
 VacuumSort | tpcds_100g_oob | table       | 155291     | Task completed successfully |
 Completed | 2025-12-22 07:17:06.61164
 VacuumSort | tpcds_100g_oob | table       | 155293     | Task completed successfully |
 Completed | 2025-12-22 07:17:37.015069
 VacuumSort | tpcds_100g_oob | table       | 155281     | Task completed successfully |
 Completed | 2025-12-22 07:18:54.903935
 VacuumSort | tpcds_100g_oob | table       | 155279     | Task completed successfully |
 Completed | 2025-12-22 07:20:13.960002
 VacuumSort | tpcds_100g_oob | table       | 155271     | Task completed successfully |
 Completed | 2025-12-22 07:21:26.095549
 VacuumSort | tpcds_100g_oob | table       | 155267     | Task completed successfully |
 Completed | 2025-12-22 07:22:48.119249
 VacuumSort | tpcds_100g_oob | table       | 155269     | Task completed successfully |
 Completed | 2025-12-22 07:24:12.010424
 VacuumSort | tpcds_100g_oob | table       | 155263     | Task completed successfully |
 Completed | 2025-12-22 07:25:35.958388
 VacuumSort | tpcds_100g_oob | table       | 155265     | Task completed successfully |
 Completed | 2025-12-22 07:26:40.580395
 VacuumSort | tpcds_100g_oob | table       | 155259     | Task completed successfully |
 Completed | 2025-12-22 07:27:15.943018
(12 rows)
```

SYS_AUTO_TABLE_OPTIMIZATION

Regista ações automatizadas executadas pelo Amazon Redshift em tabelas definidas para otimização automática.

SYS_AUTO_TABLE_OPTIMIZATION está visível apenas para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
transaction_id	long	O identificador da transação.
session_id	int	O identificador da sessão do processo que executou o comando alter.
table_id	int	O identificador da tabela.

Nome da coluna	Tipo de dados	Descrição
alter_table_type	character (32)	O tipo de recomendação. Os valores possíveis são `distkey`, `sortkey` e `encode`.
status	character (128)	O status da conclusão da recomendação. Os valores possíveis são `Start`, `Complete`, `Skipped`, `Abort`, `Checkpoint` e `Failed`.
event_time	timestamp	O carimbo de data e hora da coluna de status.
alter_from	character (200)	O estilo de distribuição anterior e as chaves de classificação da tabela antes de aplicar a recomendação. O valor é truncado em incrementos de 200 caracteres.

Consultas de exemplo

No exemplo a seguir, as linhas no resultado mostram ações executadas pelo Amazon Redshift.

```
SELECT table_id, alter_table_type, status, event_time, alter_from
FROM SYS_AUTO_TABLE_OPTIMIZATION;

 table_id |  alter_table_type   |                        status
 |         event_time          |  alter_from
----------+---------------------+-----------------------------------------------------
+-----------------------------+-----------------
   118082 | sortkey             | Start
 | 2020-08-22 19:42:20.727049  |
   118078 | sortkey             | Start
 | 2020-08-22 19:43:54.728819  |
   118082 | sortkey             | Start
 | 2020-08-22 19:42:52.690264  |
   118072 | sortkey             | Start
 | 2020-08-22 19:44:14.793572  |
   118082 | sortkey             | Failed
 | 2020-08-22 19:42:20.728917  |
   118078 | sortkey             | Complete
 | 2020-08-22 19:43:54.792705  |  SORTKEY: None;
   118086 | sortkey             | Complete
 | 2020-08-22 19:42:00.72635   |  SORTKEY: None;
```

```
  118082 | sortkey             | Complete
| 2020-08-22 19:43:34.728144  |  SORTKEY: None;
  118072 | sortkey             | Skipped:Retry exceeds the maximum limit for a table.
| 2020-08-22 19:44:46.706155  |
  118086 | sortkey             | Start
| 2020-08-22 19:42:00.685255  |
  118082 | sortkey             | Start
| 2020-08-22 19:43:34.69531   |
  118072 | sortkey             | Start
| 2020-08-22 19:44:46.703331  |
  118082 | sortkey             | Checkpoint: progress 14.755079%
| 2020-08-22 19:42:52.692828  |
  118072 | sortkey             | Failed
| 2020-08-22 19:44:14.796071  |
  116723 | sortkey             | Abort:This table is not AUTO.
| 2020-10-28 05:12:58.479233  |
  110203 | distkey             | Abort:This table is not AUTO.
| 2020-10-28 05:45:54.67259   |
```

SYS_CHILD_QUERY_TEXT

Retorna o texto SQL de uma consulta secundária.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	integer	O identificador do usuário que enviou a consulta.
query_id	bigint	O ID da consulta do usuário.
child_query_sequence	integer	A sequência da consulta de usuário regravada, começando com 1.
sequence	integer	O número de sequência para essa parte da consulta.
texto	character (200)	Os primeiros 200 caracteres do texto da consulta SQL.

Nome da coluna	Tipo de dados	Descrição
query_uuid	character (36)	O identificador globalmente exclusivo (UUID) da consulta.

Consultas de exemplo

No exemplo a seguir, as linhas no resultado mostram ações executadas pelo Amazon Redshift.

```
SELECT * from sys_child_query_text where query_id = '34487366' order by
 child_query_sequence asc, sequence asc;

user_id | query_id | child_query_sequence | sequence | text
--------|----------|----------------------|----------|-----------------------------------------
100     | 34899339 |   1                  |    0     |  /* RQEV2-aY6ZZ1ZpQK */\nwith
 venue as (\n    select venueid,\n            venuename,\n            venuestate\n
 from venue\n), event as (\n    select eventid,\n            venueid,\n            date

100     | 34899339 |   1                  |    1     |  id,\n            eventname\n
  from event\n    where eventname like '3 Doors Down'\n), users as (\n    select userid
\n    from users\n), sales as (\n    select salesid,\n            pricepaid,
100     | 34899339 |   1                  |    2     |  \n            eventid,\n
       buyerid\n    from sales\n)\nselect e.eventname,\n        v.venuename,\n
 count(distinct(u.userid)) as unique_customers,\n        sum(s.pricepaid) as total_sal

100     | 34899339 |   1                  |    3     |  es\nfrom venue as v inner join
 event e on v.venueid = e.venueid\ninner join sales s on e.eventid = s.eventid inner
 join users u on s.buyerid = u.userid\ngroup by 1,2\norder by 4 desc limit 100
```

SYS_CONNECTION_LOG

Registra em log as tentativas de autenticação e as conexões e desconexões.

SYS_CONNECTION_LOG só permanece visível para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
event	character(50)	O evento de conexão ou de autenticação.
record_time	timestamp	O horário em que o evento ocorreu.
remote_host	character(45)	O nome ou endereço IP do host remoto.
remote_port	character(32)	O número da porta do host remoto.
session_id	integer	O ID do processo associado à instrução.
database_name	character(50)	Database name.
user_name	character(50)	Nome de usuário.
auth_method	character(32)	O método de autenticação.
duration	integer	A duração da conexão em microssegundos.
ssl_version	character(50)	A versão do Secure Sockets Layer (SSL). Esta coluna fica vazia quando o SSL está desabilitado.
ssl_cipher	character(128)	A codificação do SSL.
mtu	integer	A unidade de transmissão máxima (MTU).
ssl_compression	character(64)	O tipo de compactação do SSL.
ssl_expansion	character(64)	O tipo de expansão do SSL.

Nome da coluna	Tipo de dados	Descrição
iam_auth_guid	character(36)	O ID de autenticação da IAM para a solicitação de CloudTrail.
application_name	character(250)	A iniciais ou o nome atualizado da aplicação de uma sessão.
driver_version	character(64)	A versão do driver ODBC ou JDBC que se conecta ao cluster do Amazon Redshift a partir de ferramentas de cliente SQL de terceiros.
os_version	character(64)	A versão do sistema operacional que está na máquina cliente que se conecta ao cluster do Amazon Redshift.
plugin_name	character(32)	O nome do plugin usado para se conectar ao seu cluster do Amazon Redshift.
protocol_version	integer	A versão do protocolo interno que o driver do Amazon Redshift usa ao estabelecer sua conexão com o servidor. As versões do protocolo são negociadas entre o driver e o servidor. A versão descreve os recursos disponíveis. Os valores válidos são: • 0 (BASE_SERVER_PROTOCOL_VERSION) • 1 (EXTENDED_RESULT_METADATA_SERVER_PROTOCOL_VERSION): para salvar uma viagem de ida e volta por consulta, o servidor envia informações extras de metadados do conjunto de resultados. • 2 (BINARY_PROTOCOL_VERSION): dependendo do tipo de dados do conjunto de resultados, o servidor envia dados em formato binário. • 3 (EXTENDED2_RESULT_METADATA_SERVER_PROTOCOL_VERSION): o servidor envia informações que diferenciam entre maiúsculas e minúsculas (agrupamento) de uma coluna.

Nome da coluna	Tipo de dados	Descrição
global_session_id	character(36)	O identificador exclusivo global da sessão atual. O ID da sessão persiste por meio da reinicialização da falha do nó.

Consultas de exemplo

Para visualizar os detalhes das conexões abertas, execute a consulta a seguir.

```
select record_time, user_name, database_name, remote_host, remote_port
from sys_connection_log
where event = 'initiating session'
and session_id not in
(select session_id from sys_connection_log
where event = 'disconnecting session')
order by 1 desc;

record_time         | user_name   | database_name   | remote_host   | remote_port

--------------------+-------------+-----------------+---------------
+---------------------------------
2014-11-06 20:30:06 | rdsdb       | dev             | [local]       |

2014-11-06 20:29:37 | test001     | test            | 10.49.42.138  | 11111

2014-11-05 20:30:29 | rdsdb       | dev             | 10.49.42.138  | 33333

2014-11-05 20:28:35 | rdsdb       | dev             | [local]       |
(4 rows)
```

O exemplo a seguir reflete uma tentativa falha de autenticação e uma operação de conexão e desconexão bem-sucedida.

```
select event, record_time, remote_host, user_name
from sys_connection_log order by record_time;

            event      |         record_time        |  remote_host  | user_name

-----------------------+----------------------------+---------------+---------
```

```
authentication failure | 2012-10-25 14:41:56.96391  | 10.49.42.138  | john

authenticated          | 2012-10-25 14:42:10.87613  | 10.49.42.138  | john

initiating session     | 2012-10-25 14:42:10.87638  | 10.49.42.138  | john

disconnecting session  | 2012-10-25 14:42:19.95992  | 10.49.42.138  | john

(4 rows)
```

O exemplo a seguir mostra a versão do driver ODBC, o sistema operacional na máquina cliente e o plug-in usado para se conectar ao cluster do Amazon Redshift. Neste exemplo, o plugin usado é para autenticação de driver ODBC padrão usando um nome de login e senha.

```
select driver_version, os_version, plugin_name from sys_connection_log;

driver_version                          |  os_version                       |
 plugin_name
----------------------------------------+-----------------------------------
+--------------------
Amazon Redshift ODBC Driver 1.4.15.0001 | Darwin 18.7.0 x86_64              | none
Amazon Redshift ODBC Driver 1.4.15.0001 | Linux 4.15.0-101-generic x86_64   | none
```

O exemplo a seguir mostra a versão do sistema operacional na máquina cliente, a versão do driver e a versão do protocolo.

```
select os_version, driver_version, protocol_version from sys_connection_log;

os_version                     |  driver_version              | protocol_version
-------------------------------+------------------------------+-------------------
Linux 4.15.0-101-generic x86_64 | Redshift JDBC Driver 2.0.0.0 | 2
Linux 4.15.0-101-generic x86_64 | Redshift JDBC Driver 2.0.0.0 | 2
Linux 4.15.0-101-generic x86_64 | Redshift JDBC Driver 2.0.0.0 | 2
```

SYS_COPY_JOB

Use SYS_COPY_JOB para visualizar detalhes dos comandos COPY JOB.

Essa visão contém os comandos COPY JOB que foram criados.

SYS_COPY_JOB é visível a todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
job_id	bigint	O identificador do trabalho de cópia.
job_name	character(128)	O nome do trabalho de cópia.
iam_role	character(128)	O perfil do IAM especificado na instrução COPY.
job_text	character(256)	Os parâmetros da instrução COPY.
is_auto	integer	Indica se COPY JOB é executada automaticamente pelo Amazon Redshift. 1 indica verdadeiro, 0 indica falso.
on_error_suspend	integer	Essas informações são somente para uso interno.

SYS_COPY_JOB_DETAIL

Use SYS_COPY_JOB_DETAIL para visualizar detalhes dos comandos COPY JOB.

Essa visão contém os comandos COPY JOB que foram criados. Se o COPY JOB tentar carregar um arquivo e houver falha, o arquivo será ignorado em futuras tentativas automáticas do COPY JOB.

SYS_COPY_JOB_DETAIL é visível a todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	integer	O identificador do usuário que tem o trabalho.
database_name	texto	O nome do banco de dados que contém o trabalho.
job_id	integer	O identificador do trabalho de cópia.
job_name	texto	O nome do trabalho de cópia.
file_location	texto	O nome do bucket do Amazon S3 para a entrada.
file_name	texto	O nome da chave do Amazon S3 para a entrada.
file_size	bigint	O tamanho da entrada em bytes.
file_etag	texto	A tag de entidade do Amazon S3 (ETag) para a entrada.
job_text	character(256)	Os parâmetros da instrução COPY.
modification_time	timestamp	A hora (UTC) em que o objeto S3 foi criado ou modificado ao iniciar a ingestão. No modo de listagem do S3, trata-se da hora de modificação do objeto. Para o modo de notificação, trata-se da hora em que o evento de notificação ocorreu.

Nome da coluna	Tipo de dados	Descrição
enqueue_time	timestamp	A hora (UTC) em que a entrada foi adicionada à fila de ingestão.
status	char(1)	O status da entrada. Valores válidos: P para ingestão pendente, I para ingerido, E para erro na ingestão e U para desconhecido (indicando algum tipo de estado inesperado).

O exemplo a seguir gera uma linha para entradas ingeridas.

```
SELECT * FROM SYS_COPY_JOB_DETAIL WHERE status ilike '%ingested%' limit 1;


user_id | 100
database_name | dev
job_name | many_job_4_3
job_id | 110702
file_location | saral-sqs-system4623202051-0
file_name | frenzy-9/4623202051/file_0_107
file_size | 11302
file_etag | 51b2d78ac5b5aecf4ee6f8374815ad19
modification_time | 2024-07-15 20:43:14
enqueue_time | 2024-07-15 20:44:24
status | Ingested
```

SYS_COPY_JOB_INFO

Use SYS_COPY_JOB_INFO para visualizar mensagens registradas sobre um COPY JOB.

Esta exibição contém informações sobre erros em um COPY JOB que foi executado.

SYS_COPY_JOB_INFO é visível a todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
job_id	bigint	O identificador do trabalho de cópia.
job_name	character(128)	O nome do trabalho de cópia.
database_name	character(128)	O nome do banco de dados.
record_time	timestamp	A hora (UTC) em que a mensagem foi registrada.
message	character(512)	Esta é a mensagem do evento registrado em log para um COPY JOB.

SYS_COPY_REPLACEMENTS

Exibe um log que registra quando caracteres UTF-8 inválidos são substituídos pelo comando COPY com a opção ACCEPTINVCHARS. Uma entrada de log é adicionada a SYS_COPY_REPLACEMENTS para cada uma das 100 primeiras linhas em cada fatia de nó que exigiu pelo menos uma substituição.

É possível usar essa visualização para ver informações sobre grupos de trabalho sem servidor e clusters provisionados.

SYS_COPY_REPLACEMENTS permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	integer	O ID do usuário que gerou a consulta.

Nome da coluna	Tipo de dados	Descrição
query_id	bigint	O ID da consulta. A coluna pode ser usada para unir outras tabelas e visualizações do sistema.
table_id	integer	O ID da tabela.
file_name	character (256)	O caminho completo do arquivo de entrada para o comando COPY.
column_name	character (127)	O primeiro campo que contém um caractere UTF-8 inválido.
line_number	bigint	O número da linha no arquivo de dados de entrada que contém um caractere UTF-8 inválido; -1 indica que o número da linha não está disponível, como ao copiar de um arquivo de dados colunares.
raw_line	character (1024)	Os dados brutos de carga que contêm um caractere UTF-8 inválido.

Consultas de exemplo

O exemplo a seguir retorna as substituições da operação COPY mais recente.

```
select query_idp, table_id, file_name, line_number, colname
from sys_copy_replacements
where query = pg_last_copy_id();


 query_id | table_id |   file_name                                                |
 line_number | column_name
 ---------+----------+------------------------------------------------------------
+-------------+--------
    96    |    26    | s3://DOC-EXAMPLE-BUCKET/allusers_pipe.txt             |
 123 | city
    96    |    26    | s3://DOC-EXAMPLE-BUCKET/allusers_pipe.txt             |
 456 | city
```

```
    96    |    26    | s3://DOC-EXAMPLE-BUCKET/allusers_pipe.txt            |
 789 | city
    96    |    26    | s3://DOC-EXAMPLE-BUCKET/allusers_pipe.txt            |
 012 | city
    96    |    26    | s3://DOC-EXAMPLE-BUCKET/allusers_pipe.txt            |
 119 | city
...
```

SYS_DATASHARE_CHANGE_LOG

Regista a exibição consolidada para rastrear alterações nos conjuntos de dados em clusters de produtores e consumidores.

SYS_DATASHARE_CHANGE_LOG permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	integer	O ID do usuário que executa a ação.
user_name	varchar(128)	O nome do usuário que está realizando a ação.
session_id	integer	O ID da sessão.
transaction_id	bigint	O ID da transação.
share_id	integer	O ID do datashare afetado.
share_name	varchar(128)	O nome do datashare.
source_database_id	integer	O ID do banco de dados ao qual o datashare pertence.

Nome da coluna	Tipo de dados	Descrição
source_database_name	varchar(128)	O nome do banco de dados ao qual o datashare pertence.
consumer_database_id	integer	O ID do banco de dados importado do datashare.
consumer_database_name	varchar(128)	O nome do banco de dados importado do datashare.
arn	varchar(192)	O ARN do recurso por trás do banco de dados importado.
record_time	timestamp	O timestamp da ação.
ação	varchar(128)	A ação que está sendo executada. Os valores possíveis são CREATE DATASHARE, DROP DATASHARE, GRANT ALTER, REVOKE ALTER, GRANT SHARE, REVOKE SHARE, ALTER ADD, ALTER REMOVE, ALTER SET, GRANT USAGE, REVOKE USAGE, CREATE DATABASE, GRANT ou REVOKE USAGE em um banco de dados compartilhado, DROP SHARED DATABASE e ALTER SHARED DATABASE.
status	integer	O status da ação. Os valores possíveis são SUCESS e ERROR-ERROR CODE.
share_object_type	varchar(64)	O tipo de objeto de banco de dados que foi adicionado ou removido do datashare. Os valores possíveis são esquemas, tabelas, colunas, funções e exibições. Este é um campo para o cluster produtor.
share_object_id	integer	O ID do objeto de banco de dados que foi adicionado ou removido do datashare. Este é um campo para o cluster produtor.
share_object_name	varchar(128)	O nome do objeto de banco de dados que foi adicionado ou removido do datashare. Este é um campo para o cluster produtor.

Nome da coluna	Tipo de dados	Descrição
target_user_type	varchar(16)	O tipo de usuários ou grupos aos quais um privilégio foi concedido. Este é um campo para o cluster do produtor e do consumidor.
target_user_id	integer	O ID de usuários ou grupos aos quais um privilégio foi concedido. Este é um campo para o cluster do produtor e do consumidor.
target_user_name	varchar(128)	O nome dos usuários ou grupos aos quais um privilégio foi concedido. Este é um campo para o cluster do produtor e do consumidor.
consumer_account	varchar(16)	O ID da conta do consumidor de dados. Este é um campo para o cluster produtor.
consumer_namespace	varchar(64)	O namespace da conta de consumidor de dados. Este é um campo para o cluster produtor.
producer_account	varchar(16)	O ID da conta do produtor à qual o datashare pertence. Este é um campo para o cluster do consumidor.
producer_namespace	varchar(64)	O namespace da conta do produto ao qual o datashare pertence. Este é um campo para o cluster do consumidor.
attribute_name	varchar(64)	O nome de um atributo do datashare ou do banco de dados compartilhado.
attribute_value	varchar(128)	O valor de um atributo do datashare ou do banco de dados compartilhado.
message	varchar(512)	A mensagem de erro quando uma ação falha.

Consultas de exemplo

O exemplo a seguir mostra uma visão de SYS_DATASHARE_CHANGE_LOG.

```
SELECT DISTINCT action
FROM sys_datashare_change_log
```

```
WHERE share_object_name LIKE 'tickit%';

         action
 ----------------------
  "ALTER DATASHARE ADD"
```

SYS_DATASHARE_CROSS_REGION_USAGE

Use a visão SYS_DATASHARE_CROSS_REGION_USAGE para obter um resumo do uso transferido de dados entre regiões causado por uma consulta de compartilhamento de dados entre regiões. SYS_DATASHARE_CROSS_REGION_USAGE agrega detalhes no nível do segmento.

SYS_DATASHARE_CROSS_REGION_USAGE só está visível para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
query_id	integer	A ID da consulta. Use esse valor para unir várias outras tabelas e visões do sistema.
child_query_sequence	integer	A sequência da consulta de usuário regravada, começando com 1.
segment_id	bigint	O número do segmento. Uma consulta consiste em vários segmentos e cada segmento consiste em uma ou mais etapas.
start_time	horário	O horário em UTC do início da transferência dos dados.
end_time	horário	O horário em UTC do término da transferência dos dados.
transferred_data	bigint	O número de bytes de dados transferidos de uma região produtora para uma região consumidora.
source_region	char(25)	A região produtora da qual a consulta transferiu os dados.

Nome da coluna	Tipo de dados	Descrição
query_uuid	character (36)	O identificador globalmente exclusivo (UUID) da consulta.

Consultas de exemplo

O exemplo a seguir mostra uma visão SYS_DATASHARE_CROSS_REGION_USAGE.

```
SELECT query_id, segment_id, transferred_data, source_region
from sys_datashare_cross_region_usage
where query_id = pg_last_query_id()
order by query_id, segment_id;

  query_id | segment_id | transferred_data | source_region
-----------+------------+------------------+---------------
    200048 |          2 |          4194304 |    us-west-1
    200048 |          2 |          4194304 |    us-east-2
```

SYS_DATASHARE_USAGE_CONSUMER

Registra a atividade e o uso de datashares. Esta visualização só é relevante no cluster de consumidores.

SYS_DATASHARE_USAGE_CONSUMER é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	integer	O ID do usuário que emite a solicitação.
session_id	integer	O ID do processo líder que executa a consulta.

Nome da coluna	Tipo de dados	Descrição
transaction_id	bigint	O contexto da transação atual.
request_id	varchar(50)	O ID exclusivo da chamada à API solicitada.
request_type	varchar(25)	O tipo de solicitação feita ao cluster produtor.
transaction_uid	varchar(50)	O ID exclusivo da transação.
record_time	timestamp	A hora em que a ação é registrada.
status	integer	O status da chamada de API solicitada.
erro	varchar(512)	A mensagem para um erro.

Consultas de exemplo

O exemplo a seguir mostra a visualização SYS_DATASHARE_USAGE_CONSUMER.

```
SELECT request_type, status, trim(error) AS error
FROM sys_datashare_usage_consumer

  request_type  | status | error_message
----------------+--------+---------------
 "GET RELATION" |   0    |
```

SYS_DATASHARE_USAGE_PRODUCER

Registra a atividade e o uso de datashares. Essa visualização é relevante apenas no cluster do produtor.

SYS_DATASHARE_USAGE_PRODUCER é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
share_id	integer	O ID do objeto (OID) do datashare.
share_name	varchar(128)	O nome do datashare.
request_id	varchar(50)	O ID exclusivo da chamada à API solicitada.
request_type	varchar(25)	O tipo de solicitação feita ao cluster produtor.
object_type	varchar(64)	O tipo do objeto que está sendo compartilhado do datashare. Os valores possíveis são esquemas, tabelas, colunas, funções e exibições.
object_oid	integer	O ID do objeto que está sendo compartilhado do datashare.
nome_objeto	varchar(128)	O nome do objeto que está sendo compartilhado do datashare.
consumer_account	varchar(16)	A conta da conta de consumidor com a qual o datashare é compartilhado.
consumer_namespace	varchar(64)	O namespace da conta de consumidor com a qual o datashare é compartilhado.
consumer_transaction_uid	varchar(50)	O ID de transação exclusivo da instrução no cluster do consumidor.
record_time	timestamp	A hora em que a ação é registrada.

Nome da coluna	Tipo de dados	Descrição
status	integer	O status do datashare.
erro	varchar(512)	A mensagem para um erro.
consumer_region	char(64)	A região na qual o cluster do consumidor está.

Consultas de exemplo

O exemplo a seguir mostra a visualização SYS_DATASHARE_USAGE_PRODUCER.

```
SELECT DISTINCT
FROM sys_datashare_usage_producer
WHERE object_name LIKE 'tickit%';

   request_type
 ------------------
   "GET RELATION"
```

SYS_EXTERNAL_QUERY_DETAIL

Use SYS_EXTERNAL_QUERY_DETAIL para visualizar detalhes de consultas no nível do segmento. Cada linha representa um segmento de determinada consulta WLM com detalhes como o número de linhas processadas, número de bytes processados e as informações de partição de tabelas externas no Amazon S3. Cada linha nessa visualização também terá uma entrada correspondente na exibição SYS_QUERY_DETAIL, porém essa exibição contém mais informações detalhadas relacionadas ao processamento de consultas externas.

SYS_EXTERNAL_QUERY_DETAIL é visível a todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	integer	O identificador do usuário que enviou a consulta.
query_id	bigint	O identificador da consulta externa.
transaction_id	bigint	O identificador da transação.
child_query_sequence	integer	A sequência da consulta de usuário regravada. Começa com 0, semelhante a segment_id.
segment_id	integer	O identificador do segmento da consulta.
source_type	character(32)	O tipo de origem dos dados da consulta, pode ser S3 para Redshift Spectrum e PG para consulta federada.
start_time	timestamp	O horário em que a consulta começou.
end_time	timestamp	O horário em que a consulta foi concluída.
duration	bigint	O tempo (em microsseg undos) gasto na consulta.
total_partitions	integer	O número de partições necessárias para uma consulta do Amazon S3.

Nome da coluna	Tipo de dados	Descrição
qualified_partitions	integer	O número de partições que uma consulta do Amazon S3 verificou.
scanned_files	bigint	O número de bytes processados na verificação do Amazon S3.
returned_rows	bigint	O número de linhas verificadas para uma consulta do Amazon S3 ou o número de linhas retornadas para uma consulta federada.
returned_bytes	bigint	O número de bytes verificados para uma consulta do Amazon S3 ou o número de bytes retornados para uma consulta federada.
file_format	texto	O formato dos arquivos do Amazon S3.
file_location	texto	O local do Amazon S3 da tabela externa.
external_query_text	texto	O texto da consulta no nível do segmento para uma consulta federada.
warning_message	character(4000)	A mensagem de aviso exibida quando a consulta é executada.
table_name	character(136)	O nome da tabela da etapa que está sendo operada.

Nome da coluna	Tipo de dados	Descrição
is_recursive	character(1)	Indica se há varredura recursiva para subpastas.
is_nested	character(1)	Indica se o tipo de dados da coluna aninhada é acessado.
s3list_time	bigint	A duração da listagem de arquivos em milissegundos.
get_partition_time	long	Tempo gasto para listar e qualificar partições para determinado objeto externo do AWS Glue Data Catalog e do Apache Hive.
query_uuid	character(36)	O identificador globalmente exclusivo (UUID) da consulta.

Consultas de exemplo

A consulta a seguir mostra os detalhes da consulta externa.

```
SELECT query_id,
       segment_id,
       start_time,
       end_time,
       total_partitions,
       qualified_partitions,
       scanned_files,
       returned_rows,
       returned_bytes,
       trim(external_query_text) query_text,
       trim(file_location) file_location
FROM sys_external_query_detail
ORDER BY query_id, start_time DESC
LIMIT 2;
```

Exemplo de resultado.

```
 query_id | segment_id |         start_time         |          end_time
   | total_partitions | qualified_partitions | scanned_files | returned_rows |
 returned_bytes | query_text | file_location
----------+------------+----------------------------+----------------------------
+------------------+----------------------+--------------+--------------
+----------------+------------+--------------
   763251 |          0 | 2022-02-15 22:32:23.312448 | 2022-02-15 22:32:24.036023 |
          3 |                    3 |             3 |         38203 |        2683414 |
          |
   763254 |          0 | 2022-02-15 22:32:40.17103  | 2022-02-15 22:32:40.839313 |
          3 |                    3 |             3 |         38203 |        2683414 |
          |
```

SYS_EXTERNAL_QUERY_ERROR

É possível consultar a visualização do sistema SYS_EXTERNAL_QUERY_ERROR para obter informações sobre erros de verificação do Redshift Spectrum. SYS_EXTERNAL_QUERY_ERROR exibe uma amostra dos erros registrados. O padrão são 10 entradas por consulta.

SYS_EXTERNAL_QUERY_ERROR é visível a todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	integer	O identificador do usuário que gerou essa linha.
query_id	bigint	O identificador da consulta que gerou essa linha.
file_location	char(256)	A localização dos dados que estão sendo consultados.
rowid	char(2100)	A localização do erro no arquivo. As partes de `rowid` são separadas com `:` (dois-pontos) e as outras partes podem ser adicionadas no futuro. `row_offset :row_group :row_id`

Nome da coluna	Tipo de dados	Descrição
		Um row_offset é o deslocamento (em bytes) da linha dentro do arquivo e é definido como -1 para formatos de arquivo não compatíveis. Uma tabela é dividida em row_groups, e cada grupo tem linhas com row_ids distintos .
column_name	char(127)	O nome da coluna retornada pela consulta.
original_value	char(1024)	Valor original consultado.
modified_value	char(1024)	Valor modificado retornado com base na opção de configuração de tratamento de dados especificada na consulta.
Acionador	char(128)	Opção de tratamento de dados especificada na consulta.
ação	char(128)	Ação associada à opção de tratamento de dados especificada na consulta.
action_value	char(128)	Valor do parâmetro de ação associado à opção de tratamento de dados especificada na consulta.
error_code	integer	Código do resultado da opção de tratamento de dados especificada na consulta.
query_uuid	character(36)	O identificador globalmente exclusivo (UUID) da consulta.

Consulta de exemplo

A consulta a seguir retorna a lista de linhas para as quais as operações de manipulação de dados foram executadas.

```
SELECT * FROM sys_external_query_error;
```

A consulta retorna resultados semelhantes aos resultados a seguir.

```
  user_id   query_id  file_location                                rowid
column_name           original_value             modified_value       trigger
action               action_value                 error_code
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:0
league_name           Barclays Premier League    Barclays Premier Lea UNSPECIFIED
TRUNCATE                                          156
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:0
league_nspi           34595                      32767                UNSPECIFIED
OVERFLOW_VALUE                                    199
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:1
league_nspi           34151                      32767                UNSPECIFIED
OVERFLOW_VALUE                                    199
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:2
league_name           Barclays Premier League    Barclays Premier Lea UNSPECIFIED
TRUNCATE                                          156
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:2
league_nspi           33223                      32767                UNSPECIFIED
OVERFLOW_VALUE                                    199
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:3
league_name           Barclays Premier League    Barclays Premier Lea UNSPECIFIED
TRUNCATE                                          156
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:3
league_nspi           32808                      32767                UNSPECIFIED
OVERFLOW_VALUE                                    199
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:4
league_nspi           32790                      32767                UNSPECIFIED
OVERFLOW_VALUE                                    199
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:5
league_name           Spanish Primera Division   Spanish Primera Divi UNSPECIFIED
TRUNCATE                                          156
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:6
league_name           Spanish Primera Division   Spanish Primera Divi UNSPECIFIED
TRUNCATE                                          156
```

SYS_EXTRA_COMPUTE_FOR_AUTOMATIC_OPTIMIZATION

Use SYS_EXTRA_COMPUTE_FOR_AUTOMATIC_OPTIMIZATION para visualizar os períodos de uso nos quais o Amazon Redshift executou tarefas de otimização automática usando recursos extras de computação. Para ter mais informações sobre otimização automática, consulte Otimização

automática de banco de dados. Para ter mais informações sobre otimizações automáticas executadas com recursos extras de computação, consulte Alocação de recursos extras de computação para otimização automática de banco de dados.

SYS_EXTRA_COMPUTE_FOR_AUTOMATIC_OPTIMIZATION está disponível somente para clusters provisionados.

SYS_EXTRA_COMPUTE_FOR_AUTOMATIC_OPTIMIZATION está visível apenas para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
start_time	time stamp sem fuso horário	A hora em que o período de otimização de computação extra começa.
end_time	time stamp sem fuso horário	A hora em que o período de otimização de computação extra termina.
query_count	bigint	O número de consultas de otimização automática executadas durante o período de uso.
compute_seconds	numeric(27,0)	O número de segundos usados pelo Amazon Redshift no período de uso para executar tarefas de otimização automática com recursos extras de computação.

Exemplos

Veja a seguir um exemplo de consulta que procura otimizações automáticas realizadas em 16 de setembro de 2025.

```
SELECT *
FROM sys_extra_compute_for_automatic_optimization
WHERE start_time BETWEEN '2025-09-16 00:00:00' AND '2025-09-16 23:59:59';

start_time           | end_time            | query_count | compute_seconds
---------------------+---------------------+-------------+-----------------
```

```
2025-09-16 00:00:00  | 2025-09-16 00:00:59 | 1           | 59
2025-09-16 00:01:05  | 2025-09-16 00:01:58 | 2           | 53
```

SYS_INTEGRATION_ACTIVITY

SYS_INTEGRATION_ACTIVITY exibe detalhes sobre as execuções de integração concluídas.

SYS_INTEGRATION_ACTIVITY só permanece visível para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Para obter informações sobre integrações ETL zero, consulte Working with zero-ETL integrations no Guia de gerenciamento do Amazon Redshift.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
integration_id	character (128)	O identificador associado à integração.
target_database	character (128)	O banco de dados no Amazon Redshift que recebe os dados da integração.
origem	character (128)	Os dados de origem para a integração. Entre os tipos possíveis estão `MySQL` e `PostgreSQL` .
checkpoint_name	character (128)	O nome do ponto de verificação que replica as coordenadas do log binário.
checkpoint_type	character (16)	O tipo de posto de verificação. Os valores possíveis incluem: `snapshot` e `cdc`.
checkpoint_bytes	bigint	O número de bytes nesse ponto de verificação.
last_commit_timestamp	timestamp	O carimbo de data/hora da última confirmação nesse ponto de verificação.
modified_tables	integer	O número de tabelas modificadas no ponto de verificação.

Nome da coluna	Tipo de dados	Descrição
integration_start_time	horário	A hora (UTC) em que a integração foi iniciada para esse ponto de verificação.
integration_end_time	horário	A hora (UTC) em que a integração terminou para esse ponto de verificação.

Consultas de exemplo

O comando SQL a seguir exibe o log das integrações.

```
select * from sys_integration_activity;

          integration_id              | target_database | source |
   checkpoint_name                  | checkpoint_type  | checkpoint_bytes |
 last_commit_timestamp   | modified_tables |   integration_start_time   |
 integration_end_time
--------------------------------------+-----------------+--------
+----------------------------------------------+------------------+------------------
+-------------------------+-----------------+---------------------------
+----------------------------
 76b15917-afae-4447-b7fd-08e2a5acce7b |   demo1         | MySQL  | checkpoints/
checkpoint_3_241_3_510.json     |        cdc       |        762       | 2023-05-10
 23:00:14.201 |         1       | 2023-05-10 23:00:45.054265 | 2023-05-10
 23:00:46.339826
 76b15917-afae-4447-b7fd-08e2a5acce7b |   demo1         | MySQL  | checkpoints/
checkpoint_3_16329_3_17839.json |        cdc       |       13488      | 2023-05-11
 01:33:57.411 |         2       | 2023-05-11 02:19:09.440121 | 2023-05-11
 02:19:16.090492
 76b15917-afae-4447-b7fd-08e2a5acce7b |   demo1         | MySQL  | checkpoints/
checkpoint_3_5103_3_5532.json   |        cdc       |        1657      | 2023-05-10
 23:13:14.205 |         2       | 2023-05-10 23:13:23.545487 | 2023-05-10
 23:13:25.652144
```

SYS_INTEGRATION_TABLE_ACTIVITY

SYS_INTEGRATION_TABLE_ACTIVITY exibe detalhes da atividade de inserção, exclusão e atualização de Integrações ETL zero. Há uma linha adicionada para cada ingestão concluída.

Um superusuário pode ver todas as linhas dessa tabela.

Para obter mais informações, consulte Integrações ETL zero.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
integration_id	character (128)	O identificador associado à integração.
checkpoint_name	character (128)	O nome do ponto de verificação.
target_database	character (128)	O nome do banco de dados do Amazon Redshift.
schema_name	character (128)	O nome do esquema do Amazon Redshift.
table_name	character (128)	O nome da tabela.
table_id	integer	O identificador da tabela.
record_time	timestamp	A hora (UTC) em que essa alteração foi concluída.
transaction_id	bigint	O identificador da transação.
inserted_rows	bigint	O número de linhas inseridas pela ingestão.
deleted_rows	bigint	O número de linhas excluídas pela ingestão.
updated_rows	bigint	O número de linhas atualizadas pela ingestão.
bytes_ingested	bigint	O número de bytes ingeridos.

Consultas de exemplo

O comando SQL a seguir exibe a atividade da integração.

```
select * from sys_integration_table_activity;

          integration_id              | checkpoint_name | target_database | schema_name
 |     table_name    | table_id     | record_time                | transaction_id  |
 inserted_rows  | deleted_rows | updated_rows | bytes_ingested
--------------------------------------+-----------------+-----------------
+-------------+------------------+-------------+---------------------------
+-----------------+----------------+-------------+-------------+--------------
 4798e675-8f9f-4686-b05f-92c538e19629 |                 | sample_test2    |    sample
 | SampleTestChannel |  111276      | 2023-05-12 12:40:30.656625 | 7736            |  2
             | 0            | 0            | 125
```

SYS_INTEGRATION_TABLE_STATE_CHANGE

SYS_INTEGRATION_TABLE_STATE_CHANGE exibe detalhes sobre os logs de alteração do estado da tabela das integrações.

Um superusuário pode ver todas as linhas dessa tabela.

Para ter mais informações, consulte Trabalhar com Integrações ETL zero.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
integration_id	character (128)	O identificador associado à integração.
database_name	character (128)	O nome do banco de dados do Amazon Redshift.
schema_name	character (128)	O nome do esquema do Amazon Redshift.
table_name	character (128)	O nome da tabela.

Nome da coluna	Tipo de dados	Descrição
new_state	character (128)	O estado da tabela. Os valores possíveis são `Synced`, `ResyncRequired` , `ResyncInitiated` , `Deleted`, `Failed` e `ResyncDeleted` .
table_last_replicated_checkpoint	character (128)	As coordenadas de log sincronizadas atuais.
state_change_reason	character (256)	O motivo da última transição de estado.
record_time	timestamp	O horário (UTC) em que o registro foi atualizado.

Consultas de exemplo

O comando SQL a seguir exibe o log das integrações.

```
select * from sys_integration_table_state_change;

            integration_id            | database_name | schema_name | table_name
 | new_state |  table_last_replicated_checkpoint   | state_change_reason |
 record_time
--------------------------------------+---------------+-------------+------------
+-----------+--------------------------------------+---------------------
+----------------------------
 99108e72-1cfd-414f-8cc0-0216acefac77 | perfdb        | sbtest80t3s | sbtest79   |
 Synced    | {"txn_seq":9834,"txn_id":126597515} |                     | 2023-09-20
 19:39:50.087868
 99108e72-1cfd-414f-8cc0-0216acefac77 | perfdb        | sbtest80t3s | sbtest56   |
 Synced    | {"txn_seq":9834,"txn_id":126597515} |                     | 2023-09-20
 19:39:45.54005
 99108e72-1cfd-414f-8cc0-0216acefac77 | perfdb        | sbtest80t3s | sbtest50   |
 Synced    | {"txn_seq":9834,"txn_id":126597515} |                     | 2023-09-20
 19:40:20.362504
 99108e72-1cfd-414f-8cc0-0216acefac77 | perfdb        | sbtest80t3s | sbtest18   |
 Synced    | {"txn_seq":9834,"txn_id":126597515} |                     | 2023-09-20
 19:40:32.544084
```

```
99108e72-1cfd-414f-8cc0-0216acefac77 | perfdb        | sbtest40t3s | sbtest23   |
Synced    | {"txn_seq":9834,"txn_id":126597515} |                    | 2023-09-20
15:49:05.186209
```

SYS_LOAD_DETAIL

Retorna informações para rastrear ou solucionar problemas com uma carga de dados.

Essa visualização registra o progresso de cada arquivo de dados à medida que é carregado em uma tabela de banco de dados.

Esta visualização é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	integer	O ID do usuário que gerou a entrada.
query_id	integer	ID da consulta.
file_name	character(256)	Nome do arquivo a ser carregado.
bytes_scanned	integer	O número de bytes processados na varredura do arquivo no Amazon S3.
lines_scanned	integer	O número de linhas pesquisadas na varredura do arquivo carregado. Esse número pode não corresponder ao número de linhas que são realmente carregadas. Por exemplo, a carga pode fazer a varredura mas tolerar um certo número de registros ruins com base na opção MAXERROR no comando COPY.
record_time	timestamp	O horário em que essa entrada foi atualizada pela última vez.

Nome da coluna	Tipo de dados	Descrição
splits_scanned	Número de divisões desse arquivo.	Número de divisões desse arquivo.
start_time	timestamp	Hora em que o processamento desse arquivo foi iniciado.
end_time	timestamp	Hora em que o processamento desse arquivo foi finalizado.
file_etag	character(256)	A ETag do arquivo no Amazon S3.
file_last_modified	timestamp	O carimbo de data/hora da última modificação do arquivo no Amazon S3.

Consultas de exemplo

O exemplo a seguir retorna os detalhes da última operação COPY.

```
select query_id, trim(file_name) as file, record_time
from sys_load_detail
where query_id = pg_last_copy_id();

 query_id |               file                |          record_time
----------+-----------------------------------+----------------------------
 28554    | s3://dw-tickit/category_pipe.txt | 2013-11-01 17:14:52.648486
(1 row)
```

A consulta a seguir contém as entradas de uma carga recente de tabelas para o banco de dados TICKIT:

```
select query_id, trim(file_name), record_time
from sys_load_detail
where file_name like '%tickit%' order by query_id;

 query_id |           btrim           |          record_time
----------+---------------------------+----------------------------
 22475    | tickit/allusers_pipe.txt | 2013-02-08 20:58:23.274186
 22478    | tickit/venue_pipe.txt    | 2013-02-08 20:58:25.070604
```

```
 22480    | tickit/category_pipe.txt | 2013-02-08 20:58:27.333472
 22482    | tickit/date2008_pipe.txt | 2013-02-08 20:58:28.608305
 22485    | tickit/allevents_pipe.txt| 2013-02-08 20:58:29.99489
 22487    | tickit/listings_pipe.txt | 2013-02-08 20:58:37.632939
 22593    | tickit/allusers_pipe.txt | 2013-02-08 21:04:08.400491
 22596    | tickit/venue_pipe.txt    | 2013-02-08 21:04:10.056055
 22598    | tickit/category_pipe.txt | 2013-02-08 21:04:11.465049
 22600    | tickit/date2008_pipe.txt | 2013-02-08 21:04:12.461502
 22603    | tickit/allevents_pipe.txt| 2013-02-08 21:04:14.785124
 22605    | tickit/listings_pipe.txt | 2013-02-08 21:04:20.170594

(12 rows)
```

O fato de um registro ser gravado no arquivo de log para esta visualização do sistema não significa que o carregamento foi confirmado com êxito como parte de sua transação contida. Para verificar as confirmações de carregamento, consulte a visualização STL_UTILITYTEXT e procure o registro COMMIT que corresponde a uma transação COPY. Por exemplo, essa consulta une as tabelas SYS_LOAD_DETAIL e STL_QUERY com base em uma subconsulta com a tabela STL_UTILITYTEXT:

```
select l.query_id,rtrim(l.file_name),q.transaction_id
from sys_load_detail l, sys_query_text q
where l.query_id=q.query_id
and exists
(select xid from stl_utilitytext where xid=q.transaction_id and
 rtrim("text")='COMMIT');

 query_id |           rtrim           |  transaction_id
----------+---------------------------+-----------------
 22600    | tickit/date2008_pipe.txt  | 68311
 22480    | tickit/category_pipe.txt  | 68066
  7508    | allusers_pipe.txt         | 23365
  7552    | category_pipe.txt         | 23415
  7576    | allevents_pipe.txt        | 23429
  7516    | venue_pipe.txt            | 23390
  7604    | listings_pipe.txt         | 23445
 22596    | tickit/venue_pipe.txt     | 68309
 22605    | tickit/listings_pipe.txt  | 68316
 22593    | tickit/allusers_pipe.txt  | 68305
 22485    | tickit/allevents_pipe.txt | 68071
  7561    | allevents_pipe.txt        | 23429
  7541    | category_pipe.txt         | 23415
```

```
  7558    | date2008_pipe.txt         | 23428
 22478    | tickit/venue_pipe.txt     | 68065
   526    | date2008_pipe.txt         |  2572
  7466    | allusers_pipe.txt         | 23365
 22482    | tickit/date2008_pipe.txt  | 68067
 22598    | tickit/category_pipe.txt  | 68310
 22603    | tickit/allevents_pipe.txt | 68315
 22475    | tickit/allusers_pipe.txt  | 68061
   547    | date2008_pipe.txt         |  2572
 22487    | tickit/listings_pipe.txt  | 68072
  7531    | venue_pipe.txt            | 23390
  7583    | listings_pipe.txt         | 23445
(25 rows)
```

SYS_LUDF_DETAIL

SYS_LUDF_DETAIL registra informações e métricas para funções definidas pelo usuário do Lambda (LUDFs) que foram usadas em uma consulta específica.

SYS_LUDF_DETAIL está visível somente para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	integer	O identificador do usuário que enviou a consulta que está chamando a função.
transaction_id	bigint	O identificador da transação.
query_id	bigint	Identificador de consulta de usuário.
function_oid	bigint	ID do objeto da função no catálogo.
function_position	integer	Posição numérica dessa função na etapa. Por exemplo, se houver várias chamadas da mesma função na lista SELECT de uma consulta, elas poderão ser identificadas usando esses números posicionais.

Nome da coluna	Tipo de dados	Descrição
stream_id	integer	O identificador do fluxo em que a chamada de função foi executada.
segment_id	integer	O identificador do segmento em que a chamada de função foi executada.
step_id	integer	O identificador da etapa em que a chamada de função foi executada.
lambda_function_name	char(256)	Nome da função do Lambda.
start_time	timestamp	A hora em que as chamadas começaram.
end_time	timestamp	A hora em que as chamadas terminaram.
total_duration	bigint	A duração total das chamadas em milissegundos.
invocações	integer	O número de invocações simultâneas ou externas.
total_rows	bigint	O número de linhas transmitidas/exibidas para/da chamada.
input_bytes	bigint	O número de bytes transmitidos para a chamada.
output_bytes	bigint	O número de bytes que a chamada produziu.
query_uuid	character(36)	O identificador globalmente exclusivo (UUID) da consulta.

Consultas de exemplo

O exemplo a seguir usa uma UDF do Lambda em uma consulta e, em seguida, mostra como consultar a visualização SYS_LUDF_DETAIL para ver os detalhes de execução da função.

```
SET SESSION AUTHORIZATION regular_user;
```

```
CREATE EXTERNAL FUNCTION exfunc_sum(INT,INT) RETURNS INT STABLE
LAMBDA 'lambda_sum'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-Exfunc-Test';

CREATE TABLE t_sum(c1 int, c2 int);
INSERT INTO t_sum VALUES (4,5), (6,7);
SELECT exfunc_sum(c1,c2) FROM t_sum;

-- Switch to super user in order to inspect records in the LUDF SYS view.
SET SESSION AUTHORIZATION super_user;
select * from sys_ludf_detail;
```

Exemplo de resultado:

```
 user_id | transaction_id | query_id | function_oid | function_position | stream_id
 | segment_id | step_id | lambda_function_name |         start_time         |
     end_time           | total_duration | invocations | total_rows | input_bytes |
 output_bytes
---------+----------------+----------+--------------+-------------------+-----------
+------------+---------+----------------------+---------------------------
+---------------------------+----------------+-------------+------------+-------------
+--------------
     100 |           1463 |     1544 |       111055 |                 0 |         0 |
         0 |       2 | lambda_sum           | 2026-01-06 17:23:25.165898 | 2026-01-06
 17:23:25.165898 |            414 |           1 |          2 |         277 |
 18
(1 row)
```

SYS_LOAD_ERROR_DETAIL

Use SYS_LOAD_ERROR_DETAIL para visualizar detalhes de erros de comando COPY. Cada linha representa um comando COPY. Contém tanto comandos COPY em execução como finalizados.

SYS_LOAD_ERROR_DETAIL é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	integer	O identificador do usuário que enviou a cópia.
query_id	bigint	O identificador de consulta da cópia.
transaction_id	bigint	O identificador da transação.
session_id	integer	O identificador do processo que está executando a cópia.
database_name	character(64)	O nome do banco de dados ao qual o usuário estava conectado quando a cópia foi enviada.
table_id	integer	O identificador da tabela.
start_time	timestamp	O horário (UTC) em que a cópia começou.
file_name	character(256)	O caminho completo do arquivo de entrada a ser carregado.
line_number	bigint	O número da linha no arquivo de carreamento com o erro. Quando você carrega o arquivo JSON, o número da linha da última linha do objeto JSON com o erro.
column_name	character(127)	O campo com o erro.

Nome da coluna	Tipo de dados	Descrição
column_type	character(10)	O tipo de dados do campo com o erro.
column_length	character(10)	O tamanho da coluna, se aplicável. Este campo é preenchido quando o tipo de dados tem um limite de tamanho. Por exemplo, uma coluna com um tipo de dados “character(3)” contém o valor “3”.
position	integer	A posição do erro no campo.
error_code	integer	O código do erro.
error_message	character(512)	A explicação do erro.

Consultas de exemplo

A consulta a seguir mostra os detalhes do erro de carregamento do comando de cópia para uma consulta específica.

```
SELECT query_id,
       table_id,
       start_time,
       trim(file_name) AS file_name,
       trim(column_name) AS column_name,
       trim(column_type) AS column_type,
       trim(error_message) AS error_message
FROM sys_load_error_detail
WHERE query_id = 762949
ORDER BY start_time
LIMIT 10;
```

Exemplo de resultado.

```
 query_id | table_id |         start_time         |               file_name
      | column_name | column_type |                 error_message
----------+----------+----------------------------
+-----------------------------------------+------------+------------
+-----------------------------------------------
   762949 |   137885 | 2022-02-15 22:14:46.759151 | s3://load-test/copyfail/
wrong_format_000 | id          | int4        | Invalid digit, Value 'a', Pos 0, Type:
 Integer
   762949 |   137885 | 2022-02-15 22:14:46.759151 | s3://load-test/copyfail/
wrong_format_001 | id          | int4        | Invalid digit, Value 'a', Pos 0, Type:
 Integer
```

SYS_LOAD_HISTORY

Use SYS_LOAD_HISTORY para visualizar detalhes dos comandos COPY. Cada linha representa um comando COPY com estatísticas acumuladas para alguns dos campos. Contém tanto comandos COPY em execução como finalizados.

SYS_LOAD_HISTORY é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	integer	O identificador do usuário que enviou a cópia.
query_id	bigint	O identificador de consulta da cópia.
transaction_id	bigint	O identificador da transação.
session_id	integer	O identificador do processo que está executando a cópia.
database_name	texto	O nome do banco de dados ao qual o usuário estava

Nome da coluna	Tipo de dados	Descrição
		conectado quando a operação foi enviada.
status	texto	O status da cópia. Os valores válidos são `running`, `completed` , `aborted`.
table_name	texto	O nome da tabela que está recebendo a cópia.
start_time	timestamp	O horário em que a cópia começou.
end_time	timestamp	O horário em que a cópia foi concluída.
duration	bigint	A quantidade de tempo (em microssegundos) transcorrida no comando COPY.
data_source	texto	O local do Amazon S3 de entrada de arquivos para cópia.
file_format	texto	O formato do arquivo de origem. Os formatos incluem csv, txt, json, avro, orc ou parquet.
loaded_rows	bigint	O número de linhas copiadas para uma tabela.
loaded_bytes	bigint	O número de bytes copiados para uma tabela.

Nome da coluna	Tipo de dados	Descrição
source_file_count	integer	O número da contagem de arquivos nos arquivos de origem.
source_file_bytes	bigint	O número de bytes nos arquivos de origem.
file_count_scanned	integer	O número de arquivos processados na varredura do Amazon S3.
file_bytes_scanned	bigint	O número de bytes processados na varredura do arquivo no Amazon S3.
error_count	bigint	A contagem do número de erros.
copy_job_id	bigint	O identificador do trabalho de cópia. `0` indica que não há nenhum identificador de trabalho.

Consultas de exemplo

A consulta a seguir mostra as linhas carregadas, bytes, tabelas e fonte de dados de comandos de cópia específicos.

```
SELECT query_id,
       table_name,
       data_source,
       loaded_rows,
       loaded_bytes
FROM sys_load_history
WHERE query_id IN (6389,490791,441663,74374,72297)
ORDER BY query_id,
         data_source DESC;
```

Exemplo de resultado.

```
 query_id |    table_name    |                                 data_source
              | loaded_rows | loaded_bytes
----------+------------------
+-------------------------------------------------------------------------+-------------
+---------------
     6389 | store_returns    | s3://load-test/data-sources/tpcds/2.8.0/textfile/1T/
store_returns/    |   287999764 | 1196240296158
    72297 | web_site         | s3://load-test/data-sources/tpcds/2.8.0/textfile/1T/
web_site/         |          54 |         43808
    74374 | ship_mode        | s3://load-test/data-sources/tpcds/2.8.0/textfile/1T/
ship_mode/        |          20 |          1320
   441663 | income_band      | s3://load-test/data-sources/tpcds/2.8.0/textfile/1T/
income_band/      |          20 |          2152
   490791 | customer_address | s3://load-test/data-sources/tpcds/2.8.0/textfile/1T/
customer_address/ |     6000000 |     722924305
```

A consulta a seguir mostra as linhas carregadas, bytes, tabelas e fonte de dados de comandos de cópia.

```
SELECT query_id,
       table_name,
       data_source,
       loaded_rows,
       loaded_bytes
FROM sys_load_history
ORDER BY query_id DESC
LIMIT 10;
```

Exemplo de resultado.

```
 query_id |       table_name       |                                  data_source
                         | loaded_rows |  loaded_bytes
----------+------------------------
+-------------------------------------------------------------------------
+-------------+-----------------
   491058 | web_site               | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/web_site/               |          54 |           43808
   490947 | web_sales              | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/web_sales/              |   720000376 |  22971988122819
```

```
   490923 | web_returns            | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/web_returns/           |    71997522 |     96597496325
   490918 | web_page               | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/web_page/              |        3000 |            1320
   490907 | warehouse              | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/warehouse/             |          20 |            1320
   490902 | time_dim               | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/time_dim/              |       86400 |            1320
   490876 | store_sales            | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/store_sales/           |  2879987999 | 151666241887933
   490870 | store_returns          | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/store_returns/         |   287999764 |   1196405607941
   490865 | store                  | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/store/                 |        1002 |          365507
```

A consulta a seguir mostra as linhas carregadas diariamente e bytes de comando de cópia.

```
SELECT date_trunc('day',start_time) AS exec_day,
       SUM(loaded_rows) AS loaded_rows,
       SUM(loaded_bytes) AS loaded_bytes
FROM sys_load_history
GROUP BY exec_day
ORDER BY exec_day DESC;
```

Exemplo de resultado.

```
      exec_day       | loaded_rows |   loaded_bytes
---------------------+-------------+------------------
 2022-01-20 00:00:00 |  6347386005 |  258329473070606
 2022-01-19 00:00:00 | 19042158015 |  775198502204572
 2022-01-18 00:00:00 | 38084316030 | 1550294469446883
 2022-01-17 00:00:00 | 25389544020 | 1033271084791724
 2022-01-16 00:00:00 | 19042158015 |  775222736252792
 2022-01-15 00:00:00 | 19834245387 |  798122849155598
 2022-01-14 00:00:00 | 75376544688 | 3077040926571384
```

SYS_MV_REFRESH_HISTORY

Os resultados incluem informações sobre o histórico de atualizações de todas as visualizações materializadas, incluindo visualizações materializadas do Iceberg criadas com USING ICEBERG. Os resultados incluem o tipo de atualização, como manual ou automática, e o status da atualização mais recente.

SYS_MV_REFRESH_HISTORY é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Para visualizações materializadas do Iceberg, a coluna `mv_id` é NULL. Use as colunas `mv_name`, `schema_name` e `database_name` para identificar a visualização materializada. A coluna `schema_name` contém o nome do banco de dados AWS Glue para visualizações materializadas do Iceberg.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	integer	O identificador do usuário que enviou a atualização.
session_id	integer	O identificador do processo que executa a atualização da visão materializada.
transaction_id	bigint	O identificador da transação.
database_name	char(128)	O banco de dados que contém a visão materializada.
schema_name	char(128)	O esquema da visão materializada.
mv_id	bigint	O ID do objeto da visão materializada. Para visualizações materializadas do Iceberg, esse valor é NULL porque as visualizações materializadas do Iceberg não estão registradas no catálogo interno do Amazon Redshift.
mv_name	char(128)	O nome da visão materializada.

Nome da coluna	Tipo de dados	Descrição
refresh_type	char(32)	O tipo de atualização, como manual ou automática.
status	texto	O status da atualização. Para obter informações detalhadas sobre status, consulte a coluna de status de SVL_MV_REFRESH_STATUS.
start_time	timestamp	A hora de início da atualização.
end_time	timestamp	A hora de término da atualização.
duration	bigint	A quantidade de tempo, em microssegundos, necessária para atualizar a visão materializada.
consumer_account	char(12)	O ID da conta da AWS do cluster de consumidor que iniciou a atualização. Esse campo é preenchido quando a atualização é iniciada em um cluster de consumidor em uma configuração de compartilhamento de dados.

Nome da coluna	Tipo de dados	Descrição
consumer_region	char(32)	A região da AWS do cluster de consumidor que iniciou a atualização. Esse campo é preenchido quando a atualização é iniciada em um cluster de consumidor em uma configuração de compartilhamento de dados.
consumer_namespace	char(36)	O identificador de namespace do cluster de consumidor que iniciou a atualização. Esse campo é preenchido quando a atualização é iniciada em um cluster de consumidor em uma configuração de compartilhamento de dados.

Consultas de exemplo

A consulta a seguir mostra o histórico de atualização das visões materializadas.

```
SELECT user_id,
     session_id,
     transaction_id,
     database_name,
     schema_name,
     mv_id,
     mv_name,
     refresh_type,
     status,
     start_time,
     end_time,
     duration,
     consumer_account,
     consumer_region,
     consumer_namespace
```

```
    from sys_mv_refresh_history;
```

A consulta retorna os seguintes dados de saída de exemplo:

```
 user_id | session_id | transaction_id | database_name | schema_name                |
 mv_id  |  mv_name           |  refresh_type  |  status
                                                                  |  start_time
     |  end_time                  |  duration | consumer_account | consumer_region |
 consumer_namespace
---------+------------+----------------+--------------
+---------------------------+--------+--------------------+---------------
+-------------------------------------------------------------------------------------------------
+---------------------------+---------------------------+-----------
+------------------+-----------------+-------------------------------------
       1 | 1073815659 |          15066 | dev           | test_stl_mv_refresh_schema
 | 203762 | mv_incremental     | Manual         | MV was already updated
                                                                  | 2023-10-26
 15:59:20.952179 | 2023-10-26 15:59:20.952866 |      687 |                  |
       |
       1 | 1073815659 |          15068 | dev           | test_stl_mv_refresh_schema
 | 203771 | mv_nonincremental  | Manual         | MV was already updated
                                                                  | 2023-10-26
 15:59:21.008049 | 2023-10-26 15:59:21.008658 |      609 |                  |
       |
       1 | 1073815659 |          15070 | ext_db        | producer_schema
  | 203779 | producer_mv        | Manual         | Refresh successfully updated MV
 incrementally                                                    | 2023-10-26
 15:59:21.064252 | 2023-10-26 15:59:21.064885 |      633 | 0123456789       | us-east-1
       | 623d8ff2-4391-4381-83d7-177caa6767af
       1 | 1073815659 |          15074 | dev           | test_stl_mv_refresh_schema
 | 203762 | mv_incremental     | Manual         | Refresh successfully updated MV
 incrementally                                                    | 2023-10-26
 15:59:29.693329 | 2023-10-26 15:59:43.482842 | 13789513 |                  |
       |
       1 | 1073815659 |          15076 | dev           | test_stl_mv_refresh_schema
 | 203771 | mv_nonincremental  | Manual         | Refresh successfully recomputed
 MV from scratch                                                  | 2023-10-26
 15:59:43.550184 | 2023-10-26 15:59:47.880833 |  4330649 |                  |
       |
       1 | 1073815659 |          15078 | dev           | test_stl_mv_refresh_schema
 | 203779 | mv_refresh_error   | Manual         | Refresh failed due to an internal
 error                                                          | 2023-10-26
```

```
 15:59:47.949052 | 2023-10-26 15:59:52.494681 |  4545629 |                 |
       |
       1 | 1073815659 |          15071 | dev           | test_stl_mv_refresh_schema
 | 203778 | mv_test            | Manual         | Cascade refresh failed because
 materialized view test_stl_mv_refresh_schema.child was not refreshed. | 2023-10-26
 15:30:21.432252 | 2023-10-26 15:30:21.432252 |      532 |                 |
       |
       1 | 1073815659 |          15071 | dev           | test_stl_mv_refresh_schema
 | 203761 | child              | Manual         | Refresh failed due to an internal
 error.                                                                | 2023-10-26
 15:30:21.432252 | 2023-10-26 15:30:21.432252 |      532 |                 |
       |
       1 | 1073815659 |          15069 | dev           | test_stl_mv_refresh_schema
 | 203778 | mv_test            | Manual         | Cascade refresh skipped because
 materialized view test_stl_mv_refresh_schema.child was not refreshed.| 2023-10-26
 15:21:43.550369 | 2023-10-26 15:21:43.550369 |      633
       1 | 1073815659 |          15069 | dev           | test_stl_mv_refresh_schema
 | 203761 | child              | Manual         | Refresh failed due to an internal
 error.                                                                | 2023-10-26
 15:21:43.550369 | 2023-10-26 15:21:43.550369 |      633
(10 rows)
```

SYS_MV_STATE

Os resultados incluem informações sobre o estado de todas as visões materializadas. Inclui informações da tabela base, propriedades do esquema e informações sobre eventos recentes, como descartar uma coluna.

SYS_MV_STATE é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	bigint	O ID do usuário que criou o evento.
transaction_id	bigint	O ID da transação do evento.

Nome da coluna	Tipo de dados	Descrição
database_name	char(128)	O banco de dados que contém a visão materializada.
event_desc	char(500)	O evento que acionou a alteração de estado. Exemplos de valores possíveis incluem o seguinte: • O tipo de coluna foi alterado • A coluna foi descartada • A coluna foi renomeada • O nome do esquema foi alterado • Conversão de tabela pequena • TRUNCATE • Vácuo Observe que há outros valores possíveis para essa coluna.
start_time	timestamp	A hora de início do evento.
base_table_database_name	char(128)	O nome do banco de dados da tabela base.
base_table_schema	char(128)	O esquema da tabela base.
base_table_name	char(128)	O nome da tabela base.
mv_schema	char(128)	O esquema da visão materiali zada.
mv_name	char(128)	O nome da visualização materializada.

Nome da coluna	Tipo de dados	Descrição
estado	character(32)	O estado alterado da visão materializada, como segue: • Computar novamente • Não atualizável

Consultas de exemplo

A consulta a seguir mostra o estado da visão materializada.

```
select * from sys_mv_state;
```

A consulta retorna os seguintes dados de saída de exemplo:

```
 user_id | transaction_id | database_name | event_desc                  | start_time
               | base_table_database_name | base_table_schema | base_table_name     |
 mv_schema  | mv_name                    | state
---------+----------------+---------------+-----------------------------
+-----------------------------+--------------------------+-------------------
+---------------------+-------------+-----------------------------+--------------
 106     | 12720          | tickit_db     | TRUNCATE                    | 2023-07-26
 14:59:12.788268 | tickit_db                | mv_schema         | test_table_95d6d861 |
 mv_schema   | materialized_view_a1f3f862 | Recompute
 106     | 12724          | tickit_db     | ALTER TABLE ALTER DISTSTYLE | 2023-07-26
 14:59:51.409014 | tickit_db                | mv_schema         | test_table_58102435 |
 mv_schema   | materialized_view_ca746631 | Recompute
 106     | 12720          | tickit_db     | Column was renamed          | 2023-07-26
 14:59:12.822928 | tickit_db                | mv_schema         | test_table_95d6d861 |
 mv_schema   | materialized_view_5750a8d4 | Unrefreshable
 106     | 12727          | tickit_db     | Table was renamed           | 2023-07-26
 15:00:08.051244 | tickit_db                | mv_schema         | test_table_95d6d861 |
 mv_schema   | materialized_view_5750a8d4 | Unrefreshable
 106     | 12720          | tickit_db     | Column was renamed          | 2023-07-26
 14:59:12.857755 | tickit_db                | mv_schema         | test_table_95d6d861 |
 mv_schema   | materialized_view_5750a8d4 | Unrefreshable
 106     | 12727          | tickit_db     | Table was renamed           | 2023-07-26
 15:00:08.051358 | tickit_db                | mv_schema         | test_table_95d6d861 |
 mv_schema   | materialized_view_5ef0d754 | Unrefreshable
```

```
106     | 12720          | tickit_db     | TRUNCATE                    | 2023-07-26
14:59:12.788159 | tickit_db                | mv_schema         | test_table_95d6d861 |
mv_schema   | materialized_view_5750a8d4 | Recompute
106     | 12720          | tickit_db     | Column was renamed          | 2023-07-26
14:59:12.857799 | tickit_db                | mv_schema         | test_table_95d6d861 |
mv_schema   | materialized_view_a1f3f862 | Unrefreshable
106     | 12720          | tickit_db     | TRUNCATE                    | 2023-07-26
14:59:12.788327 | tickit_db                | mv_schema         | test_table_95d6d861 |
mv_schema   | materialized_view_5ef0d754 | Recompute
106     | 12727          | tickit_db     | ALTER TABLE ALTER SORTKEY   | 2023-07-26
15:00:08.006235 | tickit_db                | mv_schema         | test_table_58102435 |
mv_schema   | materialized_view_ca746631 | Recompute
106     | 12720          | tickit_db     | Column was renamed          | 2023-07-26
14:59:12.82297  | tickit_db                | mv_schema         | test_table_95d6d861 |
mv_schema   | materialized_view_a1f3f862 | Unrefreshable
106     | 12727          | tickit_db     | Table was renamed           | 2023-07-26
15:00:08.051321 | tickit_db                | mv_schema         | test_table_95d6d861 |
mv_schema   | materialized_view_a1f3f862 | Unrefreshable
```

SYS_PROCEDURE_CALL

Use a visão SYS_PROCEDURE_CALL para obter informações sobre as chamadas de procedimento armazenado, inclusive a hora de início, a hora de término, o status de uma chamada de procedimento armazenado e a hierarquia de chamadas de procedimento armazenado aninhado. Cada chamada de procedimento armazenado recebe um ID de consulta.

SYS_PROCEDURE_CALL está visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
session_user_id	integer	O identificador do usuário que criou a sessão e é o invocador da chamada de procedimento armazenado de nível superior.
security_user_id	integer	O identificador do usuário cujos privilégios foram usados

Nome da coluna	Tipo de dados	Descrição
		para executar a instrução dentro do procedimento armazenado. Se o procedimento armazenado tiver sido chamado do procedimento armazenado DEFINER externo, esse será o user_id proprietário do procedimento armazenado externo.
query_id	integer	O identificador da consulta da chamada do procedimento armazenado.
query_text	char(4.000)	O texto da consulta de chamada de procedimento.
start_time	timestamp	A hora de início (em UTC) da execução da consulta. O timestamp usa seis dígitos de precisão para segundos fracionários, por exemplo: 2009-06-12 11:29:19.131358.
end_time	timestamp	A hora de término (em UTC) da execução da consulta. O timestamp usa seis dígitos de precisão para segundos fracionários, por exemplo: 2009-06-12 11:29:19.131358.

Nome da coluna	Tipo de dados	Descrição
status	char(10)	O status da chamada do procedimento armazenado. Quando o procedimento armazenado foi interrompido pelo sistema ou cancelado pelo usuário, o valor foi cancelado. Se procedimento armazenado for concluído, o valor será sucesso.
caller_procedure_query_id	integer	Se o procedimento armazenado tiver sido invocado por outro procedimento armazenado, essa coluna terá o ID de consulta da chamada externa. Caso contrário, o campo será NULL.
query_uuid	character(36)	O identificador globalmente exclusivo (UUID) da consulta.

Consultas de exemplo

A consulta a seguir retorna uma hierarquia de chamadas de procedimento armazenado aninhado.

```
select query_id, datediff(seconds, start_time, end_time) as elapsed_time, status,
 trim(query_text) as call, caller_procedure_query_id from sys_procedure_call;
```

Exemplo de resultado.

```
 query_id | elapsed_time | status  |                      call                      |
 caller_procedure_query_id
----------+--------------+---------+------------------------------------------------
+---------------------------
     3087 |           18 | success | CALL proc_bd906c98c45443ffa165e9552056902d(1)    |
          3085
```

```
     3085 |          18 | success | CALL proc_bd906c98c45443ffa165e9552056902d_2(1); |

(2 rows)
```

SYS_PROCEDURE_MESSAGES

SYS_PROCEDURE_MESSAGES está visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
transaction_id	bigint	O identificador da transação.
query_id	integer	O identificador da consulta da chamada do procedimento armazenado.
record_time	timestamp	A hora em UTC em que a mensagem foi gerada.
log_level	char(10)	O nível de log da mensagem gerada. Os valores possíveis são LOG, INFO, NOTICE, WARNING e EXCEPTION.
message	char(1024)	O texto da mensagem gerada.
line_number	integer	O número da linha da mensagem gerada.
query_uuid	character(36)	O identificador globalmente exclusivo (UUID) da consulta.

Consultas de exemplo

A consulta a seguir mostra um exemplo de saída de SYS_PROCEDURE_MESSAGES.

```
select transaction_id, query_id, record_time, log_level, trim(message), line_number
 from sys_procedure_messages;
```

```
transaction_id | query_id |        record_time         | log_level |           btrim
        | line_number
---------------+----------+----------------------------+-----------
+--------------------------+-------------
     25267     |   80562  | 2023-07-17 14:38:31.910136 |   NOTICE  |
 test_notice_msg_b9f1e749  |     8
     25267     |   80562  | 2023-07-17 14:38:31.910002 |    LOG    |
 test_log_msg_833c7420     |     6
     25267     |   80562  | 2023-07-17 14:38:31.910111 |    INFO   |
 test_info_msg_651373d9    |     7
     25267     |   80562  | 2023-07-17 14:38:31.910154 |   WARNING |
 test_warning_msg_831c5747 |     9
(4 rows)
```

SYS_QUERY_DETAIL

Use SYS_QUERY_DETAIL para visualizar detalhes de consultas em vários níveis de métrica, com cada linha representando detalhes sobre uma consulta do WLM específica em um determinado nível de métrica. Essa visualizações contém muitos tipos de consultas, como DDL, DML e comandos de utilitário (por exemplo, copiar e descarregar). Algumas colunas podem não ser relevantes, conforme o tipo de consulta. Por exemplo, external_scanned_bytes não é relevante para tabelas internas.

SYS_QUERY_HISTORY é visível a todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> Para verificar se uma transação que contém a consulta executada foi confirmada com êxito, é necessário realizar uma operação de junção entre as tabelas do sistema e a tabela `sys_transaction_history`. Por exemplo:
>
> ```
> SELECT
>     th.transaction_id,
>     qd.query_id,
>     th.status AS transaction_status
> FROM
> ```

```
    sys_query_detail qd
LEFT JOIN sys_query_history qh ON qd.query_id = qh.query_id
LEFT JOIN sys_transaction_history th on qh.transaction_id = th.transaction_id;
```

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	integer	O identificador do usuário que enviou a consulta.
query_id	bigint	O identificador da consulta.
child_query_sequence	integer	A sequência da consulta de usuário regravada, começando com 1.
stream_id	integer	O identificador do fluxo de execução da consulta.
segment_id	integer	O identificador do segmento da consulta.
step_id	integer	O identificador da etapa em um segmento.
step_name	texto	O nome da etapa em um segmento. Os valores possíveis são `aggregate` , `broadcast` , `delete`, `distribute` , `hash`, `hashjoin`, `insert`, `limit`, `merge`, `nestloop`, `parse`, `return`, `save`, `scan`, `sort`, `sortlimit` , `unique` e `window`.

Nome da coluna	Tipo de dados	Descrição
table_id	integer	O identificador de tabela para verificações de tabela permanentes.
table_name	character(136)	O nome da tabela da etapa que está sendo operada.
is_rrscan	caractere	Um valor que indica se a etapa é uma etapa de verificação. True (t) indica que a verificação restrita ao intervalo foi utilizada na etapa.
start_time	timestamp	O horário em que a etapa da consulta começou. Esse campo é registrado no nível do segmento, independentemente do valor na coluna `metrics_level` .
end_time	timestamp	O horário em que a etapa da consulta foi concluída. Esse campo é registrado no nível do segmento, independentemente do valor na coluna `metrics_level` .
duration	bigint	O tempo (em microssegundos) gasto na etapa. Esse campo é registrado no nível do segmento, independentemente do valor na coluna `metrics_level` .
alerta	texto	A descrição do evento de alerta.

Nome da coluna	Tipo de dados	Descrição
input_bytes	bigint	Os bytes de entrada para a etapa atual.
input_rows	bigint	As linhas de entrada para a etapa atual.
output_bytes	bigint	Os bytes de saída para a etapa atual.
output_rows	bigint	As linhas de saída para a etapa atual.
blocks_read	bigint	O número de blocos lidos pela etapa.
blocks_write	bigint	O número de blocos gravados pela etapa.
local_read_IO	bigint	O número de leitura de blocos do cache de disco local.
remote_read_IO	bigint	O número de blocos lidos remotamente.
origem	texto	O tipo de objeto de banco de dados que foi verificado. Essa coluna só tem um valor quando o valor da linha step_name é `scan`.
data_skewness	integer	A distorção da distribuição das linhas de saída entre todas as etapas. É um número no intervalo de 0% a 100%. Quanto maior o número, mais a distribuição é desequilibrada.

Nome da coluna	Tipo de dados	Descrição
time_skewness	integer	A distorção da distribuição do tempo de execução entre todas as etapas. É um número no intervalo de 0% a 100%. Quanto maior o número, mais a distribuição é desequili brada.
is_active	caractere	O estado da consulta no nível de etapa. Os valores possíveis são “t”, que mostra que a etapa está sendo executada ativamente, ou “f”, que indica que a etapa foi concluída.
spilled_block_local_disk	bigint	O número de blocos excedentes enviados ao disco local.
spilled_block_remote_disk	bigint	O número de blocos excedentes enviados ao Amazon Simple Storage Service.
step_attribute	character(64)	Contém informações sobre a etapa associada. Valores possíveis para etapas de verificação: `multi-dim ensional` .

Nome da coluna	Tipo de dados	Descrição
metrics_level	character(64)	O nível métrico da consulta. Os valores possíveis são: • Consulta secundária • Fluxo • Segmento • Etapa
plan_parent_id	integer	O identificador do nó principal do nó do plano. Um nó principal pode ter vários nós secundários. Por exemplo, uma junção de mesclagem é o nó principal das verificações nas tabelas unidas.
plan_node_id	integer	O identificador de um nó do plano associado a uma ou mais etapas na consulta.
query_uuid	character(36)	O identificador globalmente exclusivo (UUID) da consulta.

Observações de uso

SYS_QUERY_DETAIL pode conter métricas no nível de etapa, fluxo, segmento e consulta secundária. Além de usar a coluna metrics_level como referência, você pode visualizar os campos step_id, segment_id e stream_id de acordo com a tabela a seguir para ver qual nível de métrica uma determinada linha está mostrando.

Nível de métrica	valor de stream_id	valor de segment_id	valor de step_id
consulta secundária	-1	-1	-1

Nível de métrica	valor de stream_id	valor de segment_id	valor de step_id
transmissão	Um valor de etapa válido	-1	-1
segment	Um valor de etapa válido	Um valor de etapa válido	-1
etapa	Um valor de etapa válido	Um valor de etapa válido	Um valor de etapa válido

Consultas de exemplo

O exemplo a seguir retorna a saída de SYS_QUERY_DETAIL.

A consulta a seguir mostra os detalhes dos metadados da consulta no nível da etapa, incluindo o nome da etapa, input_bytes, output_bytes, input_rows, output_rows.

```
SELECT query_id,
       child_query_sequence,
       stream_id,
       segment_id,
       step_id,
       trim(step_name) AS step_name,
       duration,
       input_bytes,
       output_bytes,
       input_rows,
       output_rows
FROM sys_query_detail
WHERE query_id IN (193929)
ORDER BY query_id,
         stream_id,
         segment_id,
         step_id DESC;
```

Exemplo de resultado.

```
 query_id | child_query_sequence | stream_id | segment_id | step_id | step_name  |
 duration     | input_bytes | output_bytes | input_rows | output_rows
```

```
----------+----------------------+-----------+------------+---------+------------
+-----------------+-------------+--------------+------------+-------------
   193929 |                    2 |         0 |          0 |       3 | hash       |
      37144 |           0 |      9350272 |          0 |      292196
   193929 |                    5 |         0 |          0 |       3 | hash       |
       9492 |           0 |        23360 |          0 |        1460
   193929 |                    1 |         0 |          0 |       3 | hash       |
      46809 |           0 |      9350272 |          0 |      292196
   193929 |                    4 |         0 |          0 |       2 | return     |
       7685 |           0 |          896 |          0 |         112
   193929 |                    1 |         0 |          0 |       2 | project    |
      46809 |           0 |            0 |          0 |      292196
   193929 |                    2 |         0 |          0 |       2 | project    |
      37144 |           0 |            0 |          0 |      292196
   193929 |                    5 |         0 |          0 |       2 | project    |
       9492 |           0 |            0 |          0 |        1460
   193929 |                    3 |         0 |          0 |       2 | return     |
      11033 |           0 |        14336 |          0 |         112
   193929 |                    2 |         0 |          0 |       1 | project    |
      37144 |           0 |            0 |          0 |      292196
   193929 |                    1 |         0 |          0 |       1 | project    |
      46809 |           0 |            0 |          0 |      292196
   193929 |                    5 |         0 |          0 |       1 | project    |
       9492 |           0 |            0 |          0 |        1460
   193929 |                    3 |         0 |          0 |       1 | aggregate  |
      11033 |           0 |       201488 |          0 |          14
   193929 |                    4 |         0 |          0 |       1 | aggregate  |
       7685 |           0 |        28784 |          0 |          14
   193929 |                    5 |         0 |          0 |       0 | scan       |
       9492 |           0 |        23360 |     292196 |        1460
   193929 |                    4 |         0 |          0 |       0 | scan       |
       7685 |           0 |         1344 |        112 |         112
   193929 |                    2 |         0 |          0 |       0 | scan       |
      37144 |           0 |      7304900 |     292196 |      292196
   193929 |                    3 |         0 |          0 |       0 | scan       |
      11033 |           0 |        13440 |        112 |         112
   193929 |                    1 |         0 |          0 |       0 | scan       |
      46809 |           0 |      7304900 |     292196 |      292196
   193929 |                    5 |         0 |          0 |      -1 |            |
       9492 |       12288 |            0 |          0 |          0
   193929 |                    1 |         0 |          0 |      -1 |            |
      46809 |       16384 |            0 |          0 |          0
   193929 |                    2 |         0 |          0 |      -1 |            |
      37144 |       16384 |            0 |          0 |          0
```

```
   193929 |                   4 |         0 |          0 |      -1 |           |
     7685 |       28672 |           0 |          0 |          0
   193929 |                   3 |         0 |          0 |      -1 |           |
    11033 |      114688 |           0 |          0 |          0
```

Para visualizar as tabelas no banco de dados na ordem das mais para as menos usadas, use o exemplo a seguir. Substitua *sample_data_dev* por seu próprio banco de dados. Observe que essa consulta contará as consultas a partir da criação do cluster, mas os dados de visualização do sistema não serão salvos quando faltar espaço no data warehouse.

```
SELECT table_name, COUNT (DISTINCT query_id)
FROM SYS_QUERY_DETAIL
WHERE table_name LIKE 'sample_data_dev%'
GROUP BY table_name
ORDER BY COUNT(*) DESC;

+--------------------------------+-------+
|           table_name           | count |
+--------------------------------+-------+
| sample_data_dev.tickit.venue   |     4 |
| sample_data_dev.myunload1.venue |     3 |
| sample_data_dev.tickit.listing  |     1 |
| sample_data_dev.tickit.category |     1 |
| sample_data_dev.tickit.users    |     1 |
| sample_data_dev.tickit.date     |     1 |
| sample_data_dev.tickit.sales    |     1 |
| sample_data_dev.tickit.event    |     1 |
+--------------------------------+-------+
```

O exemplo a seguir mostra os vários níveis de métrica para uma única consulta do WLM.

```
SELECT query_id, child_query_sequence, stream_id, segment_id, step_id, step_name,
 start_time, end_time, metrics_level
FROM sys_query_detail
WHERE query_id = 1553 AND step_id = -1
ORDER BY stream_id, segment_id, step_id;

 query_id | child_query_sequence | stream_id | segment_id | step_id | step_name |
   start_time         |          end_time          | metrics_level
----------+----------------------+-----------+------------+---------+-----------
+----------------------------+----------------------------+---------------
```

```
    1553 |                    1 |        -1 |         -1 |      -1 |           |
 2024-10-17 02:28:49.814721 | 2024-10-17 02:28:49.847838 | child query
    1553 |                    1 |         0 |         -1 |      -1 |           |
 2024-10-17 02:28:49.814721 | 2024-10-17 02:28:49.835609 | stream
    1553 |                    1 |         0 |          0 |      -1 |           |
 2024-10-17 02:28:49.824677 | 2024-10-17 02:28:49.830372 | segment
    1553 |                    1 |         1 |         -1 |      -1 |           |
 2024-10-17 02:28:49.835624 | 2024-10-17 02:28:49.845773 | stream
    1553 |                    1 |         1 |          1 |      -1 |           |
 2024-10-17 02:28:49.84088  | 2024-10-17 02:28:49.842388 | segment
    1553 |                    1 |         1 |          2 |      -1 |           |
 2024-10-17 02:28:49.835926 | 2024-10-17 02:28:49.844396 | segment
    1553 |                    1 |         2 |         -1 |      -1 |           |
 2024-10-17 02:28:49.846949 | 2024-10-17 02:28:49.847838 | stream
    1553 |                    1 |         2 |          3 |      -1 |           |
 2024-10-17 02:28:49.847013 | 2024-10-17 02:28:49.847485 | segment
(8 rows)
```

SYS_QUERY_EXPLAIN

Exibe o plano EXPLAIN para uma consulta que foi enviada para execução.

SYS_QUERY_TEXT é visível a todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O identificador do usuário que enviou a consulta.
query_id	bigint	O identificador da consulta. As informações detalhadas da consulta são armazenadas em SYS_QUERY_HISTORY.
child_query_sequence	integer	A sequência da consulta de usuário regravada, começando com 1.

Nome da coluna	Tipo de dados	Descrição
plan_node_id	integer	O identificador de um nó do plano associado a uma ou mais etapas na consulta.
plan_parent_id	integer	O identificador do nó principal do nó do plano. Um nó principal pode ter vários nós secundários. Por exemplo, uma junção de mesclagem é o nó principal das verificações nas tabelas unidas.
plan_node	character(400)	O texto do nó de saída de EXPLAIN. Os nós de plano que se referem à execução em nós de computação recebem o prefixo XN na saída EXPLAIN.
plan_info	character(400)	As informações sobre o qualificador e o filtro do nó de plano. Por exemplo, as condições de junção e as restrições da cláusula WHERE estão incluídas nesta coluna.
query_uuid	character(36)	O identificador globalmente exclusivo (UUID) da consulta.

Consultas de exemplo

O exemplo a seguir é o plano EXPLAIN de uma única consulta.

```
SELECT * FROM sys_query_explain WHERE query_id = 612635 ORDER BY plan_node_id;
```

```
 userid | query_id | child_query_sequence | plan_node_id | plan_parent_id |

          plan_node

                                          |

                                                                  plan_info


--------+----------+----------------------+--------------+----------------
+-----------------------------------------------------------------------------------------------------
+-----------------------------------------------------------------------------------------------------
    100 |   612635 |                    1 |            1 |              0 | XN Limit
 (cost=3604047533041.00..3604047533041.25 rows=100 width=20)

                                            |




    100 |   612635 |                    1 |            2 |              1 |   ->  XN
 Merge  (cost=3604047533041.00..3604047533148.02 rows=42809 width=20)

                                             | Merge Key: sum(b.totalprice)




    100 |   612635 |                    1 |            3 |              2 |         ->
 XN Network  (cost=3604047533041.00..3604047533148.02 rows=42809 width=20)

                                          |   Send to leader
```

```
  100 |   612635 |                    1 |           4 |             3 |
 ->  XN Sort  (cost=3604047533041.00..3604047533148.02 rows=42809 width=20)

                                        | Sort Key: sum(b.totalprice)

  100 |   612635 |                    1 |           5 |             4 |
          ->  XN HashAggregate  (cost=2604047529640.76..2604047529747.78 rows=42809
width=20)

                                              |

  100 |   612635 |                    1 |           6 |             5 |
                ->  XN Hash Join DS_DIST_NONE  (cost=15104956.16..2602364653507.34
rows=336575226684 width=20)

                                                     | Hash Cond: (("outer".listid =
"inner".listid) AND ("outer".sellerid = "inner".sellerid))

  100 |   612635 |                    1 |           7 |             6 |
                          ->  XN Seq Scan on listing b  (cost=0.00..7884677.12
rows=788467712 width=24)

                                                       |
```

```
   100 |   612635 |                    1 |           8 |             6 |
                  ->  XN Hash  (cost=7063797.76..7063797.76 rows=706379776 width=8)

                                        |

   100 |   612635 |                    1 |           9 |             8 |
                             ->  XN Seq Scan on sales a  (cost=0.00..7063797.76
 rows=706379776 width=8)

                                                  |

(9 rows)
```

SYS_QUERY_HISTORY

Use SYS_QUERY_HISTORY para visualizar detalhes das consultas do usuário. Cada linha representa uma consulta de usuário com estatísticas acumuladas para alguns dos campos. Essa visualizações contêm muitos tipos de consultas, como linguagem de definição de dados (DDL), linguagem de manipulação de dados (DML), cópia, descarregamento e Amazon Redshift Spectrum. Contém tanto consultas em execução como finalizadas.

SYS_QUERY_HISTORY é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> Para verificar se uma transação que contém a consulta executada foi confirmada com êxito, é necessário realizar uma operação de junção entre as tabelas do sistema e a tabela `sys_transaction_history`. Por exemplo:

```
SELECT
    qh.transaction_id,
    qh.query_id,
    qh.status AS query_status,
    qh.query_type,
    TRIM(qh.query_text) AS query_text,
    th.status AS transaction_status
FROM
    sys_query_history qh
LEFT JOIN
    sys_transaction_history th ON qh.transaction_id = th.transaction_id;
```

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	integer	O identificador do usuário que enviou a consulta.
query_id	bigint	O identificador da consulta.
query_label	character(320)	O nome abreviado da consulta.
transaction_id	bigint	O identificador da transação.
session_id	integer	O identificador do processo que está executando a consulta.
database_name	character(128)	O nome do banco de dados ao qual o usuário estava

Nome da coluna	Tipo de dados	Descrição
		conectado quando a consulta foi enviada.
query_type	character(32)	O tipo de consulta, como SELECT, INSERT, UPDATE, UNLOAD COPY, COMMAND, DDL, UTILITY, CTAS e OTHER.
status	character(10)	O status da consulta. Valores válidos: planning, queued, running, returning, failed, canceled e success.
result_cache_hit	Booleano	Indica se a consulta correspon de ao cache de resultados.
start_time	timestamp	O horário em que a consulta começou.
end_time	timestamp	O horário em que a consulta foi concluída.
elapsed_time	bigint	O tempo total (em microsseg undos) transcorrido da consulta.
queue_time	bigint	O tempo total (em microsseg undos) transcorrido na fila de consultas da classe de serviço.
execution_time	bigint	O tempo total (em microsseg undos) em execução na classe de serviço.

Nome da coluna	Tipo de dados	Descrição
error_message	character(512)	O motivo por que uma consulta falhou.
returned_rows	bigint	O número de linhas retornadas ao cliente.
returned_bytes	bigint	O número de bytes retornados ao cliente.
query_text	character(4000)	A string de consulta. Essa string poderá estar truncada.
redshift_version	character(256)	A versão do Amazon Redshift quando a consulta foi executada.
usage_limit	character(150)	Lista de IDs de limite de uso atingidos pela consulta.
compute_type	varchar(32)	Indica se a consulta é executada no cluster principal ou em um cluster de escalabilidade da simultaneidade. Os valores possíveis são `primary` (a consulta é executada no cluster principal), `secondary` (a consulta é executada no cluster de simultaneidade) ou `primary-scale` (a consulta é executada no cluster de simultaneidade). Isso só se aplica ao cluster provisionado.

Nome da coluna	Tipo de dados	Descrição
compile_time	bigint	O tempo total (em microsseg undos) gasto na compilação da consulta.
planning_time	bigint	O tempo total (em microsseg undos) gasto no planejamento da consulta.
lock_wait_time	bigint	O tempo total (em microsseg undos) gasto aguardando o bloqueio da relação.
service_class_id	integer	O ID da classe de serviço. Para obter uma lista dos IDs de classe de serviço, acesse IDs da classe de serviço do WLM. Essa coluna é usada somente para consultas executadas em clusters provisionados. Em consultas executadas no modo sem servidor, essa coluna contém -1.
service_class_name	character(64)	O nome da classe de serviço. Essa coluna é usada somente para consultas executadas em clusters provisionados. Em consultas executadas no Amazon Redshift sem servidor, essa coluna não contém nenhum valor.

Nome da coluna	Tipo de dados	Descrição
query_priority	character(20)	A prioridade da fila na qual a consulta foi executada. Os valores possíveis são: • NULL • lowest • low • normal • high • highest NULL significa que não é possível usar a prioridade de consulta para a consulta em questão. Essa coluna é usada somente para consultas executadas em clusters provisionados. Em consultas executadas no Redshift sem servidor, essa coluna não contém nenhum valor.

Nome da coluna	Tipo de dados	Descrição
short_query_accelerated	character(10)	Indica se a consulta foi acelerada usando a aceleraçã o de consultas breves (SQA). Os valores possíveis são: • true • false • NULL Essa coluna é usada somente para consultas executadas em clusters provisionados. Em consultas executadas no Redshift sem servidor, essa coluna não contém nenhum valor.
user_query_hash	character(40)	O hash de consulta gerado na consulta, incluindo os respectivos literais. Consultas repetidas com o mesmo texto de consulta terão os mesmos valores user_query_hash.
generic_query_hash	character(40)	O hash de consulta gerado na consulta, excluindo os respectivos literais. Consultas repetidas com o mesmo texto de consulta e diferente s literais terão os mesmos valores user_query_hash.
query_hash_version	integer	O número da versão do hash da consulta gerado na consulta.

Nome da coluna	Tipo de dados	Descrição
result_cache_query_id	integer	Se a consulta tiver usado o armazenamento em cache do resultado, o valor desse campo será o ID da consulta que originou os resultados armazenados em cache. Se o armazenamento em cache de resultados não foi usado, o valor desse campo é `0`.
username	character(128)	O nome do usuário que enviou a consulta.
query_uuid	character(36)	O identificador globalmente exclusivo (UUID) da consulta.

Consultas de exemplo

A consulta a seguir retorna consultas em execução e enfileiradas.

```
SELECT user_id,
       query_id,
       transaction_id,
       session_id,
       status,
       trim(database_name) AS database_name,
       start_time,
       end_time,
       result_cache_hit,
       elapsed_time,
       queue_time,
       execution_time
FROM sys_query_history
WHERE status IN ('running','queued')
ORDER BY start_time;
```

Exemplo de resultado.

```
 user_id | query_id | transaction_id | session_id | status  | database_name |
 start_time         |          end_time          | result_cache_hit | elapsed_time |
 queue_time | execution_time
---------+----------+----------------+------------+---------+---------------
+--------------------------+---------------------------+-----------------
+--------------+-----------+----------------
     101 |   760705 |         852337 | 1073832321 | running | tpcds_1t      |
 2022-02-15 19:03:19.67849 | 2022-02-15 19:03:19.739811 | f               |
 61321 |          0 |              0
```

A consulta a seguir retorna a hora de início, a hora de término, a hora da fila, o tempo decorrido, o tempo de planejamento e outros metadados de uma consulta específica.

```
SELECT user_id,
       query_id,
       transaction_id,
       session_id,
       status,
       trim(database_name) AS database_name,
       start_time,
       end_time,
       result_cache_hit,
       elapsed_time,
       queue_time,
       execution_time,
       planning_time,
       trim(query_text) as query_text
FROM sys_query_history
WHERE query_id = 3093;
```

Exemplo de resultado.

```
user_id | query_id | transaction_id | session_id |   status   | database_name |
   start_time         |          end_time          | result_cache_hit | elapsed_time |
 queue_time | execution_time | planning_time | query_text
--------+----------+----------------+------------+------------
+---------------+----------------------------+---------------------------
+------------------+--------------+------------+----------------+--------------
+-------------------------------------
    106 |     3093 |          11759 | 1073750146 | success    | dev           |
 2023-03-16 16:53:17.840214 | 2023-03-16 16:53:18.106588 | f               |
 266374 |          0 |         105725 |        136589 | select count(*) from item;
```

A consulta a seguir lista as dez consultas SELECT mais recentes.

```
SELECT query_id,
       transaction_id,
       session_id,
       start_time,
       elapsed_time,
       queue_time,
       execution_time,
       returned_rows,
       returned_bytes
FROM sys_query_history
WHERE query_type = 'SELECT'
ORDER BY start_time DESC limit 10;
```

Exemplo de resultado.

```
 query_id | transaction_id | session_id |         start_time         | elapsed_time |
 queue_time | execution_time | returned_rows | returned_bytes
----------+----------------+------------+----------------------------+--------------
+------------+----------------+---------------+----------------
   526532 |          61093 | 1073840313 | 2022-02-09 04:43:24.149603 |       520571 |
        0 |         481293 |              1 |           3794
   526520 |          60850 | 1073840313 | 2022-02-09 04:38:27.24875  |       635957 |
        0 |         596601 |              1 |           3679
   526508 |          60803 | 1073840313 | 2022-02-09 04:37:51.118835 |       563882 |
        0 |         503135 |              5 |          17216
   526505 |          60763 | 1073840313 | 2022-02-09 04:36:48.636224 |       649337 |
        0 |         589823 |              1 |            652
   526478 |          60730 | 1073840313 | 2022-02-09 04:36:11.741471 |     14611321 |
        0 |       14544058 |              0 |              0
   526467 |          60636 | 1073840313 | 2022-02-09 04:34:11.91463  |     16711367 |
        0 |       16633767 |              1 |            575
   511617 |         617946 | 1074009948 | 2022-01-20 06:21:54.44481  |      9937090 |
        0 |        9899271 |            100 |          12500
   511603 |         617941 | 1074259415 | 2022-01-20 06:21:45.71744  |      8065081 |
        0 |        7582500 |            100 |           8889
   511595 |         617935 | 1074128320 | 2022-01-20 06:21:44.030876 |      1051270 |
        0 |        1014879 |              1 |             72
   511584 |         617931 | 1074030019 | 2022-01-20 06:21:42.764088 |       609033 |
        0 |         485887 |            100 |           8438
```

A consulta a seguir mostra a contagem diária de consultas de seleção e o tempo médio decorrido da consulta.

```
SELECT date_trunc('day',start_time) AS exec_day,
       status,
       COUNT(*) AS query_cnt,
       AVG(datediff (microsecond,start_time,end_time)) AS elapsed_avg
FROM sys_query_history
WHERE query_type = 'SELECT'
AND start_time >= '2022-01-14'
AND start_time <= '2022-01-18'
GROUP BY exec_day,
         status
ORDER BY exec_day,
         status;
```

Exemplo de resultado.

```
      exec_day       | status  | query_cnt | elapsed_avg
---------------------+---------+-----------+------------
 2022-01-14 00:00:00 | success |      5253 |  56608048
 2022-01-15 00:00:00 | success |      7004 |  56995017
 2022-01-16 00:00:00 | success |      5253 |  57016363
 2022-01-17 00:00:00 | success |      5309 |  55236784
 2022-01-18 00:00:00 | success |      8092 |  54355124
```

A consulta a seguir mostra o desempenho do tempo decorrido da consulta diária.

```
SELECT distinct date_trunc('day',start_time) AS exec_day,
       query_count.cnt AS query_count,
       Percentile_cont(0.5) within group(ORDER BY elapsed_time) OVER (PARTITION BY
 exec_day) AS P50_runtime,
       Percentile_cont(0.8) within group(ORDER BY elapsed_time) OVER (PARTITION BY
 exec_day) AS P80_runtime,
       Percentile_cont(0.9) within group(ORDER BY elapsed_time) OVER (PARTITION BY
 exec_day) AS P90_runtime,
       Percentile_cont(0.99) within group(ORDER BY elapsed_time) OVER (PARTITION BY
 exec_day) AS P99_runtime,
       Percentile_cont(1.0) within group(ORDER BY elapsed_time) OVER (PARTITION BY
 exec_day) AS max_runtime
FROM sys_query_history
LEFT JOIN (SELECT  date_trunc('day',start_time) AS day, count(*) cnt
```

```
           FROM sys_query_history
           WHERE query_type = 'SELECT'
           GROUP by 1) query_count
ON date_trunc('day',start_time) = query_count.day
WHERE query_type = 'SELECT'
ORDER BY exec_day;
```

Exemplo de resultado.

```
      exec_day       | query_count | p50_runtime | p80_runtime | p90_runtime |
 p99_runtime  | max_runtime
--------------------+-------------+-------------+-------------+-------------
+--------------+-------------
 2022-01-14 00:00:00 |        5253 |  16816922.0 |  69525096.0 | 158524917.8 |
 486322477.52 | 1582078873.0
 2022-01-15 00:00:00 |        7004 |  15896130.5 |  71058707.0 | 164314568.9 |
 500331542.07 | 1696344792.0
 2022-01-16 00:00:00 |        5253 |  15750451.0 |  72037082.2 | 159513733.4 |
 480372059.24 | 1594793766.0
 2022-01-17 00:00:00 |        5309 |  15394513.0 |  68881393.2 | 160254700.0 |
 493372245.84 | 1521758640.0
 2022-01-18 00:00:00 |        8092 |  15575286.5 |  68485955.4 | 154559572.5 |
 463552685.39 | 1542783444.0
 2022-01-19 00:00:00 |        5860 |  16648747.0 |  72470482.6 | 166485138.2 |
 492038228.67 | 1693483241.0
 2022-01-20 00:00:00 |        1751 |  15422072.0 |  69686381.0 | 162315385.0 |
 497066615.00 | 1439319739.0
 2022-02-09 00:00:00 |          13 |   6382812.0 |  17616161.6 |  21197988.4 |
 23021343.84 |   23168439.0
```

A consulta a seguir mostra a distribuição do tipo de consulta.

```
SELECT query_type,
       COUNT(*) AS query_count
FROM sys_query_history
GROUP BY query_type
ORDER BY query_count DESC;
```

Exemplo de resultado.

```
 query_type | query_count
```

```
------------+-------------
 UTILITY    |      134486
 SELECT     |       38537
 DDL        |        4832
 OTHER      |         768
 LOAD       |         768
 CTAS       |         748
 COMMAND    |          92
```

O exemplo a seguir mostra a diferença nos resultados do hash de consulta entre várias consultas. Observe as seguintes consultas:

```
CREATE TABLE test_table (col1 INT);

INSERT INTO test_table VALUES (1),(2);

SELECT * FROM test_table;

SELECT * FROM test_table;

SELECT col1 FROM test_table;

SELECT * FROM test_table WHERE col1=1;

SELECT * FROM test_table WHERE col1=2;

SELECT query_id, TRIM(user_query_hash) AS user_query_hash, TRIM(generic_query_hash)
 AS generic_query_hash, TRIM(query_text) AS text FROM sys_query_history ORDER BY
 start_time
DESC LIMIT 10;
```

Este é um exemplo de saída:

```
query_id | user_query_hash | generic_query_hash | text
---------+-----------------+--------------------+----------
24723049 | oPuFtjEPLTs=    | oPuFtjEPLTs=       | select query_id,
 trim(user_query_hash) as user_query_hash, trim(generic_query_hash) as
 generic_query_hash, query_hash_version, trim(query_text) as text from
 sys_query_history order by start_time\r\ndesc limit 20
24723045 | Gw2Kwdd8m2I=    | IwfRu8/XAKI=       | select * from test_table where col1=2
 limit 100
24723041 | LNw2vx0GDXo=    | IwfRu8/XAKI=       | select * from test_table where col1=1
 limit 100
```

```
24723036 | H+qep/c82Y8=    | H+qep/c82Y8=       | select col1 from test_table limit 100
24723033 | H+qep/c82Y8=    | H+qep/c82Y8=       | select * from test_table limit 100
24723029 | H+qep/c82Y8=    | H+qep/c82Y8=       | select * from test_table limit 100
24723023 | 50sirx9E1hU=    | uO36Z1a/QYs=       | insert into test_table values (1),(2)
24723021 | YSVnlivZHeo=    | YSVnlivZHeo=       | create table test_table (col1 int)
```

`SELECT * FROM test_table;` e `SELECT col1 FROM test_table;` têm o mesmo valor user_query_hash, já que test_table tem apenas uma coluna. `SELECT * FROM test_table WHERE col1=1;` e `SELECT * FROM test_table WHERE col1=2;` têm valores user_query_hash diferentes, mas valores generic_query_hash idênticos, já que as duas consultas são idênticas fora dos literais de consulta 1 e 2.

SYS_QUERY_TEXT

Use SYS_QUERY_TEXT para visualizar os textos de todas as consultas. Cada linha representa o texto de consultas com até 4.000 caracteres que começam com o número de sequência 0. Quando a instrução de consulta contém mais de 4.000 caracteres, linhas adicionais são registradas em log para a instrução por meio da incrementação do número de sequência de cada linha. Essa visualização registra em log todos os textos de consultas do usuário, como DDL, utilitário, consultas do Amazon Redshift e consultas somente do nó líder.

SYS_QUERY_TEXT é visível a todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	integer	O identificador do usuário que enviou a consulta.
query_id	bigint	O identificador da consulta.
transaction_id	bigint	O identificador da transação associada à instrução.

Nome da coluna	Tipo de dados	Descrição
session_id	integer	O identificador do processo da sessão que executa a consulta.
start_time	timestamp	A hora de início da consulta.
sequence	integer	Quando uma única instrução contém mais de 4.000 caracteres, são registrad as linhas adicionais para a instrução. O valor 0 da sequência é a primeira linha, 1 é a segunda linha e assim por diante.
texto	character (4000)	O texto da consulta SQL que está em incrementos de 4.000 caracteres. Esse campo pode conter caracteres especiais, como barra invertida (\) e nova linha (\n).
query_uuid	character(36)	O identificador globalmente exclusivo (UUID) da consulta.

Consultas de exemplo

A consulta a seguir retorna consultas em execução e enfileiradas.

```
SELECT user_id,
 query_id,
 transaction_id,
 session_id, start_time,
 sequence, trim(text) as text from sys_query_text
 ORDER BY sequence;
```

Exemplo de resultado.

```
 user_id | query_id | transaction_id | session_id |        start_time          |
 sequence |                                                    text
--------+----------+----------------+------------+----------------------------
+----------
+-------------------------------------------------------------------------------------------------
   100  |     4    |       1396     | 1073750220 | 2023-04-28 16:44:55.887184 |
   0    | SELECT trim(text) as text, sequence FROM sys_query_text WHERE query_id =
 pg_last_query_id() AND user_id > 1 AND start
_time > '2023-04-28 16:44:55.922705+00:00'::timestamp order by sequence;
```

A consulta a seguir retorna as permissões que foram concedidas ou revogadas dos grupos no banco de dados.

```
SELECT
    SPLIT_PART(text, ' ', 1) as grantrevoke,
    SPLIT_PART((SUBSTRING(text, STRPOS(UPPER(text), 'GROUP'))), ' ', 2) as group,
    SPLIT_PART((SUBSTRING(text, STRPOS(UPPER(text), ' '))), 'ON', 1) as type,
    SPLIT_PART((SUBSTRING(text, STRPOS(UPPER(text), 'ON'))), ' ', 2) || ' ' ||
 SPLIT_PART((SUBSTRING(text, STRPOS(UPPER(text), 'ON'))), ' ', 3) as entity
FROM SYS_QUERY_TEXT
WHERE (text LIKE 'GRANT%' OR text LIKE 'REVOKE%') AND text LIKE '%GROUP%';

+-------------+----------+--------+----------+
| grantrevoke |  group   |  type  |  entity  |
+-------------+----------+--------+----------+
| GRANT       | bi_group | SELECT | TABLE t1 |
| GRANT       | bi_group | SELECT | TABLE t1 |
| GRANT       | bi_group | SELECT | TABLE t1 |
| GRANT       | bi_group | USAGE  | TABLE t1 |
| GRANT       | bi_group | SELECT | TABLE t1 |
| GRANT       | bi_group | SELECT | TABLE t1 |
+-------------+----------+--------+----------+
```

SYS_REDSHIFT_TEMPLATE

Use SYS_REDSHIFT_TEMPLATE para visualizar detalhes de MODELOS do Redshift.

Esta visualização contém os MODELOS que foram criados.

SYS_REDSHIFT_TEMPLATE pode ser vista por todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
database_name	character(128)	O banco de dados do Redshift no qual o modelo é criado.
schema_name	character(128)	O esquema do Redshift no qual o modelo é criado.
template_name	character(128)	O nome do modelo.
template_type	integer	Um número inteiro que indica o tipo de comando do Redshift associado ao modelo. 1 = comando COPY
create_time	timestamp	O carimbo de data/hora de quando o modelo foi criado.
last_modified_time	timestamp	Carimbo de data/hora da última modificação do modelo.
owner_id	integer	O ID do usuário que possui o modelo.
last_modified_by	integer	O ID do usuário que modificou o modelo pela última vez.
template_parameters	texto	Uma string JSON que contém os parâmetros do modelo e os respectivos valores.

Consultas de exemplo

A seguinte consulta exibe todos os modelos que podem ser vistos pelo usuário atual:

```
SELECT * FROM SYS_REDSHIFT_TEMPLATE;
```

Exemplo de resultado.

```
 database_name | schema_name |   template_name    | template_type |
 create_time         |     last_modified_time     | owner_id | last_modified_by |
 template_parameters
---------------+-------------+--------------------+---------------
+---------------------------+----------------------------+----------
+------------------+---------------------
 dev           | s1          | shapefile_template |             1 | 2025-12-17
 22:42:02.079758 | 2025-12-17 22:42:02.079758 |      101 |              101 | {
    "SIMPLIFY_AUTO": 0.000001,
    "SHAPEFILE": 1,
    "COMPRESSION_UPDATE": 0
}
 dev           | s2          | orc_template       |             1 | 2025-12-17
 22:42:23.582815 | 2025-12-17 22:42:23.582815 |      101 |              101 | {
    "ORC": "serializetojson_default"
}
 dev           | s1          | csv_template       |             1 | 2025-12-17
 22:43:01.822361 | 2025-12-17 22:43:01.822361 |      101 |              101 | {
    "ENCRYPTED": 1,
    "CSV": 1,
    "ENCODING": 1,
    "DELIMITER": ","
}
(3 rows)
```

SYS_RESTORE_LOG

Use SYS_RESTORE_LOG para monitorar o progresso da migração de cada tabela no cluster durante um redimensionamento clássico para nós RG ou RA3. Ele captura o throughput histórico da migração de dados durante a operação de redimensionamento. Para obter mais informações sobre o redimensionamento clássico para nós RG ou RA3, consulte Redimensionamento clássico.

SYS_RESTORE_LOG só permanece visível para superusuários.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
event_time	timestamp	Um carimbo de data e hora que indica quando a entrada do log é registrada.
database_name	char(128)	O nome do banco de dados.
schema_name	char(128)	O nome do esquema.
table_name	char(128)	O nome da tabela.
table_id	integer	O ID da tabela.
ação	char(128)	A ação tomada no momento da entrada. Entre os valores podem estar: migração iniciada, ponto de verificaç ão, retomada, concluída, cancelada ou redefinida.
table_size	long	O tamanho da tabela.
total_data_processed	long	O tamanho dos dados em MB processados até este ponto da tabela.
delta_data_processed	long	Tamanho dos dados processados desde a atualização mais recente de event_time, em MB. Isso ajuda a determinar quanto dos dados foi processado desde o intervalo de tempo registrad o anteriormente. Você pode comparar isso com table_size para ter uma ideia da rapidez

Nome da coluna	Tipo de dados	Descrição
		com que o processamento de dados está sendo realizado.
message	char(512)	Uma explicação detalhada do valor na coluna de ação.
redistribution_type	char(32)	O tipo de redistribuição da tabela. Uma tarefa de conversão de chave ou de rebalanceamento regular. Para obter mais informações sobre estilos de distribuição, consulte Distribution styles.

Consultas de exemplo

A consulta a seguir calcula o throughput do processamento de dados usando SYS_RESTORE_LOG.

```
SELECT
    ROUND(sum(delta_data_processed) / 1024.0, 2) as data_processed_gb,
    ROUND(datediff(sec, min(event_time), max(event_time)) / 3600.0, 2) as duration_hr,
    ROUND(data_processed_gb/duration_hr, 2) as throughput_gb_per_hr
from sys_restore_log;
```

Exemplo de resultado.

```
 data_processed_gb | duration_hr | throughput_gb_per_hr
-------------------+-------------+----------------------
              0.91 |        8.37 |                 0.11
(1 row)
```

A consulta a seguir que mostra todos os tipos de redistribuição.

```
SELECT * from sys_restore_log ORDER BY event_time;
```

```
 database_name |     schema_name      |      table_name      | table_id |
 action             | total_data_processed | delta_data_processed |         event_time
        | table_size | message |   redistribution_type
---------------+----------------------+----------------------+----------
+----------------------------+----------------------+----------------------
+----------------------------+------------+---------+--------------------------
 dev           | schemaaaa877096d844d | customer_key         |   106424 |
 Redistribution started      |                    0 |                       | 2024-01-05
 02:18:00.744977 |        325 |         | Restore Distkey Table
 dev           | schemaaaa877096d844d | dp30907_t2_autokey   |   106430 |
 Redistribution started      |                    0 |                       | 2024-01-05
 02:18:02.756675 |         90 |         | Restore Distkey Table
 dev           | schemaaaa877096d844d | dp30907_t2_autokey   |   106430 |
 Redistribution completed    |                   90 |                    90 | 2024-01-05
 02:23:30.643718 |         90 |         | Restore Distkey Table
 dev           | schemaaaa877096d844d | customer_key         |   106424 |
 Redistribution completed    |                  325 |                   325 | 2024-01-05
 02:23:45.998249 |        325 |         | Restore Distkey Table
 dev           | schemaaaa877096d844d | dp30907_t1_even      |   106428 |
 Redistribution started      |                    0 |                       | 2024-01-05
 02:23:46.083849 |         30 |         | Rebalance Disteven Table
 dev           | schemaaaa877096d844d | dp30907_t5_auto_even |   106436 |
 Redistribution started      |                    0 |                       | 2024-01-05
 02:23:46.855728 |         45 |         | Rebalance Disteven Table
 dev           | schemaaaa877096d844d | dp30907_t5_auto_even |   106436 |
 Redistribution completed    |                   45 |                    45 | 2024-01-05
 02:24:16.343029 |         45 |         | Rebalance Disteven Table
 dev           | schemaaaa877096d844d | dp30907_t1_even      |   106428 |
 Redistribution completed    |                   30 |                    30 | 2024-01-05
 02:24:20.584703 |         30 |         | Rebalance Disteven Table
 dev           | schemaefd028a2a48a4c | customer_even        |   130512 |
 Redistribution started      |                    0 |                       | 2024-01-05
 04:54:55.641741 |        190 |         | Restore Disteven Table
 dev           | schemaefd028a2a48a4c | customer_even        |   130512 |
 Redistribution checkpointed |     29.4342113157737 |     29.4342113157737 | 2024-01-05
 04:55:04.770696 |        190 |         | Restore Disteven Table
(8 rows)
```

SYS_RESTORE_STATE

Use SYS_RESTORE_STATE para monitorar o progresso da migração de cada tabela durante um redimensionamento clássico. Isso se aplica especificamente quando o tipo de nó de destino é RG

ou RA3. Para obter mais informações sobre o redimensionamento clássico para nós RG ou RA3, consulte Redimensionamento clássico.

SYS_RESTORE_STATE só é visível a superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	integer	O identificador do usuário que enviou a consulta.
database_name	char(64)	O nome do banco de dados da tabela.
schema_id	integer	O ID do esquema da tabela.
table_id	integer	O ID da tabela.
table_name	char(128)	O nome da tabela.
redistribution_status	char(128)	O status do progresso da redistribuição da tabela. Os valores possíveis são `Completed` , `In progress` e `Pending`.
percentage_redistributed	flutuação	A porcentagem do progresso da redistribuição da tabela. Os valores possíveis giram entre 0% a 100%. Por exemplo, um valor de 25 indica que 25% dos dados são redistribuídos.
redistribution_type	char(32)	O tipo de redistribuição da tabela. Uma tarefa de conversão de chave ou de rebalanceamento regular. Para obter mais informações

Nome da coluna	Tipo de dados	Descrição
		sobre estilos de distribuição, consulte Distribution styles.

Consultas de exemplo

A consulta a seguir exibe registros de consultas em execução e em fila.

```
SELECT * FROM sys_restore_state;
```

Exemplo de resultado.

```
 userid | database_name | schema_id | table_id |   table_name   | redistribution_status
 | precentage_redistributed |   redistribution_type
--------+---------------+-----------+----------+----------------
+-----------------------+--------------------------+------------------------
    1   |     test1     |   124865  |  124878  | customer_key_4 |         Pending
 |      0                   |  Rebalance Disteven Table
    1   |      dev      |   124865  |  124874  | customer_key_3 |         Pending
 |      0                   |  Rebalance Disteven Table
    1   |      dev      |   124865  |  124870  | customer_key_2 |        Completed
 |     100                  |  Rebalance Disteven Table
    1   |      dev      |   124865  |  124866  | customer_key_1 |       In progress
 |     13.52                |  Restore Distkey Table
```

O seguinte exibe o status do processamento de dados.

```
SELECT
    redistribution_status, ROUND(SUM(block_count) / 1024.0, 2) AS total_size_gb
FROM sys_restore_state sys inner join stv_tbl_perm stv
    on sys.table_id = stv.id
GROUP BY sys.redistribution_status;
```

Exemplo de resultado.

```
 redistribution_status | total_size_gb
-----------------------+---------------
 Completed             |          0.07
 Pending               |          0.71
```

```
 In progress           |          0.20
(3 rows)
```

SYS_SCHEMA_QUOTA_VIOLATIONS

Registra a ocorrência, o ID de transação e outras informações úteis quando uma cota de esquema é excedida. Esta tabela do sistema é uma conversão de STL_SCHEMA_QUOTA_VIOLATIONS.

R_SYS_SYS_SCHEMA_QUOTA_VIOLATIONS permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
owner_id	integer	O ID do proprietário do esquema.
user_id	integer	O ID do usuário que gerou a entrada.
transaction_id	bigint	O ID da transação associada à instrução.
session_id	integer	O ID do processo associado à instrução.
schema_id	integer	O ID do esquema ou namespace.
schema_name	character (128)	O nome do esquema ou namespace.
quota	integer	A quantidade de espaço em disco (em MB) que o esquema pode usar.
disk_usage	integer	O espaço em disco (em MB) atualmente usado pelo esquema.
record_time	time stamp sem fuso horário	O horário de ocorrência da violação.

Consultas de exemplo

A consulta a seguir mostra o resultado de uma violação de cota:

```
SELECT user_id, TRIM(schema_name) "schema_name", quota, disk_usage, record_time FROM
sys_schema_quota_violations WHERE SCHEMA_NAME = 'sales_schema' ORDER BY timestamp DESC;
```

Essa consulta retorna o seguinte exemplo de resultado para o esquema especificado:

```
user_id| schema_name  | quota | disk_usage | record_time
-------+--------------+-------+------------+----------------------------
104    | sales_schema | 2048  | 2798       | 2020-04-20 20:09:25.494723
(1 row)
```

SYS_SERVERLESS_USAGE

Use SYS_SERVERLESS_USAGE para visualizar detalhes do uso de recursos do Amazon Redshift Serverless. Essa visualização do sistema não se aplica a clusters provisionados do Amazon Redshift.

Essa visualização contém o resumo de uso da tecnologia sem servidor, inclusive quanta capacidade computacional é usada para processar consultas e a quantidade de armazenamento gerenciado do Amazon Redshift usado, com granularidade de um minuto. A capacidade computacional é medida em unidades de processamento do Redshift (RPUs) e medida para as workloads executadas em RPU-segundo por segundo. As RPUs são utilizadas para processar consultas nos dados carregados no data warehouse, consultados de um data lake do Amazon S3 ou acessados de bancos de dados operacionais usando uma consulta federada. O Amazon Redshift Serverless retém as informações em SYS_SERVERLESS_USAGE por sete dias.

Para ver exemplos de faturamento de custos de computação, consulte Faturamento do Amazon Redshift Serverless.

SYS_SERVERLESS_USAGE só permanece visível para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
start_time	timestamp	O horário em que o intervalo começou.
end_time	timestamp	O horário em que o intervalo foi concluído.
compute_seconds	double precision	A unidade computaci onal (RPU) por segundo acumulada consumida durante esse intervalo de tempo. Esse valor representa a capacidade de RPU de base alocada para a conta.
compute_capacity	double precision	O número médio de unidades de computação (Unidades de processamento do Redshift, ou RPUs) usadas durante esse intervalo de tempo. O valor compute_capacity pode ser alterado de maneira dinâmica.
data_storage	bigint	O espaço médio de armazenamento de dados usado durante esse intervalo. O armazenamento de dados utilizado pode mudar dinamicamente à medida que os dados são carregados ou excluídos do banco de dados.

Nome da coluna	Tipo de dados	Descrição
cross_region_transferred_data	bigint	Os dados acumulados que são transferidos em bytes para o compartilhamento de dados entre regiões durante esse intervalo de tempo.
charged_seconds	bigint	O acúmulo de segundos de unidade computacional (RPU) consumidos durante esse intervalo de tempo. Isso é calculado após o término das transações, portanto pode ser 0 enquanto uma transação é executada. Use charged_seconds para calcular o custo de um grupo de trabalho do Amazon Redshift Serverless. Esse valor representa a capacidade de RPU alocada para o grupo de trabalho do Amazon Redshift Serverless.

Nome da coluna	Tipo de dados	Descrição
charged_extra_compute_for_automatic_optimization_seconds	bigint	Os segundos acumulados de unidade de computação (RPU) cobrados para otimizações automáticas durante esse intervalo de tempo. Esse valor representa a capacidade de RPU alocada para o grupo de trabalho do Amazon Redshift Serverless. Para ter mais informações sobre otimizações automáticas executadas com recursos extras de computação, consulte Alocação de recursos extras de computação para otimização automática de banco de dados.

Observações de uso

- Há situações em que compute_seconds é 0, mas charged_seconds é maior que 0 ou vice-versa. Esse é um comportamento normal resultante da forma como os dados são registrados na visualização do sistema. Para uma representação mais precisa dos detalhes de uso da tecnologia sem servidor, recomendamos agregar os dados.

Exemplo

Para obter a cobrança total de horas de RPU usadas em um intervalo de tempo consultando charged_seconds, execute a seguinte consulta:

```
select trunc(start_time) "Day",
(sum(charged_seconds)/3600::double precision) * <Price for 1 RPU> as cost_incurred
from sys_serverless_usage
group by 1
```

```
order by 1
```

Observe que pode haver tempo ocioso durante o intervalo. O tempo ocioso não aumenta o consumo de RPUs.

SYS_SESSION_HISTORY

Use SYS_SESSION_HISTORY para visualizar informações sobre as sessões ativas atuais e o histórico da sessão.

SYS_SESSION_HISTORY está visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	char(50)	O identificador do usuário que gerou a entrada.
session_id	integer	O ID da sessão associada à instrução.
database_name	char(50)	O nome do banco de dados.
status	char	O status da sessão Os valores possíveis são `active`, `timed out` e `closed`.
session_timeout	integer	O tempo máximo em segundos em que uma sessão permanece inativa ou ociosa antes do tempo limite. 0 indica que nenhum tempo limite está definido.
start_time	timestamp	O carimbo de data/hora em que a conexão foi estabelecida.
end_time	timestamp	O carimbo de data/hora em que a conexão foi interrompida.

Exemplo

A seguir é apresentado um exemplo de saída de SYS_SESSION_HISTORY.

```
select * from sys_session_history;
 user_id | session_id |   database_name  | status | session_timeout |
 start_time         |          end_time
---------+------------+------------------+--------+-----------------
+----------------------------+----------------------------
       1 | 1073971370 | dev              | closed |        0        | 2023-07-17
 15:50:12.030104 | 2023-07-17 15:50:12.123218
       1 | 1073979694 | dev              | closed |        0        | 2023-07-17
 15:50:24.117947 | 2023-07-17 15:50:24.131859
       1 | 1073873049 | dev              | closed |        0        | 2023-07-17
 15:49:29.067398 | 2023-07-17 15:49:29.070294
       1 | 1073873086 | database18127a4a | closed |        0        | 2023-07-17
 15:49:29.119018 | 2023-07-17 15:49:29.125925
       1 | 1073832112 | dev              | closed |        0        | 2023-07-17
 15:49:29.164688 | 2023-07-17 15:49:29.179631
       1 | 1073987697 | dev              | closed |        0        | 2023-07-17
 15:49:29.26749  | 2023-07-17 15:49:29.273034
       1 | 1073922429 | dev              | closed |        0        | 2023-07-17
 15:49:33.35315  | 2023-07-17 15:49:33.367499
       1 | 1073766783 | dev              | closed |        0        | 2023-07-17
 15:49:45.38237  | 2023-07-17 15:49:45.396902
       1 | 1073807506 | dev              | active |        0        | 2023-07-17
 15:51:48        |
```

SYS_SPATIAL_SIMPLIFY

É possível consultar a visualização do sistema SYS_SPATIAL_SIMPLIFY para obter informações sobre objetos de geometria espacial simplificada usando o comando COPY. Quando você usa COPY em um shapefile, você pode especificar as opções de ingestão SIMPLIFY `tolerance`, SIMPLIFY AUTO e SIMPLIFY AUTO `max_tolerance`. O resultado da simplificação é resumido na visualização de sistema SYS_SPATIAL_SIMPLIFY.

Quando SIMPLIFY AUTO `max_tolerance` estiver definido, essa visualização contém uma linha para cada geometria que excedeu o tamanho máximo. Quando SIMPLIFY `tolerance` estiver definido, então uma linha para toda a operação COPY é armazenada. Esta linha faz referência ao ID da consulta COPY e à tolerância de simplificação especificada.

Para obter mais informações sobre o carregamento de um shapefile, consulte Carregar um shapefile no Amazon Redshift.

SYS_SPATIAL_SIMPLIFY é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
query_id	bigint	O ID da consulta (comando COPY) que gerou essa linha.
line_number	bigint	Quando COPY `SIMPLIFY AUTO`for especificado, esse valor será o número de registro do registro simplificado no shapefile.
maximum_tolerance	double precision	O valor da tolerância de distância especificado no comando COPY. Este é o valor máximo de tolerância usando a opção `SIMPLIFY AUTO` ou o valor de tolerância fixa usando a opção `SIMPLIFY`.
initial_size	bigint	O tamanho em bytes do valor de dados `GEOMETRY` antes da simplificação.
simplified	char(1)	Quando a opção COPY `SIMPLIFY AUTO` é especificada, `t` se a geometria foi simplificada com sucesso, ou `f` caso contrário. A geometria pode não ser simplificada com sucesso se após a simplificação com a tolerância máxima dada seu tamanho ainda for maior do que o tamanho máximo da geometria.
final_size	bigint	Quando a opção COPY`SIMPLIFY AUTO` for especificada, este é o tamanho em bytes da geometria após a simplificação.
final_tolerance	double precision	Tolerância final escolhida para a simplificação.

Consulta de exemplo

A consulta a seguir retorna a lista de registros que COPY simplificou.

```
SELECT * FROM sys_spatial_simplify;


 query_id | line_number | maximum_tolerance | initial_size | simplified | final_size |
  final_tolerance
----------+-------------+-------------------+--------------+------------+------------
+----------------------
    20    |     1184704 |                -1 |      1513736 | t          |    1008808 |
  1.276386653895e-05
    20    |     1664115 |                -1 |      1233456 | t          |    1023584 |
 6.11707814796635e-06
```

SYS_STREAM_SCAN_ERRORS

Registra erros de registros carregados por meio da ingestão por streaming.

SYS_STREAM_SCAN_ERRORS permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
external_schema_name	character (128)	O nome do fluxo do Kinesis ou do esquema do tópico do Amazon MSK. Há diferenciação entre letras maiúsculas e minúsculas.
stream_name	character (255)	O nome do fluxo ou do tópico. Há diferenciação entre letras maiúsculas e minúsculas.
mv_name	character (128)	O nome da visão materializada associada. Vazio, se não houver. Há diferenciação entre letras maiúsculas e minúsculas.

Nome da coluna	Tipo de dados	Descrição
transaction_id	bigint	O ID da transação.
query_id	bigint	O ID da consulta.
stream_timestamp_type	character (1)	O tipo de carimbo de data/hora da transmissão. Há diferenciação entre letras maiúsculas e minúsculas.
stream_timestamp	timestamp sem fuso horário	A hora em que o registro chegou.
record_time	timestamp sem fuso horário	A hora em que a mensagem de erro foi registrada.
partition_id	character (128)	O ID da partição/fragmento. Há diferenciação entre letras maiúsculas e minúsculas.
position	character (128)	A posição do registro. Isso corresponde ao número de sequência no Kinesis ou ao deslocamento no Amazon MSK. Há diferenciação entre letras maiúsculas e minúsculas.
error_code	integer	O código do erro.
error_reason	character (128)	O motivo do erro. Há diferenciação entre letras maiúsculas e minúsculas.

SYS_STREAM_SCAN_STATES

Registra os estados de varredura dos registros carregados por meio da ingestão por streaming.

SYS_STREAM_SCAN_STATES permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
external_schema_name	character (128)	O nome do esquema externo. Há diferenciação entre letras maiúsculas e minúsculas.
stream_name	character (255)	Nome do streaming. Há diferenciação entre letras maiúsculas e minúsculas.
mv_name	character (128)	O nome da visão materializada associada. Vazio, se não houver. Há diferenciação entre letras maiúsculas e minúsculas.
transaction_id	bigint	O ID da transação.
query_id	bigint	O ID da consulta.
record_time	timestamp sem fuso horário	A hora em que os dados foram registrados.
partition_id	character (128)	O ID da partição ou do fragmento. Há diferenciação entre letras maiúsculas e minúsculas.

Nome da coluna	Tipo de dados	Descrição
latest_position	character (128)	A posição do último registro lido no lote. Isso corresponde ao número de sequência no Kinesis ou ao deslocamento no Amazon MSK. Há diferenciação entre letras maiúsculas e minúsculas.
scanned_rows	bigint	O número de registros que foram verificados no lote.
skipped_rows	bigint	O número de registros que foram ignorados no lote.
scanned_bytes	bigint	O número de bytes que foram verificados no lote.
stream_record_time_min	timestamp sem fuso horário	Hora de chegada do primeiro registro do lote no Kinesis ou Amazon MSK.
stream_record_time_max	timestamp sem fuso horário	Hora de chegada do último registro do lote no Kinesis ou Amazon MSK.

A consulta a seguir mostra dados de fluxo e tópico para consultas específicas.

```
select
 query_id,mv_name::varchar,external_schema_name::varchar,stream_name::varchar,sum(scanned_rows)
 total_records,
sum(scanned_bytes) total_bytes from sys_stream_scan_states where query in
 (5401180,8601939) group by 1,2,3,4;

  query_id  |    mv_name     | external_schema_name |   stream_name   | total_records |
  total_bytes
```

```
------------+----------------+----------------------+-----------------+--------------
+----------------
 5401180    | kinesistest    | kinesis              | kinesisstream   |    1493255696 |
 3209006490704
 8601939    | msktest        | msk                  | mskstream       |      14677023 |
   31056580668
(2 rows)
```

SYS_TRANSACTION_HISTORY

Use SYS_TRANSACTION_HISTORY para ver detalhes de uma transação ao rastrear uma consulta.

SYS_TRANSACTION_HISTORY só permanece visível para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	integer	O ID do usuário que gerou a entrada.
transaction_id	bigint	O ID da transação.
isolation_level	texto	O nível de isolamento da transação. Os possíveis valores são `Serializable` e `Snapshot Isolation` .
status	texto	O status da transação. Os status possíveis são `committed` e `rolledback` .
transaction_start_time	timestamp	A hora de início da transação.
commit_start_time	timestamp	A hora de início da confirmação.

Nome da coluna	Tipo de dados	Descrição
commit_end_time	timestamp	A hora de término da confirmação.
blocks_committed	bigint	O número de blocos que precisavam ser gravados como parte dessa confirmaç ão.
undo_transaction_id	bigint	O ID da transação de desfazer, caso alguma transação tenha sido desfeita. Do contrário, esse valor será -1.

Consultas de exemplo

```
select * from sys_transaction_history order by transaction_start_time desc;

 user_id | transaction_id | isolation_level |   status   |   transaction_start_time
   |     commit_start_time      |      commit_end_time       | blocks_committed |
 undo_transaction_id
---------+----------------+-----------------+------------+--------------------------
+---------------------------+----------------------------+------------------
+---------------------
     100 |           1310 | Serializable    | committed  | 2023-08-27 21:03:11.822205 |
 2023-08-28 21:03:11.825287 | 2023-08-28 21:03:11.854883 |               17 |
       -1
     101 |           1345 | Serializable    | committed  | 2023-08-27 21:03:12.000278 |
 2023-08-28 21:03:12.003736 | 2023-08-28 21:03:12.030061 |               17 |
       -1
     102 |           1367 | Serializable    | committed  | 2023-08-27 21:03:12.1532   |
 2023-08-28 21:03:12.156124 | 2023-08-28 21:03:12.186468 |               17 |
       -1
     100 |           1370 | Serializable    | committed  | 2023-08-27 21:03:12.199316 |
 2023-08-28 21:03:12.204854 | 2023-08-28 21:03:12.238186 |               24 |
       -1
```

```
    100 |           1408 | Serializable    | committed  | 2023-08-27 21:03:53.891107 |
2023-08-28 21:03:53.894825 | 2023-08-28 21:03:53.927465 |               17 |
       -1
    100 |           1409 | Serializable    | rolledback | 2023-08-27 21:03:53.936431 |
2000-01-01 00:00:00        | 2023-08-28 21:04:08.712532 |                0 |
     1409
    101 |           1415 | Serializable    | committed  | 2023-08-27 21:04:24.283188 |
2023-08-28 21:04:24.289196 | 2023-08-28 21:04:24.374318 |               25 |
       -1
    101 |           1416 | Serializable    | committed  | 2023-08-27 21:04:24.38818  |
2023-08-28 21:04:24.391688 | 2023-08-28 21:04:24.415135 |               17 |
       -1
    100 |           1417 | Serializable    | rolledback | 2023-08-27 21:04:24.424252 |
2000-01-01 00:00:00        | 2023-08-28 21:04:28.354826 |                0 |
     1417
    101 |           1418 | Serializable    | rolledback | 2023-08-27 21:04:24.425195 |
2000-01-01 00:00:00        | 2023-08-28 21:04:28.680355 |                0 |
     1418
    100 |           1420 | Serializable    | committed  | 2023-08-27 21:04:28.697607 |
2023-08-28 21:04:28.702374 | 2023-08-28 21:04:28.735541 |               23 |
       -1
    101 |           1421 | Serializable    | committed  | 2023-08-27 21:04:28.744854 |
2023-08-28 21:04:28.749344 | 2023-08-28 21:04:28.779029 |               23 |
       -1
    101 |           1423 | Serializable    | committed  | 2023-08-27 21:04:28.78942  |
2023-08-28 21:04:28.791556 | 2023-08-28 21:04:28.817485 |               16 |
       -1
    100 |           1430 | Serializable    | committed  | 2023-08-27 21:04:28.917788 |
2023-08-28 21:04:28.919993 | 2023-08-28 21:04:28.944812 |               16 |
       -1
    102 |           1494 | Serializable    | committed  | 2023-08-27 21:04:37.029058 |
2023-08-28 21:04:37.033137 | 2023-08-28 21:04:37.062001 |               16 |
       -1
```

SYS_UDF_LOG

Registra mensagens de erro e aviso definidas pelo sistema que são geradas durante a execução da função definida pelo usuário (UDF).

SYS_UDF_LOG está visível somente para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
query_id	bigint	O identificador da consulta.
function_name	texto	O nome da função definida pelo usuário.
record_time	timestamp	A hora em que o log foi criado.
sequence	integer	A sequência de uma única mensagem de log.
message	texto	O texto da mensagem de log.
query_uuid	character(36)	O identificador globalmente exclusivo (UUID) da consulta.

Consultas de exemplo

O exemplo a seguir mostra como as UDFs tratam os erros definidos pelo sistema. O primeiro bloco mostra a definição de uma função de UDF que retorna o inverso de um argumento. Quando você executa a função e fornece 0 como argumento, a função retorna um erro. A última instrução retorna a mensagem de erro registrada em SYS_UDF_LOG.

```
-- Create a function to find the inverse of a number.
CREATE OR REPLACE FUNCTION f_udf_inv(a int)

RETURNS float

IMMUTABLE AS $$return 1/a

$$ LANGUAGE plpythonu;

-- Run the function with 0 to create an error.
Select f_udf_inv(0);

-- Query SYS_UDF_LOG to view the message.
Select query_id, record_time, message::varchar from sys_udf_log;
```

```
query_id    |    record_time              |                message
----------+----------------------------
+-----------------------------------------------------
2211        | 2023-08-23 15:53:11.360538 |  ZeroDivisionError: integer division or
 modulo by zero line 2, in f_udf_inv\n return 1/a\n
```

O exemplo a seguir adiciona o registro em log e uma mensagem de aviso à UDF para que operações de divisão por zero gerem uma mensagem de aviso em vez de interromper o processamento com uma mensagem de erro.

```
-- Create a function to find the inverse of a number and log a warning if you input 0.
CREATE OR REPLACE FUNCTION f_udf_inv_log(a int)
  RETURNS float IMMUTABLE
 AS $$
  import logging
  logger = logging.getLogger() #get root logger
  if a==0:
    logger.warning('You attempted to divide by zero.\nReturning zero instead of error.
\n')
    return 0
  else:
     return 1/a
$$ LANGUAGE plpythonu;

-- Run the function with 0 to trigger the warning.
Select f_udf_inv_log(0);

-- Query SYS_UDF_LOG to view the message.
Select query_id, record_time, message::varchar from sys_udf_log;

 query_id |        record_time         |                                    message
----------+----------------------------
+-------------------------------------------------------------------------------
     0   | 2023-08-23 16:10:48.833503 | WARNING: You attempted to divide by zero.
\nReturning zero instead of error.\n
```

SYS_UNLOAD_DETAIL

Use SYS_UNLOAD_DETAIL para visualizar detalhes de uma operação UNLOAD. Ela registra uma linha para cada arquivo criado por uma instrução UNLOAD. Por exemplo, se uma UNLOAD criar 12 arquivos, a SYS_UNLOAD_DETAIL conterá 12 linhas correspondentes.

SYS_UNLOAD_DETAIL é visível a todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	integer	O identificador do usuário que gerou a entrada.
query_id	integer	O identificador de consulta do comando UNLOAD.
session_id	integer	O ID do processo associado à instrução da consulta.
transaction_id	bigint	O ID da transação associada à instrução da consulta.
file_name	character (1280)	O caminho completo do objeto Amazon S3 para o arquivo.
start_time	timestamp	O horário em que a transação começou.
end_time	timestamp	O horário em que a transação foi concluída.
line_count	bigint	O número de linhas descarregadas no arquivo.
transfer_size	bigint	O número de bytes transferidos.
file_format	character (10)	O formato dos arquivos descarregados.

Consultas de exemplo

A consulta a seguir mostra os detalhes da consulta descarregada, incluindo formato, linhas e contagem de arquivos do comando de descarga.

```
select query_id, substring(file_name, 0, 50), transfer_size, file_format from
 sys_unload_detail;
```

Exemplo de resultado.

```
 query_id |                    substring                                | transfer_size
 | file_format
----------+-----------------------------------------------------------
+---------------+-------------
     9272 | s3://amzn-s3-demo-bucket/my_unload_doc_venue0000_part_00.gz  |
 395886 | Text
     9272 | s3://amzn-s3-demo-bucket/my_unload_doc_venue0001_part_00.gz  |
 406444 | Text
     9272 | s3://amzn-s3-demo-bucket/my_unload_doc_venue0002_part_00.gz  |
 409431 | Text
     9272 | s3://amzn-s3-demo-bucket/my_unload_doc_venue0003_part_00.gz  |
 403051 | Text
     9272 | s3://amzn-s3-demo-bucket/my_unload_doc_venue0004_part_00.gz  |
 413592 | Text
     9272 | s3://amzn-s3-demo-bucket/my_unload_doc_venue0005_part_00.gz  |
 395689 | Text
(6 rows)
```

SYS_UNLOAD_HISTORY

Use SYS_LOAD_HISTORY para visualizar detalhes dos comandos UNLOAD. Cada linha representa um comando UNLOAD com estatísticas acumuladas para alguns dos campos. Contém tanto comandos UNLOAD em execução como finalizados.

SYS_UNLOAD_HISTORY é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	integer	O identificador do usuário que enviou o descarregamento.
query_id	bigint	O identificador de consulta do comando UNLOAD.
transaction_id	bigint	O identificador da transação.
session_id	integer	O identificador do processo que está executando o descarregamento.
database_name	texto	O nome do banco de dados ao qual o usuário estava conectado quando a operação foi enviada.
status	texto	O status do comando UNLOAD. Entre os valores válidos estão: `running`, `completed` , `aborted` e `unknown`.
start_time	timestamp	O horário em que o descarregamento começou.
end_time	timestamp	O horário em que o descarregamento foi concluído.
duration	bigint	A quantidade de tempo (em microssegundos) transcorrida no comando UNLOAD.
file_format	texto	O formato dos arquivos de saída.

Nome da coluna	Tipo de dados	Descrição
compression_type	texto	O tipo de compactação.
unloaded_location	texto	O local do Amazon S3 dos arquivos descarregados.
unloaded_rows	bigint	O número de linhas.
unloaded_files_count	bigint	A contagem de arquivos dos arquivos de saída.
unloaded_files_size	bigint	O tamanho dos arquivos de saída.
error_message	texto	A mensagem de erro do comando UNLOAD.

Consultas de exemplo

A consulta a seguir mostra os detalhes da consulta descarregada, incluindo formato, linhas e contagem de arquivos do comando de descarga.

```
SELECT query_id,
       file_format,
       start_time,
       duration,
       unloaded_rows,
       unloaded_files_count
FROM sys_unload_history
ORDER BY query_id,
file_format limit 100;
```

Exemplo de resultado.

```
 query_id | file_format |         start_time         | duration | unloaded_rows |
 unloaded_files_count
----------+-------------+----------------------------+----------+---------------
+----------------------
```

```
  527067 | Text        | 2022-02-09 05:18:35.844452 |  5932478 |           10 |
           1
```

SYS_USERLOG

Registra os detalhes das seguintes alterações de um usuário de banco de dados:

- Criar usuário
- Descartar usuário
- Alterar usuário (renomear)
- Alterar usuário (alterar as propriedades)

É possível consultar essa visualização para ver informações sobre grupos de trabalho sem servidor e clusters provisionados.

SYS_USERLOG só permanece visível para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	integer	O identificador do usuário que enviou o descarregamento.
user_name	character(50)	O nome de usuário afetado pelas alterações.
original_user_name	character(50)	O nome de usuário original em uma ação de renomear. Esse campo fica vazio para todas as outras ações.
ação	character(10)	A ação ocorrida. Os valores válidos são alter, create, drop e rename.

Nome da coluna	Tipo de dados	Descrição
has_create_db_privs	integer	Se for verdadeiro (um valor de 1), o usuário terá permissões para criar um banco de dados.
is_superuser	integer	Se verdadeiro (um valor de 1), o usuário poderá atualizar catálogos do sistema.
has_update_catalog_privs	integer	Se verdadeiro (um valor de 1), o usuário poderá atualizar catálogos do sistema.
password_expiration	timestamp	A data de validade da senha.
session_id	integer	O ID do processo.
transaction_id	bigint	O ID da transação.
record_time	timestamp	O horário (em UTC) de início da consulta.

Consultas de exemplo

O exemplo a seguir executa quatro ações do usuário e, depois, consulta a visualização SYS_USERLOG.

```
CREATE USER userlog1 password 'Userlog1';
ALTER USER userlog1 createdb createuser;
ALTER USER userlog1 rename  to userlog2;
DROP user userlog2;

SELECT user_id, user_name, original_user_name, action, has_create_db_privs,
 is_superuser from SYS_USERLOG order by record_time desc;
```

```
user_id |  user_name | original_user_name |  action | has_create_db_privs |
 is_superuser
```

```
--------+-----------+--------------------+---------+---------------------+-----------
    108 |  userlog2  |                    | drop    |                   1 |   1
    108 |  userlog2  |     userlog1       | rename  |                   1 |   1
    108 |  userlog1  |                    | alter   |                   1 |   1
    108 |  userlog1  |                    | create  |                   0 |   0
 (4 rows)
```

SYS_VACUUM_HISTORY

Use SYS_VACUUM_HISTORY para visualizar detalhes das consultas de vacuum. Para obter informações sobre o comando VACUUM, consulte VACUUM.

SYS_VACUUM_HISTORY permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
user_id	integer	O ID do usuário que iniciou a consulta.
transaction_id	long	O ID da transação para a instrução VACUUM.
query_id	long	O identificador de consulta para a instrução VACUUM. É possível associar essa tabela à visualização SYS_QUERY _DETAIL para ver as instruçõe s SQL individuais que são executadas para determina da transação VACUUM. Se você limpar o banco de dados inteiro, cada tabela é limpada em uma transação separada. Para operações VACUUM

Nome da coluna	Tipo de dados	Descrição
		automatizadas, esse valor é nulo.
database_name	texto	O nome do banco de dados.
schema_name	texto	O nome do esquema.
table_name	texto	O nome da tabela.
table_id	integer	O ID da tabela.
vacuum_type	caractere	O tipo de operação VACUUM. Os valores possíveis são: • **Delete** • **Sort** • **Reindex** • **Recluster** • **Full** Para obter mais informaçõ es sobre os tipos de vacuum, consulte VACUUM.
is_automatic	booleano	`true` se a operação for um vacuum automático. Caso contrário,, `false`.

Nome da coluna	Tipo de dados	Descrição
status	caractere	A descrição da atividade que está sendo processada no momento como parte da operação de limpeza: • Inicializar • Sort • Merge • Excluir • Select • Failed • Completo • Ignorado • Criar ordem de INTERLEAVED SORTKEY
start_time	timestamp	A hora em que a operação de vacuum começou.
end_time	timestamp	A hora em que a operação de vacuum terminou. Se a operação estiver em andamento, esse campo estará em branco.
record_time	timestamp	A hora em que a operação de vacuum foi registrada em SYS_VACUUM_HISTORY.

Nome da coluna	Tipo de dados	Descrição
duration	integer	O número de microsseg undos entre o início e o fim da operação de vacuum. Se a operação de vacuum estiver em andamento, esse campo estará em branco.
rows_before_vacuum	bigint	O número real de linhas na tabela e mais todas as linhas excluídas que ainda estão armazenadas em disco (esperando para serem limpadas).
size_before_vacuum	integer	O tamanho da tabela antes do início da operação de vacuum, em MB.
reclaimable_rows	bigint	O número de linhas que a operação de vácuo estima que recuperará antes de iniciar.
reclaimed_rows	bigint	O número de linhas recuperad as pela operação de vacuum.
reclaimed_blocks	bigint	O número de blocos recuperados pela operação de vacuum.
sortedrows_before_vacuum	integer	O número de linhas classific adas na tabela antes do início da operação de vacuum.

Nome da coluna	Tipo de dados	Descrição
sortedrows_after_vacuum	integer	O número adicional de linhas classificadas na tabela após a conclusão da operação de vacuum. Isso não inclui as linhas contadas em `sortedrows_before_vacuum` .

Mapeamento de visualizações do sistema para migrar para visualizações de monitoramento de SYS

Quando você migra o cluster provisionado do Amazon Redshift para o Amazon Redshift sem servidor, as consultas de monitoramento ou diagnóstico podem se referir a visualizações do sistema que só estão disponíveis em clusters provisionados. É possível atualizar as consultas para usar as visualizações de monitoramento de SYS. Esta página fornece mapeamentos de visualizações somente provisionados para SYS para você consultar ao atualizar consultas.

Tópicos

- SYS_QUERY_HISTORY
- SYS_QUERY_DETAIL
- SYS_RESTORE_LOG
- SYS_RESTORE_STATE
- SYS_TRANSACTION_HISTORY
- SYS_QUERY_TEXT
- SYS_CONNECTION_LOG
- SYS_SESSION_HISTORY
- SYS_LOAD_DETAIL
- SYS_LOAD_HISTORY
- SYS_LOAD_ERROR_DETAIL
- SYS_UNLOAD_HISTORY
- SYS_UNLOAD_DETAIL

- SYS_COPY_REPLACEMENTS
- SYS_DATASHARE_USAGE_CONSUMER
- SYS_DATASHARE_USAGE_PRODUCER
- SYS_DATASHARE_CROSS_REGION_USAGE
- SYS_DATASHARE_CHANGE_LOG
- SYS_EXTERNAL_QUERY_DETAIL
- SYS_EXTERNAL_QUERY_ERROR
- SYS_VACUUM_HISTORY
- SYS_ANALYZE_HISTORY
- SYS_ANALYZE_COMPRESSION_HISTORY
- SYS_MV_REFRESH_HISTORY
- SYS_MV_STATE
- SYS_PROCEDURE_CALL
- SYS_PROCEDURE_MESSAGES
- SYS_UDF_LOG
- SYS_USERLOG
- SYS_SCHEMA_QUOTA_VIOLATIONS
- SYS_SPATIAL_SIMPLIFY

SYS_QUERY_HISTORY

Algumas ou todas as colunas das tabelas a seguir também são definidas em SYS_QUERY_HISTORY.

- STL_DDLTEXT
- STL_ERROR
- STL_QUERY
- STL_UTILITYTEXT
- STL_WLM_QUERY
- STV_INFLIGHT

- STV_RECENTS
- STV_WLM_QUERY_STATE
- SVL_COMPILE
- SVL_MULTI_STATEMENT_VIOLATIONS
- SVL_QLOG
- SVL_QUERY_QUEUE_INFO
- SVL_STATEMENTTEXT
- SVL_TERMINATE

SYS_QUERY_DETAIL

Algumas ou todas as colunas das tabelas a seguir também são definidas em SYS_QUERY_DETAIL.

- STL_AGGR
- STL_ALERT_EVENT_LOG
- STL_BCAST
- STL_DELETE
- STL_DIST
- STL_EXPLAIN
- STL_HASH
- STL_HASHJOIN
- STL_INSERT
- STL_LIMIT
- STL_MERGE
- STL_MERGEJOIN
- STL_NESTLOOP
- STL_PARSE
- STL_PLAN_INFO
- STL_PROJECT
- STL_QUERY_METRICS

- STL_RETURN
- STL_SAVE
- STL_SCAN
- STL_SORT
- STL_STREAM_SEGS
- STL_UNIQUE
- STL_WINDOW
- STV_EXEC_STATE
- STV_QUERY_METRICS
- SVCS_QUERY_SUMMARY
- SVL_QUERY_METRICS
- SVL_QUERY_METRICS_SUMMARY
- SVL_QUERY_REPORT
- SVL_QUERY_SUMMARY
- SVV_QUERY_STATE

SYS_RESTORE_LOG

Algumas ou todas as colunas da tabela a seguir também são definidas em SYS_RESTORE_LOG.

- SVL_RESTORE_ALTER_TABLE_PROGRESS

SYS_RESTORE_STATE

Algumas ou todas as colunas da tabela a seguir também são definidas em SYS_RESTORE_STATE.

- STV_XRESTORE_ALTER_QUEUE_STATE

SYS_TRANSACTION_HISTORY

Algumas ou todas as colunas das tabelas a seguir também são definidas em SYS_TRANSACTION_HISTORY.

- STL_COMMIT_STATS
- STL_TR_CONFLICT
- STL_UNDONE

SYS_QUERY_TEXT

Algumas ou todas as colunas da tabela a seguir também são definidas em SYS_QUERY_TEXT.

- STL_QUERYTEXT

SYS_CONNECTION_LOG

Algumas ou todas as colunas da tabela a seguir também são definidas em SYS_CONNECTION_LOG.

- STL_CONNECTION_LOG

SYS_SESSION_HISTORY

Algumas ou todas as colunas das tabelas a seguir também são definidas em SYS_SESSION_HISTORY.

- STL_SESSIONS
- STL_RESTARTED_SESSIONS
- STV_SESSIONS

SYS_LOAD_DETAIL

Algumas ou todas as colunas da tabela a seguir também são definidas em SYS_LOAD_DETAIL.

- STL_LOAD_COMMITS

SYS_LOAD_HISTORY

Algumas ou todas as colunas da tabela a seguir também são definidas em SYS_LOAD_HISTORY.

- STL_LOAD_COMMITS

SYS_LOAD_ERROR_DETAIL

Algumas ou todas as colunas das tabelas a seguir também são definidas em SYS_LOAD_ERROR_DETAIL.

- STL_LOADERROR_DETAIL
- STL_LOAD_ERRORS

SYS_UNLOAD_HISTORY

Algumas ou todas as colunas da tabela a seguir também são definidas em SYS_UNLOAD_HISTORY.

- STL_UNLOAD_LOG

SYS_UNLOAD_DETAIL

Algumas ou todas as colunas da tabela a seguir também são definidas em SYS_UNLOAD_DETAIL.

- STL_UNLOAD_LOG

SYS_COPY_REPLACEMENTS

Algumas ou todas as colunas da tabela a seguir também são definidas em SYS_COPY_REPLACEMENTS.

- STL_REPLACEMENTS

SYS_DATASHARE_USAGE_CONSUMER

Algumas ou todas as colunas da tabela a seguir também são definidas em SYS_DATASHARE_USAGE_CONSUMER.

- SVL_DATASHARE_USAGE_CONSUMER

SYS_DATASHARE_USAGE_PRODUCER

Algumas ou todas as colunas da tabela a seguir também são definidas em SYS_DATASHARE_USAGE_PRODUCER.

- SVL_DATASHARE_USAGE_PRODUCER

SYS_DATASHARE_CROSS_REGION_USAGE

Algumas ou todas as colunas da tabela a seguir também são definidas em SYS_DATASHARE_CROSS_REGION_USAGE.

- SVL_DATASHARE_CROSS_REGION_USAGE

SYS_DATASHARE_CHANGE_LOG

Algumas ou todas as colunas da tabela a seguir também são definidas em SYS_DATASHARE_CHANGE_LOG.

- SVL_DATASHARE_CHANGE_LOG

SYS_EXTERNAL_QUERY_DETAIL

Algumas ou todas as colunas das tabelas a seguir também são definidas em SYS_EXTERNAL_QUERY_DETAIL.

- SVL_FEDERATED_QUERY
- SVL_S3LIST
- SVL_S3QUERY
- SVL_S3QUERY_SUMMARY

SYS_EXTERNAL_QUERY_ERROR

Algumas ou todas as colunas das tabelas a seguir também são definidas em SYS_EXTERNAL_QUERY_ERROR.

- SVL_SPECTRUM_SCAN_ERROR

SYS_VACUUM_HISTORY

Algumas ou todas as colunas das tabelas a seguir também são definidas em SYS_VACUUM_HISTORY.

- STL_VACUUM
- SVL_VACUUM_PERCENTAGE
- SVV_VACUUM_PROGRESS
- SVV_VACUUM_SUMMARY

SYS_ANALYZE_HISTORY

Algumas ou todas as colunas das tabelas a seguir também são definidas em SYS_ANALYZE_HISTORY.

- STL_ANALYZE

SYS_ANALYZE_COMPRESSION_HISTORY

Algumas ou todas as colunas das tabelas a seguir também são definidas em SYS_ANALYZE_COMPRESSION_HISTORY.

- STL_ANALYZE_COMPRESSION

SYS_MV_REFRESH_HISTORY

Algumas ou todas as colunas das tabelas a seguir também são definidas em SYS_MV_REFRESH_HISTORY.

- SVL_MV_REFRESH_STATUS

SYS_MV_STATE

Algumas ou todas as colunas das tabelas a seguir também são definidas em SYS_MV_STATE.

- STL_MV_STATE

SYS_PROCEDURE_CALL

Algumas ou todas as colunas das tabelas a seguir também são definidas em SYS_PROCEDURE_CALL.

- SVL_STORED_PROC_CALL

SYS_PROCEDURE_MESSAGES

Algumas ou todas as colunas das tabelas a seguir também são definidas em SYS_PROCEDURE_MESSAGES.

- SVL_STORED_PROC_MESSAGES

SYS_UDF_LOG

Algumas ou todas as colunas das tabelas a seguir também são definidas em SYS_UDF_LOG.

- SVL_UDF_LOG

SYS_USERLOG

Algumas ou todas as colunas das tabelas a seguir também são definidas em SYS_USERLOG.

- STL_USERLOG

SYS_SCHEMA_QUOTA_VIOLATIONS

Algumas ou todas as colunas das tabelas a seguir também são definidas em SYS_SCHEMA_QUOTA_VIOLATIONS.

- STL_SCHEMA_QUOTA_VIOLATIONS

SYS_SPATIAL_SIMPLIFY

Algumas ou todas as colunas das tabelas a seguir também são definidas em SYS_SPATIAL_SIMPLIFY.

- SVL_SPATIAL_SIMPLIFY

Monitoramento do sistema (somente provisionado)

As tabelas e visualizações do sistema a seguir podem ser consultadas em clusters provisionados. As visualizações e tabelas STL e STV contêm um subconjunto de dados encontrados em várias tabelas do sistema. Elas fornecem um acesso mais rápido e mais fácil aos dados que são consultados com frequência nessas tabelas.

As visões SVCS fornecem detalhes sobre consultas nos clusters principal e de escalabilidade da simultaneidade. As visualizações SVL fornecem informações somente para consultas executadas no cluster principal, com exceção de SVL_STATEMENTTEXT. O SVL_STATEMENTTEXT pode conter informações para consultas executadas em clusters de escalabilidade simultânea, bem como no cluster principal.

Tópicos

- Visualizações STL de registro em log
- Tabelas STV para dados de snapshot
- Visualizações SVCS para o cluster principal e clusters de escalabilidade de simultaneidade
- Visualizações SVL para o cluster principal

Visualizações STL de registro em log

As visualizações do sistema STL são geradas a partir dos arquivos de log do Amazon Redshift para fornecer um histórico do sistema.

Esses arquivos residem em cada nó no cluster do data warehouse. As visualizações STL pegam as informações dos logs e as formatam em visualizações utilizáveis para administradores de sistema.

Retenção de log: as visualizações do sistema STL retêm sete dias do histórico de log. A retenção de log é garantida para todos os tamanhos e tipos de nós de cluster e não é afetada por mudanças na workload do cluster. A retenção de log também não é afetada pelo status do cluster, como quando o cluster está pausado. Você tem menos de sete dias de histórico de log somente no caso em que o cluster é novo. Não é necessário realizar nenhuma ação para reter os logs, mas será preciso copiar periodicamente os dados de log para outras tabelas ou descarregá-los no Amazon S3 para reter aqueles com mais de sete dias de existência.

Tópicos

- STL_AGGR
- STL_ALERT_EVENT_LOG
- STL_ANALYZE
- STL_ANALYZE_COMPRESSION
- STL_BCAST
- STL_COMMIT_STATS
- STL_CONNECTION_LOG
- STL_DDLTEXT
- STL_DELETE
- STL_DISK_FULL_DIAG
- STL_DIST
- STL_ERROR
- STL_EXPLAIN
- STL_FILE_SCAN
- STL_HASH
- STL_HASHJOIN
- STL_INSERT
- STL_LIMIT
- STL_LOAD_COMMITS
- STL_LOAD_ERRORS
- STL_LOADERROR_DETAIL
- STL_MERGE
- STL_MERGEJOIN
- STL_MV_STATE
- STL_NESTLOOP
- STL_PARSE
- STL_PLAN_INFO
- STL_PROJECT
- STL_QUERY

- STL_QUERY_METRICS
- STL_QUERYTEXT
- STL_REPLACEMENTS
- STL_RESTARTED_SESSIONS
- STL_RETURN
- STL_S3CLIENT
- STL_S3CLIENT_ERROR
- STL_SAVE
- STL_SCAN
- STL_SCHEMA_QUOTA_VIOLATIONS
- STL_SESSIONS
- STL_SORT
- STL_SSHCLIENT_ERROR
- STL_STREAM_SEGS
- STL_TR_CONFLICT
- STL_UNDONE
- STL_UNIQUE
- STL_UNLOAD_LOG
- STL_USAGE_CONTROL
- STL_USERLOG
- STL_UTILITYTEXT
- STL_VACUUM
- STL_WINDOW
- STL_WLM_ERROR
- STL_WLM_RULE_ACTION
- STL_WLM_QUERY

STL_AGGR

Analisa as etapas de execução agregadas para consultas. Essas etapas ocorrem durante a execução de funções agregadas e de cláusulas GROUP BY.

STL_AGGR permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_AGGR contém apenas consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
slice	integer	O número que identifica a fatia em que a consulta estava sendo executada.
segment	integer	O número que identifica o segmento da consulta.
etapa	integer	Etapa da consulta que foi executada.
starttime	timestamp	O horário (em UTC) de início da consulta. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Horário em UTC em que a consulta foi finalizada. O tempo total inclui consultas e execução, com seis dígitos de precisão

Nome da coluna	Tipo de dados	Descrição
		para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
tasknum	integer	Número do processo de tarefa de consulta que foi atribuído para executar a etapa.
rows	bigint	O número total de linhas que foram processadas.
bytes	bigint	O tamanho, em bytes, de todas as linhas de saída da etapa.
slots	integer	O número de buckets de hash.
occupied	integer	O número de slots que contêm registros.
maxlength	integer	O tamanho do maior slot.
tbl	integer	ID da tabela.
is_diskbased	character(1)	Se o valor é true (t), a consulta foi executada como uma operação em disco. Se o valor é false (f), a consulta foi executada na memória.
workmem	bigint	O número de bytes da memória de trabalho atribuída à etapa.
type	character(6)	O tipo da etapa. Os valores válidos são: • HASHED. Indica que a etapa usou uma agregação não classificada e agrupada. • PLAIN. Indica que a etapa usou uma agregação escalar e não agrupada. • SORTED. Indica que a etapa usou uma agregação classificada e agrupada.
resizes	integer	Essas informações são somente para uso interno.
flushable	integer	Essas informações são somente para uso interno.

Consultas de exemplo

Retorna informações sobre as etapas de execução agregadas para SLICE 1 e TBL 239.

```
select query, segment, bytes, slots, occupied, maxlength, is_diskbased, workmem, type
from stl_aggr where slice=1 and tbl=239
order by rows
limit 10;
```

```
 query | segment | bytes |  slots  | occupied | maxlength | is_diskbased |  workmem  |
 type
-------+---------+-------+---------+----------+-----------+--------------+-----------
+--------
   562 |       1 |     0 | 4194304 |        0 |         0 | f            | 383385600 |
 HASHED
   616 |       1 |     0 | 4194304 |        0 |         0 | f            | 383385600 |
 HASHED
   546 |       1 |     0 | 4194304 |        0 |         0 | f            | 383385600 |
 HASHED
   547 |       0 |     8 |       0 |        0 |         0 | f            |         0 |
 PLAIN
   685 |       1 |    32 | 4194304 |        1 |         0 | f            | 383385600 |
 HASHED
   652 |       0 |     8 |       0 |        0 |         0 | f            |         0 |
 PLAIN
   680 |       0 |     8 |       0 |        0 |         0 | f            |         0 |
 PLAIN
   658 |       0 |     8 |       0 |        0 |         0 | f            |         0 |
 PLAIN
   686 |       0 |     8 |       0 |        0 |         0 | f            |         0 |
 PLAIN
   695 |       1 |    32 | 4194304 |        1 |         0 | f            | 383385600 |
 HASHED
(10 rows)
```

STL_ALERT_EVENT_LOG

Registra um alerta quando o otimizador de consultas identifica as condições que podem indicar problemas de performance. Use a visualização STL_ALERT_EVENT_LOG para identificar oportunidades para melhorar a performance da consulta.

Uma consulta consiste em vários segmentos e cada segmento consiste em uma ou mais etapas. Para obter mais informações, consulte Processamento de consulta.

STL_ALERT_EVENT_LOG é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_ALERT_EVENT_LOG contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
slice	integer	O número que identifica a fatia em que a consulta estava sendo executada.
segment	integer	O número que identifica o segmento da consulta.
etapa	integer	Etapa da consulta que foi executada.
pid	integer	O ID do processo associado à instrução e à fatia. A mesma consulta poderá ter vários PIDs, se for executada em várias fatias.
xid	bigint	O ID da transação associada à instrução.

Nome da coluna	Tipo de dados	Descrição
event	character (1024)	A descrição do evento de alerta.
solução	character (1024)	A solução recomendada.
event_time	timestamp	O horário (em UTC) de início da consulta. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .

Observações de uso

Você pode usar o STL_ALERT_EVENT_LOG para identificar problemas potenciais nas consultas, depois siga as práticas recomendadas em Ajustar o desempenho da consulta para otimizar o design do banco de dados e reescrever as consultas. A tabela STL_ALERT_EVENT_LOG registra os seguintes alertas:

- Ausência de estatísticas

  Não há estatísticas. Execute o ANALYZE depois de carregar dados ou de atualizações significativas e use o STATUPDATE com as operações COPY. Para obter mais informações, consulte Práticas recomendadas do Amazon Redshift para criar consultas.
- Loop aninhado

  Um loop aninhado é geralmente um produto cartesiano. Avalie sua consulta para garantir que todas as tabelas participantes sejam unidas de forma eficiente.
- Filtro muito seletivo

  A taxa de linhas retornadas em relação às linhas pesquisadas na varredura é menor do que 0.05. Linhas verificadas refere-se ao valor de `rows_pre_user_filter` e linhas retornadas refere-se ao valor das linhas na visualização do sistema STL_SCAN. Indica que a consulta está fazendo a varredura em um número excepcionalmente grande de linhas para determinar o conjunto de resultados. Isso pode ser causado pela falta ou por erros de chaves de classificação. Para obter mais informações, consulte Chaves de classificação.
- Excesso de linhas fantasma

Uma varredura ignorou um número relativamente grande de linhas que estão marcadas como excluídas, mas não como limpadas, ou de linhas que foram inseridas, mas não confirmadas. Para obter mais informações, consulte Vacuum de tabelas.

- Grande distribuição

  Mais de 1.000.000 de linhas foram redistribuídas para uma junção hash ou agregação. Para obter mais informações, consulte Distribuição de dados para otimização de consultas.

- Grande transmissão

  Mais de 1.000.000 de linhas foram transmitidas para uma junção hash. Para obter mais informações, consulte Distribuição de dados para otimização de consultas.

- Execução em série

  Um estilo de redistribuição DS_DIST_ALL_INNER foi indicado no plano de consulta, o que força uma execução em série, pois toda a tabela interna foi redistribuída para um único nó. Para obter mais informações, consulte Distribuição de dados para otimização de consultas.

Consultas de exemplo

As consultas a seguir mostram eventos de alerta para quatro consultas.

```
SELECT query, substring(event,0,25) as event,
substring(solution,0,25) as solution,
trim(event_time) as event_time from stl_alert_event_log order by query;

 query |             event             |          solution            |     event_time

-------+-------------------------------+------------------------------
+---------------------
  6567 | Missing query planner statist | Run the ANALYZE command      | 2014-01-03
 18:20:58
  7450 | Scanned a large number of del | Run the VACUUM command to rec| 2014-01-03
 21:19:31
  8406 | Nested Loop Join in the query | Review the join predicates to| 2014-01-04
 00:34:22
 29512 | Very selective query filter:r | Review the choice of sort key| 2014-01-06
 22:00:00

(4 rows)
```

STL_ANALYZE

Registra os detalhes das operações ANALYZE.

STL_ANALYZE só permanece visível para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_ANALYZE_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
xid	long	O ID da transação.
banco de dados	char(30)	O nome do banco de dados.
table_id	integer	O ID da tabela.
status	char(15)	O resultado do comando de análise. Os valores possíveis são `Full`, `Skipped` e `PredicateColumn` .
rows	double	O número total de linhas na tabela.
modified_ rows	double	O número total de linhas que foram modificadas desde a última operação ANALYZE.
threshold _percent	integer	O valor do parâmetro `analyze_threshold_percent` .
is_auto	char(1)	O valor será true (`t`) se a operação incluir uma operação de análise do Amazon Redshift por padrão. O valor será falso (`f`) se o comando ANALYZE foi executado explicitamente.

Nome da coluna	Tipo de dados	Descrição
starttime	timestamp	O horário (em UTC) de início da execução da operação análise.
endtime	timestamp	O horário (em UTC) de término da execução da operação de análise.
prevtime	timestamp	O horário (em UTC) em que a tabela foi analisada anteriorm ente.
num_predi cate_cols	integer	O número atual de colunas de predicado na tabela.
num_new_p redicate_ cols	integer	O número de novas colunas de predicado na tabela, desde a operação de análise anterior.
is_backgr ound	character(1)	O valor será verdadeiro (`t`) se a análise foi executada por uma operação de análises automáticas. Caso contrário, o valor será falso (`f`).
auto_anal yze_phase	character(100)	Reservado para uso interno.
schema_na me	char(128)	O nome do esquema para a tabela.
table_name	char(136)	O nome da tabela.

Consultas de exemplo

O exemplo a seguir une a STV_TBL_PERM para mostrar o nome da tabela e os detalhes de execução.

```
select distinct a.xid, trim(t.name) as name, a.status, a.rows, a.modified_rows,
 a.starttime, a.endtime
from stl_analyze a
```

```
join stv_tbl_perm t  on t.id=a.table_id
where name = 'users'
order by starttime;

xid    | name  | status          | rows  | modified_rows | starttime           |
 endtime
-------+-------+-----------------+-------+---------------+---------------------
+---------------------
  1582 | users | Full            | 49990 |         49990 | 2016-09-22 22:02:23 |
 2016-09-22 22:02:28
244287 | users | Full            | 24992 |         74988 | 2016-10-04 22:50:58 |
 2016-10-04 22:51:01
244712 | users | Full            | 49984 |         24992 | 2016-10-04 22:56:07 |
 2016-10-04 22:56:07
245071 | users | Skipped         | 49984 |             0 | 2016-10-04 22:58:17 |
 2016-10-04 22:58:17
245439 | users | Skipped         | 49984 |          1982 | 2016-10-04 23:00:13 |
 2016-10-04 23:00:13
(5 rows)
```

STL_ANALYZE_COMPRESSION

Registra os detalhes das operações de análise de compactação durante os comandos COPY ou ANALYZE COMPRESSION.

STL_ANALYZE_COMPRESSION permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_ANALYZE_COMPRESSION_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.

Nome da coluna	Tipo de dados	Descrição
start_time	timestamp	O horário em que a operação de análise de compactação foi iniciada.
xid	bigint	O ID da transação da operação de análise de compactação.
tbl	integer	O ID da tabela da tabela que foi analisada.
tablename	character(128)	O nome da tabela da tabela que foi analisada.
col	integer	O índice da coluna na tabela que foi analisada para determinar a codificação de compactação.
old_encoding	character(15)	O tipo de codificação antes da análise de compactação.
new_encoding	character(15)	O tipo de codificação após a análise de compactação.

Nome da coluna	Tipo de dados	Descrição
mode	character(14)	Os valores possíveis são: PRESET Especifica que `new_encoding` é determinado pelo comando COPY do Amazon Redshift com base no tipo de dados da coluna. Não é criada nenhuma amostra de dados. ON Especifica que `new_encoding` é determinado pelo comando COPY do Amazon Redshift com base em uma análise de dados de amostra. ANALYZE ONLY Especifica que `new_encoding` é determinado pelo comando ANALYZE COMPRESSION do Amazon Redshift com base em uma análise de dados de amostra. No entanto, o tipo de codificação da coluna analisada não é alterado.
best_compression_encoding	character(15)	O tipo de codificação que oferece a melhor taxa de compactação.
recommended_bytes	character(15)	Os bytes usados ao adotar a nova codificação.
best_compression_bytes	character(15)	Os bytes usados ao adotar a melhor codificação de compactação.
ndv	bigint	O número de valores distintos nas linhas com amostras.

Consultas de exemplo

O exemplo a seguir inspeciona os detalhes da análise de compactação na tabela `lineitem` pelo último comando COPY executado na mesma sessão.

```
select xid, tbl, btrim(tablename) as tablename, col, old_encoding, new_encoding,
 best_compression_encoding, mode
from stl_analyze_compression
where xid = (select xid from stl_query where query = pg_last_copy_id()) order by col;

 xid  |  tbl   | tablename | col |  old_encoding   |  new_encoding   |
 best_compression_encoding |      mode
------+--------+-----------+-----+-----------------+-----------------
+---------------------------+----------------
 5308 | 158961 | $lineitem |   0 | mostly32        | az64            | delta
          | ON
 5308 | 158961 | $lineitem |   1 | mostly32        | az64            | az64
          | ON
 5308 | 158961 | $lineitem |   2 | lzo             | az64            | az64
          | ON
 5308 | 158961 | $lineitem |   3 | delta           | az64            | az64
          | ON
 5308 | 158961 | $lineitem |   4 | bytedict        | az64            | bytedict
          | ON
 5308 | 158961 | $lineitem |   5 | mostly32        | az64            | az64
          | ON
 5308 | 158961 | $lineitem |   6 | delta           | az64            | az64
          | ON
 5308 | 158961 | $lineitem |   7 | delta           | az64            | az64
          | ON
 5308 | 158961 | $lineitem |   8 | lzo             | lzo             | lzo
          | ON
 5308 | 158961 | $lineitem |   9 | runlength       | runlength       | runlength
          | ON
 5308 | 158961 | $lineitem |  10 | delta           | az64            | az64
          | ON
 5308 | 158961 | $lineitem |  11 | delta           | az64            | az64
          | ON
 5308 | 158961 | $lineitem |  12 | delta           | az64            | az64
          | ON
 5308 | 158961 | $lineitem |  13 | bytedict        | bytedict        | bytedict
          | ON
 5308 | 158961 | $lineitem |  14 | bytedict        | bytedict        | bytedict
          | ON
```

```
 5308 | 158961 | $lineitem |  15 | text255         | text255         | text255
          | ON
(16 rows)
```

STL_BCAST

Registra informações sobre a atividade de rede durante a execução das etapas de uma consulta que transmite dados. O tráfego de rede é capturado pelos números de linhas, bytes e pacotes que são enviados pela rede durante uma determinada etapa em uma determinada fatia. A duração da etapa é a diferença entre os horários de início e término registrados.

Para identificar as etapas de transmissão em uma consulta, procure pelos rótulos bcast na exibição SVL_QUERY_SUMMARY ou execute o comando EXPLAIN e, em seguida, procure pelos atributos da etapa que incluem o bcast.

STL_BCAST permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_BCAST contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.

Nome da coluna	Tipo de dados	Descrição
slice	integer	O número que identifica a fatia em que a consulta estava sendo executada.
segment	integer	O número que identifica o segmento da consulta.
etapa	integer	Etapa da consulta que foi executada.
starttime	timestamp	O horário (em UTC) de início da consulta. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Horário em UTC em que a consulta foi finalizada. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
tasknum	integer	Número do processo de tarefa de consulta que foi atribuído para executar a etapa.
rows	bigint	O número total de linhas que foram processadas.
bytes	bigint	O tamanho, em bytes, de todas as linhas de saída da etapa.
packets	integer	O número total de pacotes enviados pela rede.

Consultas de exemplo

O exemplo a seguir retorna as informações transmitidas pelas consultas com um ou mais pacotes e onde a diferença entre o início e o término da consulta seja de um segundo ou mais.

```
select query, slice, step, rows, bytes, packets, datediff(seconds, starttime, endtime)
from stl_bcast
where packets>0 and datediff(seconds, starttime, endtime)>0;
```

```
 query | slice | step | rows | bytes | packets | date_diff
```

```
-------+-------+------+------+-------+---------+-----------
   453 |     2 |    5 |    1 |   264 |       1 |         1
   798 |     2 |    5 |    1 |   264 |       1 |         1
  1408 |     2 |    5 |    1 |   264 |       1 |         1
  2993 |     0 |    5 |    1 |   264 |       1 |         1
  5045 |     3 |    5 |    1 |   264 |       1 |         1
  8073 |     3 |    5 |    1 |   264 |       1 |         1
  8163 |     3 |    5 |    1 |   264 |       1 |         1
  9212 |     1 |    5 |    1 |   264 |       1 |         1
  9873 |     1 |    5 |    1 |   264 |       1 |         1
(9 rows)
```

STL_COMMIT_STATS

Fornece métricas relativas à performance da operação de confirmação, incluindo informações sobre o tempo dos vários estágios de confirmação e o número de blocos confirmados. Consulte a STL_COMMIT_STATS para determinar que parte de uma transação foi gasta na operação de confirmação e o quanto as filas estão sendo utilizadas.

STL_COMMIT_STATS é visível somente para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_TRANSACTION_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
xid	bigint	O identificador da transação que está sendo confirmada.
node	integer	Número do nó. -1 é o nó líder.
startqueue	timestamp	O início da colocação na fila de confirmação.
startwork	timestamp	O início da confirmação.
endflush	timestamp	O término da fase de liberação de blocos não gravados.

Nome da coluna	Tipo de dados	Descrição
endstage	timestamp	O término da fase de preparação dos metadados.
endlocal	timestamp	O término da fase de confirmação local.
startglobal	timestamp	O início da fase global.
endtime	timestamp	O término da confirmação.
queuelen	bigint	O número de transações que foram processadas antes desta transação na fila de confirmação.
permblocks	bigint	O número de blocos permanentes existentes no momento desta confirmação.
newblocks	bigint	O número de novos blocos permanentes existentes no momento desta confirmação.
dirtyblocks	bigint	O número de blocos que precisavam ser gravados como parte desta confirmação.
headers	bigint	O número de cabeçalhos de blocos que precisavam ser gravados como parte desta confirmação.
numxids	integer	O número de transações de DML ativas.
oldestxid	bigint	O XID da transação ativa de DML mais antiga.
extwritelatency	bigint	Essas informações são somente para uso interno.
metadatawritten	int	Essas informações são somente para uso interno.
tombstonedblocks	bigint	Essas informações são somente para uso interno.

Nome da coluna	Tipo de dados	Descrição
tossedblocks	bigint	Essas informações são somente para uso interno.
batched_by	bigint	Essas informações são somente para uso interno.

Consulta de exemplo

```
select node, datediff(ms,startqueue,startwork) as queue_time,
datediff(ms, startwork, endtime) as commit_time, queuelen
from stl_commit_stats
where xid = 2574
order by node;

node | queue_time   | commit_time | queuelen
-----+--------------+-------------+---------
  -1 |            0 |         617 |        0
   0 | 444950725641 |         616 |        0
   1 | 444950725636 |         616 |        0
```

STL_CONNECTION_LOG

Registra em log as tentativas de autenticação e as conexões e desconexões.

STL_CONNECTION_LOG só permanece visível para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_CONNECTION_LOG. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
event	character(50)	O evento de conexão ou de autenticação.

Nome da coluna	Tipo de dados	Descrição
recordtime	timestamp	O horário em que o evento ocorreu.
remotehost	character(45)	O nome ou endereço IP do host remoto.
remoteport	character(32)	O número da porta do host remoto.
pid	integer	O ID do processo associado à instrução.
dbname	character(50)	Database name.
username	character(50)	User name.
authmethod	character(32)	O método de autenticação.
duration	integer	A duração da conexão em microssegundos.
sslversion	character(50)	A versão do Secure Sockets Layer (SSL). Esta coluna fica vazia quando o SSL está desabilitado.
sslcipher	character(128)	A codificação do SSL.
mtu	integer	A unidade de transmissão máxima (MTU).
sslcompression	character(64)	O tipo de compactação do SSL.
sslexpansion	character(64)	O tipo de expansão do SSL.
iamauthguid	character(36)	O ID de autenticação da IAM para a solicitação de CloudTrail.
application_name	character(250)	A iniciais ou o nome atualizado da aplicação de uma sessão.
os_version	character(64)	A versão do sistema operacional que está na máquina cliente que se conecta ao cluster do Amazon Redshift.

Nome da coluna	Tipo de dados	Descrição
driver_version	character(64)	A versão do driver ODBC ou JDBC que se conecta ao cluster do Amazon Redshift a partir de ferramentas de cliente SQL de terceiros.
plugin_name	character(32)	O nome do plugin usado para se conectar ao seu cluster do Amazon Redshift.
protocol_version	integer	A versão do protocolo interno que o driver do Amazon Redshift usa ao estabelecer sua conexão com o servidor. As versões do protocolo são negociadas entre o driver e o servidor. A versão descreve os recursos disponíveis. Os valores válidos são: • 0 (BASE_SERVER_PROTOCOL_VERSION) • 1 (EXTENDED_RESULT_METADATA_SERVER_PROTOCOL_VERSION): para salvar uma viagem de ida e volta por consulta, o servidor envia informações extras de metadados do conjunto de resultados. • 2 (BINARY_PROTOCOL_VERSION): dependendo do tipo de dados do conjunto de resultados, o servidor envia dados em formato binário. • 3 (EXTENDED2_RESULT_METADATA_SERVER_PROTOCOL_VERSION): o servidor envia informações que diferenciam entre maiúsculas e minúsculas (agrupamento) de uma coluna.
sessionid	character(36)	O identificador exclusivo global da sessão atual. O ID da sessão persiste por meio da reinicialização da falha do nó.
compression	character(16)	O algoritmo de compactação em uso para a conexão.

Consultas de exemplo

Para visualizar os detalhes das conexões abertas, execute a consulta a seguir.

```
select recordtime, username, dbname, remotehost, remoteport
from stl_connection_log
where event = 'initiating session'
and pid not in
(select pid from stl_connection_log
where event = 'disconnecting session')
order by 1 desc;

recordtime          | username    | dbname     | remotehost    | remoteport

--------------------+-------------+------------+---------------+------------
2014-11-06 20:30:06 | rdsdb       | dev        | [local]       |

2014-11-06 20:29:37 | test001     | test       | 10.49.42.138  | 11111

2014-11-05 20:30:29 | rdsdb       | dev        | 10.49.42.138  | 33333

2014-11-05 20:28:35 | rdsdb       | dev        | [local]       |
(4 rows)
```

O exemplo a seguir reflete uma tentativa falha de autenticação e uma operação de conexão e desconexão bem-sucedida.

```
select event, recordtime, remotehost, username
from stl_connection_log order by recordtime;

            event      |         recordtime        |  remotehost   | username

-----------------------+---------------------------+---------------+---------
authentication failure | 2012-10-25 14:41:56.96391 | 10.49.42.138 | john

authenticated          | 2012-10-25 14:42:10.87613 | 10.49.42.138 | john

initiating session     | 2012-10-25 14:42:10.87638 | 10.49.42.138 | john

disconnecting session  | 2012-10-25 14:42:19.95992 | 10.49.42.138 | john

(4 rows)
```

O exemplo a seguir mostra a versão do driver ODBC, o sistema operacional na máquina cliente e o plug-in usado para se conectar ao cluster do Amazon Redshift. Neste exemplo, o plugin usado é para autenticação de driver ODBC padrão usando um nome de login e senha.

```
select driver_version, os_version, plugin_name from stl_connection_log;

driver_version                          |  os_version                       |
 plugin_name
----------------------------------------+-----------------------------------
+--------------------
Amazon Redshift ODBC Driver 1.4.15.0001 | Darwin 18.7.0 x86_64              | none
Amazon Redshift ODBC Driver 1.4.15.0001 | Linux 4.15.0-101-generic x86_64   | none
```

O exemplo a seguir mostra a versão do sistema operacional na máquina cliente, a versão do driver e a versão do protocolo.

```
select os_version, driver_version, protocol_version from stl_connection_log;

os_version                      |  driver_version              | protocol_version
--------------------------------+------------------------------+--------------------
Linux 4.15.0-101-generic x86_64 | Redshift JDBC Driver 2.0.0.0 | 2
Linux 4.15.0-101-generic x86_64 | Redshift JDBC Driver 2.0.0.0 | 2
Linux 4.15.0-101-generic x86_64 | Redshift JDBC Driver 2.0.0.0 | 2
```

STL_DDLTEXT

Captura as seguintes instruções de DDL que foram executadas no sistema.

Essas instruções de DDL incluem as seguintes consultas e objetos:

- CREATE SCHEMA, TABLE, VIEW
- DROP SCHEMA, TABLE, VIEW
- ALTER SCHEMA, TABLE

Consulte também STL_QUERYTEXT, STL_UTILITYTEXT e SVL_STATEMENTTEXT. Essas exibições fornecem uma linha do tempo dos comandos de SQL que são executados no sistema; esse histórico é útil para fins de solução de problemas e para criar uma trilha de auditoria de todas as atividades do sistema.

Use as colunas STARTTIME e ENDTIME para saber quais instruções foram registradas em um determinado período. Os blocos longos de texto SQL são quebrados em linhas de 200 caracteres; a coluna SEQUENCE identifica os fragmentos de texto que pertencem a uma única instrução.

STL_DDLTEXT permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_QUERY_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
xid	bigint	O ID da transação associada à instrução.
pid	integer	O ID do processo associado à instrução.
rótulo	character(320)	O nome do arquivo usado para executar a consulta ou um rótulo definido com o comando SET QUERY_GROUP. Se a consulta não for baseada em arquivos ou o parâmetro QUERY_GROUP não estiver definido, o valor deste campo será branco.
starttime	timestamp	O horário (em UTC) de início da consulta. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Horário em UTC em que a consulta foi finalizada. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .

Nome da coluna	Tipo de dados	Descrição
sequence	integer	Quando uma única instrução contém mais de 200 caracteres, são registradas linhas adicionais para essa instrução. O valor 0 da sequência é a primeira linha, 1 é a segunda, e assim por diante.
texto	character(200)	O texto em SQL, em incrementos de 200 caracteres. Esse campo pode conter caracteres especiais como barra invertida (\\) e nova linha (\n).

Consultas de exemplo

A consulta a seguir retorna registros que incluem instruções de DDL executadas anteriormente.

```
select xid, starttime, sequence, substring(text,1,40) as text
from stl_ddltext order by xid desc, sequence;
```

Esta é uma saída de exemplo que mostra quatro instruções CREATE TABLE. As instruções de DDL aparecem na coluna `text`, que é truncada para legibilidade.

```
 xid  |         starttime          | sequence |                  text
------+----------------------------+----------
+----------------------------------------
 1806 | 2013-10-23 00:11:14.709851 |        0 | CREATE TABLE supplier ( s_suppkey int4
 N
 1806 | 2013-10-23 00:11:14.709851 |        1 |  s_comment varchar(101) NOT NULL )
 1805 | 2013-10-23 00:11:14.496153 |        0 | CREATE TABLE region ( r_regionkey int4
 N
 1804 | 2013-10-23 00:11:14.285986 |        0 | CREATE TABLE partsupp ( ps_partkey int8
 1803 | 2013-10-23 00:11:14.056901 |        0 | CREATE TABLE part ( p_partkey int8 NOT
 N
 1803 | 2013-10-23 00:11:14.056901 |        1 | ner char(10) NOT NULL , p_retailprice
 nu
(6 rows)
```

Reconstrução de SQL armazenado

O SQL a seguir lista as linhas armazenadas na coluna `text` de STL_DDLTEXT. As linhas são ordenadas por `xid` e `sequence`. Se o SQL original for mais extenso que várias linhas de 200 caracteres, STL_DDLTEXT poderá conter várias linhas por `sequence`.

```
SELECT xid, sequence, LISTAGG(CASE WHEN LEN(RTRIM(text)) = 0 THEN text ELSE RTRIM(text)
 END, '') WITHIN GROUP (ORDER BY sequence) as query_statement
FROM stl_ddltext GROUP BY xid, sequence ORDER BY xid, sequence;
```

```
xid     |  sequence | query_statement
--------+-----------+-----------------
7886671    0          create external schema schema_spectrum_uddh\nfrom data catalog
\ndatabase 'spectrum_db_uddh'\niam_role ''\ncreate external database if not exists;
7886752    0          CREATE EXTERNAL TABLE schema_spectrum_uddh.soccer_league\n(\n
  league_rank smallint,\n  prev_rank   smallint,\n  club_name   varchar(15),\n
 league_name varchar(20),\n  league_off  decimal(6,2),\n  le
7886752    1          ague_def  decimal(6,2),\n  league_spi  decimal(6,2),\n
 league_nspi smallint\n)\nROW FORMAT DELIMITED \n    FIELDS TERMINATED BY ',' \n
 LINES TERMINATED BY '\\n\\l'\nstored as textfile\nLOCATION 's
7886752    2          3://mybucket-spectrum-uddh/'\ntable properties
 ('skip.header.line.count'='1');
...
```

Para reconstruir o SQL armazenado na coluna `text` de STL_DDLTEXT, execute a instrução SQL a seguir. Ela reúne instruções de DDL de um ou mais segmentos na coluna `text`. Antes de executar o SQL reconstruído, substitua os caracteres especiais (\n) por uma nova linha no cliente SQL. Os resultados a seguir da instrução SELECT reúnem três linhas em sequência para reconstruir o SQL, no campo `query_statement`.

```
SELECT LISTAGG(CASE WHEN LEN(RTRIM(text)) = 0 THEN text ELSE RTRIM(text) END) WITHIN
 GROUP (ORDER BY sequence) as query_statement
FROM stl_ddltext GROUP BY xid, endtime order by xid, endtime;
```

```
query_statement
--------------
create external schema schema_spectrum_uddh\nfrom data catalog\ndatabase
 'spectrum_db_uddh'\niam_role ''\ncreate external database if not exists;
```

```
CREATE EXTERNAL TABLE schema_spectrum_uddh.soccer_league\n(\n  league_rank smallint,
\n  prev_rank   smallint,\n  club_name   varchar(15),\n  league_name varchar(20),\n
 league_off  decimal(6,2),\n  league_def  decimal(6,2),\n  league_spi  decimal(6,2),
\n  league_nspi smallint\n)\nROW FORMAT DELIMITED \n    FIELDS TERMINATED BY ',' \n
  LINES TERMINATED BY '\\n\\l'\nstored as textfile\nLOCATION 's3://mybucket-spectrum-
uddh/'\ntable properties ('skip.header.line.count'='1');
```

STL_DELETE

Analisa as etapas de execução de exclusões das consultas.

STL_DELETE permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_DELETE contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
slice	integer	O número que identifica a fatia em que a consulta estava sendo executada.
segment	integer	O número que identifica o segmento da consulta.

Nome da coluna	Tipo de dados	Descrição
etapa	integer	Etapa da consulta que foi executada.
starttime	timestamp	O horário (em UTC) de início da consulta. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Horário em UTC em que a consulta foi finalizada. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
tasknum	integer	Número do processo de tarefa de consulta que foi atribuído para executar a etapa.
rows	bigint	O número total de linhas que foram processadas.
tbl	integer	ID da tabela.

Consultas de exemplo

Para criar uma linha na STL_DELETE, o exemplo a seguir insere uma linha na tabela EVENT e, em seguida, exclui a linha.

Primeiro, insira uma linha na tabela EVENT e verifique se ela foi inserida.

```
insert into event(eventid,venueid,catid,dateid,eventname)
values ((select max(eventid)+1 from event),95,9,1857,'Lollapalooza');
```

```
select * from event
where eventname='Lollapalooza'
order by eventid;
```

```
 eventid | venueid | catid | dateid |  eventname   |      starttime
---------+---------+-------+--------+--------------+---------------------
```

```
    4274 |     102 |     9 |   1965 | Lollapalooza | 2008-05-01 19:00:00
    4684 |     114 |     9 |   2105 | Lollapalooza | 2008-10-06 14:00:00
    5673 |     128 |     9 |   1973 | Lollapalooza | 2008-05-01 15:00:00
    5740 |      51 |     9 |   1933 | Lollapalooza | 2008-04-17 15:00:00
    5856 |     119 |     9 |   1831 | Lollapalooza | 2008-01-05 14:00:00
    6040 |     126 |     9 |   2145 | Lollapalooza | 2008-11-15 15:00:00
    7972 |      92 |     9 |   2026 | Lollapalooza | 2008-07-19 19:30:00
    8046 |      65 |     9 |   1840 | Lollapalooza | 2008-01-14 15:00:00
    8518 |      48 |     9 |   1904 | Lollapalooza | 2008-03-19 15:00:00
    8799 |      95 |     9 |   1857 | Lollapalooza |
(10 rows)
```

Agora, exclua a linha que você adicionou na tabela EVENT e verifique se ela foi excluída.

```
delete from event
where eventname='Lollapalooza' and eventid=(select max(eventid) from event);
```

```
select * from event
where eventname='Lollapalooza'
order by eventid;
```

```
 eventid | venueid | catid | dateid |  eventname   |      starttime
---------+---------+-------+--------+--------------+---------------------
    4274 |     102 |     9 |   1965 | Lollapalooza | 2008-05-01 19:00:00
    4684 |     114 |     9 |   2105 | Lollapalooza | 2008-10-06 14:00:00
    5673 |     128 |     9 |   1973 | Lollapalooza | 2008-05-01 15:00:00
    5740 |      51 |     9 |   1933 | Lollapalooza | 2008-04-17 15:00:00
    5856 |     119 |     9 |   1831 | Lollapalooza | 2008-01-05 14:00:00
    6040 |     126 |     9 |   2145 | Lollapalooza | 2008-11-15 15:00:00
    7972 |      92 |     9 |   2026 | Lollapalooza | 2008-07-19 19:30:00
    8046 |      65 |     9 |   1840 | Lollapalooza | 2008-01-14 15:00:00
    8518 |      48 |     9 |   1904 | Lollapalooza | 2008-03-19 15:00:00
(9 rows)
```

Em seguida, execute uma consulta na stl_delete para visualizar as etapas da exclusão. Neste exemplo, a consulta retornou mais de 300 linhas e, portanto, a saída abaixo foi reduzida para possibilitar a exibição.

```
select query, slice, segment, step, tasknum, rows, tbl from stl_delete order by query;
```

```
 query | slice | segment | step | tasknum | rows |  tbl
-------+-------+---------+------+---------+------+--------
     7 |     0 |       0 |    1 |       0 |    0 | 100000
     7 |     1 |       0 |    1 |       0 |    0 | 100000
     8 |     0 |       0 |    1 |       2 |    0 | 100001
     8 |     1 |       0 |    1 |       2 |    0 | 100001
     9 |     0 |       0 |    1 |       4 |    0 | 100002
     9 |     1 |       0 |    1 |       4 |    0 | 100002
    10 |     0 |       0 |    1 |       6 |    0 | 100003
    10 |     1 |       0 |    1 |       6 |    0 | 100003
    11 |     0 |       0 |    1 |       8 |    0 | 100253
    11 |     1 |       0 |    1 |       8 |    0 | 100253
    12 |     0 |       0 |    1 |       0 |    0 | 100255
    12 |     1 |       0 |    1 |       0 |    0 | 100255
    13 |     0 |       0 |    1 |       2 |    0 | 100257
    13 |     1 |       0 |    1 |       2 |    0 | 100257
    14 |     0 |       0 |    1 |       4 |    0 | 100259
    14 |     1 |       0 |    1 |       4 |    0 | 100259
    ...
```

STL_DISK_FULL_DIAG

Registra informações sobre erros registrados quando o disco está cheio.

STL_DISK_FULL_DIAG só permanece visível para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
currenttime	bigint	O dia e a hora em que o erro foi gerado em microssegundos desde 1° de janeiro de 2000.
node_num	bigint	O identificador do nó.
query_id	bigint	O identificador da consulta que causou o erro.

Nome da coluna	Tipo de dados	Descrição		
temp_blocks	bigint	O número de blocos temporários criado pela consulta.		

Consultas de exemplo

O exemplo a seguir retorna detalhes sobre os dados armazenados quando há um erro de disco cheio.

```
select * from stl_disk_full_diag
```

O exemplo a seguir converte `currenttime` em um carimbo de data e hora.

```
select '2000-01-01'::timestamp + (currenttime/1000000.0)* interval '1 second' as
 currenttime,node_num,query_id,temp_blocks from pg_catalog.stl_disk_full_diag;
```

```
        currenttime         | node_num | query_id | temp_blocks
----------------------------+----------+----------+-------------
 2019-05-18 19:19:18.609338 |        0 |   569399 |       70982
 2019-05-18 19:37:44.755548 |        0 |   569580 |       70982
 2019-05-20 13:37:20.566916 |        0 |   597424 |       70869
```

STL_DIST

Registra informações sobre a atividade de rede durante a execução das etapas de uma consulta que distribui dados. O tráfego de rede é capturado pelos números de linhas, bytes e pacotes que são enviados pela rede durante uma determinada etapa em uma determinada fatia. A duração da etapa é a diferença entre os horários de início e término registrados.

Para identificar as etapas de distribuição em uma consulta, procure pelos rótulos dist na exibição QUERY_SUMMARY ou execute o comando EXPLAIN e, em seguida, procure pelos atributos da etapa que incluem o dist.

STL_DIST permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_DIST contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
slice	integer	O número que identifica a fatia em que a consulta estava sendo executada.
segment	integer	O número que identifica o segmento da consulta.
etapa	integer	Etapa da consulta que foi executada.
starttime	timestamp	O horário (em UTC) de início da consulta. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Horário em UTC em que a consulta foi finalizada. O tempo total inclui consultas e execução, com seis dígitos

Nome da coluna	Tipo de dados	Descrição
		de precisão para segundos fracionários. Por exemplo: **2009-06-12 11:29:19.131358** .
tasknum	integer	Número do processo de tarefa de consulta que foi atribuído para executar a etapa.
rows	bigint	O número total de linhas que foram processadas.
bytes	bigint	O tamanho, em bytes, de todas as linhas de saída da etapa.
packets	integer	O número total de pacotes enviados pela rede.

Consultas de exemplo

O exemplo a seguir retorna informações de distribuição para consultas com um ou mais pacotes e duração maior do que zero.

```
select query, slice, step, rows, bytes, packets,
datediff(seconds, starttime, endtime) as duration
from stl_dist
where packets>0 and datediff(seconds, starttime, endtime)>0
order by query
limit 10;
```

```
 query  | slice | step |  rows  |  bytes  | packets | duration
--------+-------+------+--------+---------+---------+----------
    567 |     1 |    4 |  49990 | 6249564 |     707 |        1
    630 |     0 |    5 |   8798 |  408404 |      46 |        2
    645 |     1 |    4 |   8798 |  408404 |      46 |        1
    651 |     1 |    5 | 192497 | 9226320 |    1039 |        6
    669 |     1 |    4 | 192497 | 9226320 |    1039 |        4
    675 |     1 |    5 |   3766 |  194656 |      22 |        1
    696 |     0 |    4 |   3766 |  194656 |      22 |        1
    705 |     0 |    4 |    930 |   44400 |       5 |        1
 111525 |     0 |    3 |     68 |   17408 |       2 |        1
(9 rows)
```

STL_ERROR

Registra erros de processamento interno gerados pelo mecanismo de banco de dados do Amazon Redshift. A tabela STL_ERROR não registra os erros ou mensagens de SQL. As informações na STL_ERROR são úteis para solucionar determinados erros. Um engenheiro de suporte da AWS pode solicitar que você forneça essas informações como parte do processo de solução de problemas.

STL_ERROR permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_QUERY_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Para uma lista de códigos de erros que podem ser gerados enquanto os dados são carregados no comando Copiar, consulte Referência de erros de carregamento.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
process	character(12)	O processo que gerou a exceção.
recordtime	timestamp	O horário em que o erro ocorreu.
pid	integer	ID do processo. A tabela STL_QUERY contém os IDs do processo e os IDs exclusivos das consultas concluídas.
errcode	integer	O código de erro correspondente à categoria do erro.
arquivo	character(90)	O nome do arquivo de origem onde o erro ocorreu.

Nome da coluna	Tipo de dados	Descrição
linenum	integer	O número da linha no arquivo de origem onde o erro ocorreu.
context	character(100)	A causa do erro.
erro	character(512)	A mensagem de erro.

Consultas de exemplo

O exemplo a seguir recuperar as informações sobre um erro da STL_ERROR.

```
select process, errcode, linenum as line,
trim(error) as err
from stl_error;

   process    | errcode | line |                                 err
--------------+---------+------
+-----------------------------------------------------------------
 padbmaster   |    8001 |  194 | Path prefix: s3://redshift-downloads/testnulls/
venue.txt*
 padbmaster   |    8001 |  529 | Listing bucket=redshift-downloads prefix=tests/
category-csv-quotes
 padbmaster   |       2 |  190 | database "template0" is not currently accepting
 connections
 padbmaster   |      32 | 1956 | pq_flush: could not send data to client: Broken pipe
(4 rows)
```

STL_EXPLAIN

Exibe o plano EXPLAIN para uma consulta que foi enviada para execução.

STL_EXPLAIN permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_EXPLAIN contém apenas as consultas executadas em clusters principais. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_QUERY_EXPLAIN. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
nodeid	integer	O identificador do nó de plano, onde o nó mapeia para uma ou mais etapas na execução da consulta.
parentid	integer	O identificador do nó de plano para um nó pai. Um nó pai tem um certo número de nós filho. Por exemplo, uma junção de mesclagem é o pai das varreduras em tabelas unidas.
plannode	character(400)	O texto do nó de saída de EXPLAIN. Os nós de plano que se referem à execução em nós de computação recebem o prefixo **XN** na saída EXPLAIN.
info	character(400)	As informações sobre o qualificador e o filtro do nó de plano. Por exemplo, as condições de junção e as restrições da cláusula WHERE estão incluídas nesta coluna.
user_query_id	bigint	O identificador da consulta enviada pelo usuário, conforme registrado na coluna query_id de SYS_QUERY_EXPLAIN.

Nome da coluna	Tipo de dados	Descrição
child_query_sequence	integer	O número de sequência da consulta filha regravada na consulta do usuário, começando com 1.
query_uuid	character(36)	O identificador globalmente exclusivo (UUID) da consulta.

Consultas de exemplo

Considere a seguinte saída de EXPLAIN para uma consulta de junção agregada:

```
explain select avg(datediff(day, listtime, saletime)) as avgwait
from sales, listing where sales.listid = listing.listid;
                                QUERY PLAN

------------------------------------------------------------------------------
 XN Aggregate  (cost=6350.30..6350.31 rows=1 width=16)
  ->  XN Hash Join DS_DIST_NONE  (cost=47.08..6340.89 rows=3766 width=16)
        Hash Cond: ("outer".listid = "inner".listid)
        -> XN Seq Scan on listing  (cost=0.00..1924.97 rows=192497 width=12)
        -> XN Hash  (cost=37.66..37.66 rows=3766 width=12)
              -> XN Seq Scan on sales  (cost=0.00..37.66 rows=3766 width=12)
(6 rows)
```

Se você executar essa consulta e o ID de consulta for 10, use a tabela STL_EXPLAIN para visualizar os mesmos tipos de informações que o comando EXPLAIN retorna:

```
select query,nodeid,parentid,substring(plannode from 1 for 30),
substring(info from 1 for 20) from stl_explain
where query=10 order by 1,2;

query| nodeid |parentid|           substring            |    substring
-----+--------+--------+--------------------------------+-------------------
10   |      1 |      0 |XN Aggregate  (cost=6717.61..6  |
10   |      2 |      1 |  -> XN Merge Join DS_DIST_NO   | Merge Cond:("outer"
10   |      3 |      2 |       -> XN Seq Scan on lis    |
10   |      4 |      2 |       -> XN Seq Scan on sal    |
(4 rows)
```

Considere a seguinte consulta:

```
select event.eventid, sum(pricepaid)
from event, sales
where event.eventid=sales.eventid
group by event.eventid order by 2 desc;

eventid |   sum
--------+----------
    289 | 51846.00
   7895 | 51049.00
   1602 | 50301.00
    851 | 49956.00
   7315 | 49823.00
...
```

Se o ID da consulta for 15, a seguinte consulta de visualização do sistema retornará os nós do plano que foram concluídos. Nesse caso, a ordem dos nós é invertida para mostrar a ordem real de execução:

```
select query,nodeid,parentid,substring(plannode from 1 for 56)
from stl_explain where query=15 order by 1, 2 desc;

query|nodeid|parentid|                              substring
-----+------+--------+-----------------------------------------------------------
15   |    8 |      7 |                                    -> XN Seq Scan on eve
15   |    7 |      5 |                              -> XN Hash(cost=87.98..87.9
15   |    6 |      5 |                              -> XN Seq Scan on sales(cos
15   |    5 |      4 |                         -> XN Hash Join DS_DIST_OUTER(cos
15   |    4 |      3 |                   -> XN HashAggregate(cost=862286577.07..
15   |    3 |      2 |          -> XN Sort(cost=1000862287175.47..10008622871
15   |    2 |      1 |  -> XN Network(cost=1000862287175.47..1000862287197.
15   |    1 |      0 |XN Merge(cost=1000862287175.47..1000862287197.46 rows=87
(8 rows)
```

A consulta a seguir recupera os IDs de consulta de todos os planos de consulta que contêm uma função de janela:

```
select query, trim(plannode) from stl_explain
where plannode like '%Window%';

query|                                     btrim
```

```
-----+------------------------------------------------------------------------
26   | -> XN Window(cost=1000985348268.57..1000985351256.98 rows=170 width=33)
27   | -> XN Window(cost=1000985348268.57..1000985351256.98 rows=170 width=33)
(2 rows)
```

STL_FILE_SCAN

Retorna os arquivos que o Amazon Redshift leu ao carregar dados usando o comando COPY.

Consultar esta visualização pode ajudar a solucionar erros de carregamento de dados. STL_FILE_SCAN pode ser particularmente útil para identificar problemas em carregamentos de dados paralelos, porque carregamentos de dados paralelos normalmente carregam muitos arquivos com um único comando COPY.

STL_FILE_SCAN permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_FILE_SCAN contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_LOAD_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.

Nome da coluna	Tipo de dados	Descrição
slice	integer	O número que identifica a fatia em que a consulta estava sendo executada.
name	character(90)	O caminho completo e o nome do arquivo que foi carregado.
lines	bigint	O número de linhas lidas do arquivo.
bytes	bigint	O número de bytes lidos do arquivo.
loadtime	bigint	O tempo gasto no carregamento do arquivo (em microsseg undos).
curtime	Timestamp	Timestamp que representa a hora em que o Amazon Redshift começou a processar o arquivo.
is_parcial	integer	Valor que, se true (1), indica que o arquivo de entrada é dividido em intervalos durante uma operação COPY. Se esse valor for false (0), o arquivo de entrada não será dividido.
start_offset	bigint	Valor que, se o arquivo de entrada for dividido durante uma operação COPY, indica o valor de deslocamento da divisão (em bytes). Se o arquivo não estiver dividido, esse valor será 0.

Consultas de exemplo

A consulta a seguir recupera os nomes e tempos de carregamento de todos os arquivos que o Amazon Redshift levou mais de 1.000.000 de microssegundos para ler.

```
select trim(name)as name, loadtime from stl_file_scan
where loadtime > 1000000;
```

Essa consulta retorna os dados de saída de exemplo a seguir.

```
          name            | loadtime
--------------------------+----------
```

```
 listings_pipe.txt         |  9458354
 allusers_pipe.txt         |  2963761
 allevents_pipe.txt        |  1409135
 tickit/listings_pipe.txt  |  7071087
 tickit/allevents_pipe.txt |  1237364
 tickit/allusers_pipe.txt  |  2535138
 listings_pipe.txt         |  6706370
 allusers_pipe.txt         |  3579461
 allevents_pipe.txt        |  1313195
 tickit/allusers_pipe.txt  |  3236060
 tickit/listings_pipe.txt  |  4980108
(11 rows)
```

STL_HASH

Analisa as etapas de execução de hash para as consultas.

STL_HASH permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_HASH contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.

Nome da coluna	Tipo de dados	Descrição
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
slice	integer	O número que identifica a fatia em que a consulta estava sendo executada.
segment	integer	O número que identifica o segmento da consulta.
etapa	integer	Etapa da consulta que foi executada.
starttime	timestamp	O horário (em UTC) de início da consulta. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Horário em UTC em que a consulta foi finalizada. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
tasknum	integer	Número do processo de tarefa de consulta que foi atribuído para executar a etapa.
rows	bigint	O número total de linhas que foram processadas.
bytes	bigint	O tamanho, em bytes, de todas as linhas de saída da etapa.
slots	integer	O número total de buckets de hash.
occupied	integer	O número total de slots que contêm registros.
maxlength	integer	O tamanho do maior slot.
tbl	integer	ID da tabela.

Nome da coluna	Tipo de dados	Descrição
is_diskbased	character(1)	Se o valor é true (t), a consulta foi realizada como uma operação em disco. Se o valor é false (f), a consulta foi realizada na memória.
workmem	bigint	O número total de bytes da memória de trabalho atribuída à etapa.
num_parts	integer	O número total de partições em que a tabela hash é particionada durante uma etapa de hash.
est_rows	bigint	O número estimado de linhas para o hash.
num_blocks_permitted	integer	Essas informações são somente para uso interno.
resizes	integer	Essas informações são somente para uso interno.
soma de verificação	bigint	Essas informações são somente para uso interno.
runtime_filter_size	integer	O tamanho do filtro do tempo de execução em bytes.
max_runtime_filter_size	integer	O tamanho máximo do filtro do tempo de execução em bytes.

Consultas de exemplo

O exemplo a seguir retorna informações sobre o número de partições que foram usadas em um hash para a consulta 720, e indica que nenhuma das etapas foi executada em disco.

```
select slice, rows, bytes, occupied, workmem, num_parts, est_rows,
 num_blocks_permitted, is_diskbased
from stl_hash
```

```
where query=720 and segment=5
order by slice;
```

```
 slice | rows | bytes  | occupied | workmem  | num_parts | est_rows |
 num_blocks_permitted | is_diskbased
-------+------+--------+----------+----------+-----------+----------
+----------------------+--------------
     0 |  145 | 585800 |        1 | 88866816 |        16 |        1 |
 52              f
     1 |    0 |      0 |        0 |        0 |        16 |        1 |
 52              f
(2 rows)
```

STL_HASHJOIN

Analisa as etapas de execução da junção hash para as consultas.

STL_HASHJOIN permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_HASHJOIN contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.

Nome da coluna	Tipo de dados	Descrição
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
slice	integer	O número que identifica a fatia em que a consulta estava sendo executada.
segment	integer	O número que identifica o segmento da consulta.
etapa	integer	Etapa da consulta que foi executada.
starttime	timestamp	O horário (em UTC) de início da consulta. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Horário em UTC em que a consulta foi finalizada. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
tasknum	integer	Número do processo de tarefa de consulta que foi atribuído para executar a etapa.
rows	bigint	O número total de linhas que foram processadas.
tbl	integer	ID da tabela.
num_parts	integer	O número total de partições em que a tabela hash é particionada durante uma etapa de hash.

Nome da coluna	Tipo de dados	Descrição
join_type	integer	O tipo de junção para a etapa: • 0. A consulta usou uma junção interna. • 1. A consulta usou uma junção externa esquerda. • 2. A consulta usou uma junção externa completa. • 3. A consulta usou uma junção externa direita. • 4. A consulta usou um operador UNION. • 5. A consulta usou uma condição IN. • 6. Essas informações são somente para uso interno. • 7. Essas informações são somente para uso interno. • 8. Essas informações são somente para uso interno. • 9. Essas informações são somente para uso interno. • 10. Essas informações são somente para uso interno. • 11. Essas informações são somente para uso interno. • 12. Essas informações são somente para uso interno.
hash_looped	character(1)	Essas informações são somente para uso interno.
switched_parts	character(1)	Indica se os lados da compilação (ou externo) e da investigação (ou interno) foram trocados.
used_prefetching	character(1)	Essas informações são somente para uso interno.
hash_segment	integer	O segmento de etapa de hash correspondente.
hash_step	integer	O número de etapa da etapa de hash correspondente.
soma de verificação	bigint	Essas informações são somente para uso interno.

Nome da coluna	Tipo de dados	Descrição
distribuição	integer	Essas informações são somente para uso interno.

Consultas de exemplo

O exemplo a seguir retorna o número de partições usadas em uma junção hash para a consulta 720.

```
select query, slice, tbl, num_parts
from stl_hashjoin
where query=720 limit 10;
```

```
 query | slice | tbl | num_parts
-------+-------+-----+-----------
   720 |     0 | 243 |         1
   720 |     1 | 243 |         1
(2 rows)
```

STL_INSERT

Analisa as etapas de execução de inserções das consultas.

STL_INSERT permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_INSERT contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
slice	integer	O número que identifica a fatia em que a consulta estava sendo executada.
segment	integer	O número que identifica o segmento da consulta.
etapa	integer	Etapa da consulta que foi executada.
starttime	timestamp	O horário (em UTC) de início da consulta. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Horário em UTC em que a consulta foi finalizada. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
tasknum	integer	Número do processo de tarefa de consulta que foi atribuído para executar a etapa.
rows	bigint	O número total de linhas que foram processadas.
tbl	integer	ID da tabela.
inserted_mega_value	character(1)	Essas informações são somente para uso interno. Essas informações mostram se a etapa de inserção especificada inseriu um valor grande. Um valor grande será armazenado em vários blocos. O tamanho do bloco é de 1 MB por

Nome da coluna	Tipo de dados	Descrição
		padrão; um valor grande é maior que 1 MB em uma configuração padrão.

Consultas de exemplo

O exemplo a seguir retorna as etapas de execução de inserção para a consulta mais recente.

```
select slice, segment, step, tasknum, rows, tbl
from stl_insert
where query=pg_last_query_id();
```

```
 slice | segment | step | tasknum | rows  |  tbl
-------+---------+------+---------+-------+--------
     0 |       2 |    2 |      15 | 24958 | 100548
     1 |       2 |    2 |      15 | 25032 | 100548
(2 rows)
```

STL_LIMIT

Analisa as etapas de execução que ocorrem quando uma cláusula LIMIT é usada em uma consulta SELECT.

STL_LIMIT permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_LIMIT contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
slice	integer	O número que identifica a fatia em que a consulta estava sendo executada.
segment	integer	O número que identifica o segmento da consulta.
etapa	integer	Etapa da consulta que foi executada.
starttime	timestamp	O horário (em UTC) de início da consulta. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Horário em UTC em que a consulta foi finalizada. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
tasknum	integer	Número do processo de tarefa de consulta que foi atribuído para executar a etapa.
rows	bigint	O número total de linhas que foram processadas.
soma de verificação	bigint	Essas informações são somente para uso interno.

Consultas de exemplo

Para gerar uma linha na tabela STL_LIMIT, este exemplo primeiro executa a consulta a seguir, com a tabela VENUE usando a cláusula LIMIT.

```
select * from venue
order by 1
limit 10;
```

```
 venueid |         venuename          |    venuecity    | venuestate | venueseats
---------+----------------------------+-----------------+------------+------------
       1 | Toyota Park                | Bridgeview      | IL         |          0
       2 | Columbus Crew Stadium      | Columbus        | OH         |          0
       3 | RFK Stadium                | Washington      | DC         |          0
       4 | CommunityAmerica Ballpark  | Kansas City     | KS         |          0
       5 | Gillette Stadium           | Foxborough      | MA         |      68756
       6 | New York Giants Stadium    | East Rutherford | NJ         |      80242
       7 | BMO Field                  | Toronto         | ON         |          0
       8 | The Home Depot Center      | Carson          | CA         |          0
       9 | Dick's Sporting Goods Park | Commerce City   | CO         |          0
      10 | Pizza Hut Park             | Frisco          | TX         |          0
(10 rows)
```

Em seguida, execute as seguintes consultas para encontrar o ID de consulta da última consulta que você executou com a tabela VENUE.

```
select max(query)
from stl_query;
```

```
  max
--------
 127128
(1 row)
```

Opcionalmente, execute a seguinte consulta para verificar se o ID de consulta corresponde à consulta LIMIT que você executou anteriormente.

```
select query, trim(querytxt)
from stl_query
where query=127128;
```

```
 query  |                 btrim
--------+----------------------------------------
 127128 | select * from venue order by 1 limit 10;
```

```
(1 row)
```

Finalmente, execute a seguinte consulta para obter informações sobre a consulta LIMIT da tabela STL_LIMIT.

```
select slice, segment, step, starttime, endtime, tasknum
from stl_limit
where query=127128
order by starttime, endtime;
```

```
  slice | segment | step |         starttime          |          endtime           |
 tasknum
 -------+---------+------+----------------------------+----------------------------
+---------
      1 |       1 |    3 | 2013-09-06 22:56:43.608114 | 2013-09-06 22:56:43.609383 |
   15
      0 |       1 |    3 | 2013-09-06 22:56:43.608708 | 2013-09-06 22:56:43.609521 |
   15
  10000 |       2 |    2 | 2013-09-06 22:56:43.612506 | 2013-09-06 22:56:43.612668 |
    0
(3 rows)
```

STL_LOAD_COMMITS

Retorna informações para rastrear ou solucionar problemas com uma carga de dados.

Essa visualização registra o progresso de cada arquivo de dados à medida que é carregado em uma tabela de banco de dados.

STL_LOAD_COMMITS permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_LOAD_COMMITS contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS

SYS_LOAD_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
slice	integer	A fatia carregada para essa entrada.
name	character (256)	O valor definido pelo sistema.
filename	character(256)	O nome do arquivo que está sendo rastreado.
byte_offset	integer	Essas informações são somente para uso interno.
lines_scanned	integer	O número de linhas pesquisadas na varredura do arquivo carregado. Esse número pode não corresponder ao número de linhas que são realmente carregadas. Por exemplo, a carga pode fazer a varredura mas tolerar um certo número de registros ruins com base na opção MAXERROR no comando COPY.
erros	integer	Essas informações são somente para uso interno.
curtime	timestamp	O horário em que essa entrada foi atualizada pela última vez.
status	integer	Essas informações são somente para uso interno.
file_format	character(16)	Formato do arquivo carregado. Os valores possíveis são: • Avro • JSON • ORC

Nome da coluna	Tipo de dados	Descrição
		• Parquet • Texto
is_parcial	integer	Valor que, se true (1), indica que o arquivo de entrada é dividido em intervalos durante uma operação COPY. Se esse valor for false (0), o arquivo de entrada não será dividido.
start_offset	bigint	Valor que, se o arquivo de entrada for dividido durante uma operação COPY, indica o valor de deslocamento da divisão (em bytes). Cada divisão de arquivo é registrada como um registro separado com o valor start_offset correspondente. Se o arquivo não estiver dividido, esse valor será 0.
copy_job_id	bigint	O identificador do trabalho de cópia. `0` indica que não há nenhum identificador de trabalho.

Consultas de exemplo

O exemplo a seguir retorna os detalhes da última operação COPY.

```
select query, trim(filename) as file, curtime as updated
from stl_load_commits
where query = pg_last_copy_id();

 query |               file                |          updated
-------+-----------------------------------+---------------------------
 28554 | s3://dw-tickit/category_pipe.txt | 2013-11-01 17:14:52.648486
(1 row)
```

A consulta a seguir contém as entradas de uma carga recente de tabelas para o banco de dados TICKIT:

```
select query, trim(filename), curtime
from stl_load_commits
where filename like '%tickit%' order by query;
```

```
 query |           btrim           |          curtime
-------+---------------------------+---------------------------
 22475 | tickit/allusers_pipe.txt  | 2013-02-08 20:58:23.274186
 22478 | tickit/venue_pipe.txt     | 2013-02-08 20:58:25.070604
 22480 | tickit/category_pipe.txt  | 2013-02-08 20:58:27.333472
 22482 | tickit/date2008_pipe.txt  | 2013-02-08 20:58:28.608305
 22485 | tickit/allevents_pipe.txt | 2013-02-08 20:58:29.99489
 22487 | tickit/listings_pipe.txt  | 2013-02-08 20:58:37.632939
 22593 | tickit/allusers_pipe.txt  | 2013-02-08 21:04:08.400491
 22596 | tickit/venue_pipe.txt     | 2013-02-08 21:04:10.056055
 22598 | tickit/category_pipe.txt  | 2013-02-08 21:04:11.465049
 22600 | tickit/date2008_pipe.txt  | 2013-02-08 21:04:12.461502
 22603 | tickit/allevents_pipe.txt | 2013-02-08 21:04:14.785124
 22605 | tickit/listings_pipe.txt  | 2013-02-08 21:04:20.170594

(12 rows)
```

O fato de um registro ser gravado no arquivo de log para esta visualização do sistema não significa que o carregamento foi confirmado com êxito como parte de sua transação contida. Para verificar as confirmações de carregamento, consulte a visualização STL_UTILITYTEXT e procure o registro COMMIT que corresponde a uma transação COPY. Por exemplo, essa consulta une as tabelas STL_LOAD_COMMITS e STL_QUERY com base em uma subconsulta com a tabela STL_UTILITYTEXT:

```
select l.query,rtrim(l.filename),q.xid
from stl_load_commits l, stl_query q
where l.query=q.query
and exists
(select xid from stl_utilitytext where xid=q.xid and rtrim("text")='COMMIT');

 query |           rtrim           |  xid
-------+---------------------------+-------
 22600 | tickit/date2008_pipe.txt  | 68311
 22480 | tickit/category_pipe.txt  | 68066
  7508 | allusers_pipe.txt         | 23365
  7552 | category_pipe.txt         | 23415
  7576 | allevents_pipe.txt        | 23429
  7516 | venue_pipe.txt            | 23390
  7604 | listings_pipe.txt         | 23445
 22596 | tickit/venue_pipe.txt     | 68309
 22605 | tickit/listings_pipe.txt  | 68316
 22593 | tickit/allusers_pipe.txt  | 68305
```

```
 22485 | tickit/allevents_pipe.txt | 68071
  7561 | allevents_pipe.txt        | 23429
  7541 | category_pipe.txt         | 23415
  7558 | date2008_pipe.txt         | 23428
 22478 | tickit/venue_pipe.txt     | 68065
   526 | date2008_pipe.txt         |  2572
  7466 | allusers_pipe.txt         | 23365
 22482 | tickit/date2008_pipe.txt  | 68067
 22598 | tickit/category_pipe.txt  | 68310
 22603 | tickit/allevents_pipe.txt | 68315
 22475 | tickit/allusers_pipe.txt  | 68061
   547 | date2008_pipe.txt         |  2572
 22487 | tickit/listings_pipe.txt  | 68072
  7531 | venue_pipe.txt            | 23390
  7583 | listings_pipe.txt         | 23445
(25 rows)
```

Os exemplos a seguir destacam os valores de coluna is_partial e start_offset.

```
-- Single large file copy without scan range
SELECT count(*) FROM stl_load_commits WHERE query = pg_last_copy_id();
1

-- Single large uncompressed, delimited file copy with scan range
SELECT count(*) FROM stl_load_commits WHERE query = pg_last_copy_id();
16

-- Scan range offset logging in the file at 64MB boundary.
SELECT start_offset FROM stl_load_commits
WHERE query = pg_last_copy_id() ORDER BY start_offset;
0
67108864
134217728
201326592
268435456
335544320
402653184
469762048
536870912
603979776
671088640
738197504
805306368
```

```
872415232
939524096
1006632960
```

STL_LOAD_ERRORS

Exibe os registros de todos os erros de carregamento do Amazon Redshift.

STL_LOAD_ERRORS contém um histórico de todos os erros de carregamento do Amazon Redshift. Consulte Referência de erros de carregamento para obter uma lista abrangente de possíveis erros de carregamento e explicações.

Consulte STL_LOADERROR_DETAIL para obter detalhes adicionais, como a linha de dados exata e coluna onde ocorreu um erro de análise, depois de consultar STL_LOAD_ERRORS para descobrir informações gerais sobre o erro.

STL_LOAD_ERRORS permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_LOAD_ERRORS contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_LOAD_ERROR_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
slice	integer	A fatia onde o erro ocorreu.

Nome da coluna	Tipo de dados	Descrição
tbl	integer	ID da tabela.
starttime	timestamp	O horário (em UTC) de início do carregamento.
sessão	integer	O ID de sessão para a sessão executando o carregamento.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
filename	character(256)	O caminho completo do arquivo de entrada para a carga.
line_number	bigint	O número da linha no arquivo de carga com o erro. Para o comando COPY do JSON, o número da linha da última linha do objeto JSON com o erro.
colname	character(127)	O campo com erro.
type	character(10)	O tipo de dados do campo.
col_length	character(10)	O tamanho da coluna, se aplicável. Este campo é preenchido quando o tipo de dados tem um limite de tamanho. Por exemplo, uma coluna com um tipo de dados "character(3)" conterá o valor "3".
position	integer	A posição do erro no campo.
raw_line	character(1024)	Os dados de carga brutos que contêm o erro. Os caracteres multibyte nos dados de carga são substituídos por um ponto.
raw_field_value	char(1024)	O valor da análise preliminar para o campo "colname" que leva ao erro de análise.
err_code	integer	Código de erro.
err_reason	character(100)	A explicação para o erro.

Nome da coluna	Tipo de dados	Descrição
is_parcial	integer	Valor que, se true (1), indica que o arquivo de entrada é dividido em intervalos durante uma operação COPY. Se esse valor for false (0), o arquivo de entrada não será dividido.
start_offset	bigint	Valor que, se o arquivo de entrada for dividido durante uma operação COPY, indica o valor de deslocamento da divisão (em bytes). Se o número da linha no arquivo for desconhecido, o número da linha será -1. Se o arquivo não estiver dividido, esse valor será 0.
copy_job_id	bigint	O identificador do trabalho de cópia. `0` indica que não há nenhum identificador de trabalho.

Consultas de exemplo

A consulta a seguir une as tabelas STL_LOAD_ERRORS e STL_LOADERROR_DETAIL para exibir os detalhes dos erros que ocorreram durante a carga mais recente.

```
select d.query, substring(d.filename,14,20),
d.line_number as line,
substring(d.value,1,16) as value,
substring(le.err_reason,1,48) as err_reason
from stl_loaderror_detail d, stl_load_errors le
where d.query = le.query
and d.query = pg_last_copy_id();

 query |    substring      | line |  value   |             err_reason
-------+-------------------+------+----------+----------------------------
   558| allusers_pipe.txt |  251 | 251      | String contains invalid or
                                              unsupported UTF8 code
   558| allusers_pipe.txt |  251 | ZRU29FGR | String contains invalid or
                                              unsupported UTF8 code
   558| allusers_pipe.txt |  251 | Kaitlin  | String contains invalid or
                                              unsupported UTF8 code
   558| allusers_pipe.txt |  251 | Walter   | String contains invalid or
```

```
                                                    unsupported UTF8 code
```

O exemplo a seguir usa a tabela STL_LOAD_ERRORS com a STV_TBL_PERM para criar uma exibição nova e, em seguida, usa essa exibição para determinar que erros ocorreram durante o carregamento de dados na tabela EVENT:

```
create view loadview as
(select distinct tbl, trim(name) as table_name, query, starttime,
trim(filename) as input, line_number, colname, err_code,
trim(err_reason) as reason
from stl_load_errors sl, stv_tbl_perm sp
where sl.tbl = sp.id);
```

Em seguida, a consulta abaixo retorna o último erro ocorrido durante o carregamento na tabela EVENT:

```
select table_name, query, line_number, colname, starttime,
trim(reason) as error
from loadview
where table_name ='event'
order by line_number limit 1;
```

A consulta retorna o último erro de carga ocorrido na tabela EVENT. Se não houver erros de carga, a consulta retorna zero linhas. Neste exemplo, a consulta retorna um único erro:

```
 table_name | query | line_number | colname | error | starttime
------+-----+----+----+-----------------------------------------------------------
+----------------------
event | 309 |  0 |  5 | Error in Timestamp value or format [%Y-%m-%d %H:%M:%S] |
 2014-04-22 15:12:44

(1 row)
```

Nos casos em que o comando COPY divide automaticamente dados de arquivos grandes, descompactados e delimitados por texto para facilitar o paralelismo, as colunas line_number, is_parcial e start_offset exibem as informações relativas às divisões. (O número da linha pode ser desconhecido nos casos em que o número da linha do arquivo original não está disponível.)

```
--scan ranges information
```

```
SELECT line_number, POSITION, btrim(raw_line), btrim(raw_field_value),
btrim(err_reason), is_partial, start_offset FROM stl_load_errors
WHERE query = pg_last_copy_id();

--result
-1,51,"1008771|13463413|463414|2|28.00|38520.72|0.06|0.07|NO|1998-08-30|1998-09-25|
1998-09-04|TAKE BACK RETURN|RAIL|ans cajole sly","NO","Char length exceeds DDL
 length",1,67108864
```

STL_LOADERROR_DETAIL

Exibe um log de erros de análise de dados que ocorreram no uso do comando COPY para carregar tabelas. Para poupar espaço em disco, um máximo 20 erros por fatia de nó é registrado para cada operação de carga.

Um erro de análise ocorre quando o Amazon Redshift não consegue analisar um campo em uma linha de dados ao carregá-lo em uma tabela. Por exemplo, se uma coluna da tabela está esperando um tipo de dados inteiro e o arquivo de dados contém uma string de caracteres alfabéticos no campo, isso causa um erro de análise.

Consulte a tabela STL_LOADERROR_DETAIL para obter detalhes adicionais, como a linha e a coluna exatas de um erro de análise, depois de consultar a tabela STL_LOAD_ERRORS para obter as informações gerais sobre o erro.

A visualização STL_LOADERROR_DETAIL contém todas as colunas de dados, incluindo e antes da coluna onde ocorreu o erro de análise. Use o campo VALUE para visualizar os valores dos dados que foram analisados nesta coluna, incluindo as colunas que foram analisadas corretamente até o erro ocorrer.

Esta visualização é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_LOADERROR_DETAIL contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS

SYS_LOAD_ERROR_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
slice	integer	A fatia onde o erro ocorreu.
sessão	integer	O ID de sessão para a sessão executando o carregamento.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
filename	character(256)	O caminho completo do arquivo de entrada para a carga.
line_number	bigint	O número da linha no arquivo de carga com o erro.
field	integer	O campo com erro.
colname	character(1024)	Nome da coluna.
valor	character(1024)	O valor do dado analisado no campo. (Ele pode estar truncado). Os caracteres multibyte nos dados de carga são substituídos por um ponto.
is_null	integer	Indica se o valor analisado é nulo.
type	character(10)	O tipo de dados do campo.
col_length	character(10)	O tamanho da coluna, se aplicável. Este campo é preenchido quando o tipo de dados tem um limite de tamanho. Por exemplo, uma coluna com um tipo de dados "character(3)" conterá o valor "3".

Consulta de exemplo

A consulta a seguir une as tabelas STL_LOAD_ERRORS e STL_LOADERROR_DETAIL para exibir os detalhes de um erro de análise ocorrido durante o carregamento da tabela EVENT, cujo ID de tabela é 100133:

```
select d.query, d.line_number, d.value,
le.raw_line, le.err_reason
from stl_loaderror_detail d, stl_load_errors le
where
d.query = le.query
and tbl = 100133;
```

O exemplo de saída a seguir mostra as colunas carregadas com êxito, incluindo a coluna com o erro. Neste exemplo, duas colunas foram carregadas com êxito antes de ocorrer o erro de análise na terceira coluna, onde uma string de caracteres foi analisada incorretamente para um campo que esperava um número inteiro. Como o campo esperava um inteiro, ele analisou a string "aaa", que é um dado não inicializado, como um caractere nulo e gerou um erro de análise. A saída mostra o valor bruto, o valor analisado e motivo do erro:

```
query  | line_number | value | raw_line | err_reason
-------+-------------+-------+----------+----------------
4      |      3      |  1201 |  1201    | Invalid digit
4      |      3      |   126 |   126    | Invalid digit
4      |      3      |       |   aaa    | Invalid digit
(3 rows)
```

Quando uma consulta une as tabelas STL_LOAD_ERRORS e STL_LOADERROR_DETAIL, ela exibe um motivo de erro para cada coluna da linha de dados, o que significa simplesmente que ocorreu um erro nessa linha. A última linha dos resultados é a coluna onde o erro de análise ocorreu.

STL_MERGE

Analisa as etapas de execução de mesclagem para as consultas. Essas etapas ocorrem quando os resultados de operações paralelas (como classificações e junções) são mesclados no processamento subsequente.

STL_MERGE permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_MERGE contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
slice	integer	O número que identifica a fatia em que a consulta estava sendo executada.
segment	integer	O número que identifica o segmento da consulta.
etapa	integer	Etapa da consulta que foi executada.
starttime	timestamp	O horário (em UTC) de início da consulta. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Horário em UTC em que a consulta foi finalizada. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .

Nome da coluna	Tipo de dados	Descrição
tasknum	integer	Número do processo de tarefa de consulta que foi atribuído para executar a etapa.
rows	bigint	O número total de linhas que foram processadas.

Consultas de exemplo

O exemplo a seguir retorna 10 resultados de execuções de mesclagem.

```
select query, step, starttime, endtime, tasknum, rows
from stl_merge
limit 10;
```

```
 query | step |       starttime     |        endtime      | tasknum | rows
-------+------+---------------------+---------------------+---------+------
     9 |    0 | 2013-08-12 20:08:14 | 2013-08-12 20:08:14 |       0 |    0
    12 |    0 | 2013-08-12 20:09:10 | 2013-08-12 20:09:10 |       0 |    0
    15 |    0 | 2013-08-12 20:10:24 | 2013-08-12 20:10:24 |       0 |    0
    20 |    0 | 2013-08-12 20:11:27 | 2013-08-12 20:11:27 |       0 |    0
    26 |    0 | 2013-08-12 20:12:28 | 2013-08-12 20:12:28 |       0 |    0
    32 |    0 | 2013-08-12 20:14:33 | 2013-08-12 20:14:33 |       0 |    0
    38 |    0 | 2013-08-12 20:16:43 | 2013-08-12 20:16:43 |       0 |    0
    44 |    0 | 2013-08-12 20:17:05 | 2013-08-12 20:17:05 |       0 |    0
    50 |    0 | 2013-08-12 20:18:48 | 2013-08-12 20:18:48 |       0 |    0
    56 |    0 | 2013-08-12 20:20:48 | 2013-08-12 20:20:48 |       0 |    0
(10 rows)
```

STL_MERGEJOIN

Analisa as etapas de execução da junção de mesclagem para as consultas.

STL_MERGEJOIN permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_MERGEJOIN contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
slice	integer	O número que identifica a fatia em que a consulta estava sendo executada.
segment	integer	O número que identifica o segmento da consulta.
etapa	integer	Etapa da consulta que foi executada.
starttime	timestamp	O horário (em UTC) de início da consulta. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Horário em UTC em que a consulta foi finalizada. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .

Nome da coluna	Tipo de dados	Descrição
tasknum	integer	Número do processo de tarefa de consulta que foi atribuído para executar a etapa.
rows	bigint	O número total de linhas que foram processadas.
tbl	integer	ID da tabela. Este é o ID da tabela interna que foi usada na junção de mesclagem.
soma de verificação	bigint	Essas informações são somente para uso interno.

Consultas de exemplo

O exemplo a seguir retorna os resultados da junção de mesclagem para a consulta mais recente.

```
select sum(s.qtysold), e.eventname
from event e, listing l, sales s
where e.eventid=l.eventid
and l.listid= s.listid
group by e.eventname;

select * from stl_mergejoin where query=pg_last_query_id();
```

```
 userid | query | slice | segment | step |         starttime   |          endtime    |
 tasknum | rows | tbl
--------+-------+-------+---------+------+---------------------+---------------------
+---------+------+-----
    100 | 27399 |     3 |       4 |    4 | 2013-10-02 16:30:41 | 2013-10-02 16:30:41 |
     19 |43428 | 240
    100 | 27399 |     0 |       4 |    4 | 2013-10-02 16:30:41 | 2013-10-02 16:30:41 |
     19 |43159 | 240
    100 | 27399 |     2 |       4 |    4 | 2013-10-02 16:30:41 | 2013-10-02 16:30:41 |
     19 |42778 | 240
    100 | 27399 |     1 |       4 |    4 | 2013-10-02 16:30:41 | 2013-10-02 16:30:41 |
     19 |43091 | 240
```

STL_MV_STATE

A visualização STL_MV_STATE contém uma linha para cada transição de estado de uma visão materializada.

Para obter mais informações sobre visões materializadas, consulte Visões materializadas no Amazon Redshift.

STL_MV_STATE é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_MV_STATE. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	bigint	O ID do usuário que criou o evento.
starttime	timestamp	A hora de início do evento.
xid	bigint	O id da transação do evento.
event_desc	char(500)	O evento que acionou a alteração de estado. Alguns exemplos de valores incluem o seguinte: • O tipo de coluna foi alterado • A coluna foi descartada • A coluna foi renomeada • O nome do esquema foi alterado • Conversão de mesa pequena • TRUNCATE • Vácuo

Nome da coluna	Tipo de dados	Descrição
		Observe que há outros valores possíveis para essa coluna.
db_name	char(128)	O banco de dados que contém a visualização materializada.
base_table_schema	char(128)	O esquema da tabela base.
base_table_name	char(128)	O nome da tabela base.
mv_schema	char(128)	O esquema da visão materializada.
mv_name	char(128)	O nome da visualização materializada.
estado	character(32)	O estado alterado da visualização materializada, como segue: • Computar novamente • Não atualizável

A tabela a seguir mostra exemplos de combinações de `event_desc` e `state`.

```
          event_desc     |     state
-------------------------+---------------
 TRUNCATE                | Recompute
 TRUNCATE                | Recompute
 Small table conversion  | Recompute
 Vacuum                  | Recompute
 Column was renamed      | Unrefreshable
 Column was dropped      | Unrefreshable
 Table was renamed       | Unrefreshable
 Column type was changed | Unrefreshable
 Schema name was changed | Unrefreshable
```

Consulta de exemplo

Para exibir o log de transições de estado de visualizações materializadas, execute a consulta a seguir.

```
select * from stl_mv_state;
```

Essa consulta retorna os seguintes dados de saída de exemplo:

```
 userid |         starttime          | xid  |            event_desc       | db_name |
 base_table_schema   |   base_table_name    |      mv_schema       | mv_name       |
  state
--------+----------------------------+------+-----------------------------+---------
+---------------------+---------------------+---------------------+--------------
+---------------
    138 | 2020-02-14 02:21:25.578885 | 5180 | TRUNCATE                    | dev     |
 public              | mv_base_table        | public              | mv_test       |
 Recompute
    138 | 2020-02-14 02:21:56.846774 | 5275 | Column was dropped          | dev     |
                     | mv_base_table        | public              | mv_test       |
 Unrefreshable
    100 | 2020-02-13 22:09:53.041228 | 1794 | Column was renamed          | dev     |
                     | mv_base_table        | public              | mv_test       |
 Unrefreshable
      1 | 2020-02-13 22:10:23.630914 | 1893 | ALTER TABLE ALTER SORTKEY   | dev     |
 public              | mv_base_table_sorted | public              | mv_test       |
 Recompute
      1 | 2020-02-17 22:57:22.497989 | 8455 | ALTER TABLE ALTER DISTSTYLE | dev     |
 public              | mv_base_table        | public              | mv_test       |
 Recompute
    173 | 2020-02-17 22:57:23.591434 | 8504 | Table was renamed           | dev     |
                     | mv_base_table        | public              | mv_test       |
 Unrefreshable
    173 | 2020-02-17 22:57:27.229423 | 8592 | Column type was changed     | dev     |
                     | mv_base_table        | public              | mv_test       |
 Unrefreshable
    197 | 2020-02-17 22:59:06.212569 | 9668 | TRUNCATE                    | dev     |
 schemaf796e415850f4f | mv_base_table        | schemaf796e415850f4f | mv_test       |
 Recompute
    138 | 2020-02-14 02:21:55.705655 | 5226 | Column was renamed          | dev     |
                     | mv_base_table        | public              | mv_test       |
 Unrefreshable
```

```
     1 | 2020-02-14 02:22:26.292434 | 5325 | ALTER TABLE ALTER SORTKEY   | dev     |
public               | mv_base_table_sorted | public               | mv_test      |
Recompute
```

STL_NESTLOOP

Analisa as etapas de execução da junção de loops aninhados para as consultas.

STL_NESTLOOP permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_NESTLOOP contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
slice	integer	O número que identifica a fatia em que a consulta estava sendo executada.
segment	integer	O número que identifica o segmento da consulta.
etapa	integer	Etapa da consulta que foi executada.

Nome da coluna	Tipo de dados	Descrição
starttime	timestamp	O horário (em UTC) de início da consulta. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **2009-06-12 11:29:19.131358** .
endtime	timestamp	Horário em UTC em que a consulta foi finalizada. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **2009-06-12 11:29:19.131358** .
tasknum	integer	Número do processo de tarefa de consulta que foi atribuído para executar a etapa.
rows	bigint	O número total de linhas que foram processadas.
tbl	integer	ID da tabela.
soma de verificação	bigint	Essas informações são somente para uso interno.

Consultas de exemplo

Como a consulta a seguir omite a junção da tabela CATEGORY, ela gera um produto cartesiano parcial, o que não é recomendado. Ela está sendo mostrada aqui para ilustrar um loop aninhado.

```
select count(event.eventname), event.eventname, category.catname, date.caldate
from event, category, date
where event.dateid = date.dateid
group by event.eventname, category.catname, date.caldate;
```

A consulta a seguir mostra os resultados da consulta anterior na visualização STL_NESTLOOP.

```
select query, slice, segment as seg, step,
datediff(msec, starttime, endtime) as duration, tasknum, rows, tbl
from stl_nestloop
where query = pg_last_query_id();
```

```
 query | slice | seg | step | duration | tasknum | rows  | tbl
-------+-------+-----+------+----------+---------+-------+-----
  6028 |     0 |   4 |    5 |       41 |      22 | 24277 | 240
  6028 |     1 |   4 |    5 |       26 |      23 | 24189 | 240
  6028 |     3 |   4 |    5 |       25 |      23 | 24376 | 240
  6028 |     2 |   4 |    5 |       54 |      22 | 23936 | 240
```

STL_PARSE

Analisa as etapas de consulta que analisam strings e as definem como valores binários para o carregamento.

STL_PARSE permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_PARSE contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
slice	integer	O número que identifica a fatia em que a consulta estava sendo executada.

Nome da coluna	Tipo de dados	Descrição
segment	integer	O número que identifica o segmento da consulta.
etapa	integer	Etapa da consulta que foi executada.
starttime	timestamp	O horário (em UTC) de início da consulta. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Horário em UTC em que a consulta foi finalizada. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
tasknum	integer	Número do processo de tarefa de consulta que foi atribuído para executar a etapa.
rows	bigint	O número total de linhas que foram processadas.

Consultas de exemplo

O exemplo a seguir retorna todos os resultados das etapas da consulta para a fatia 1 e o segmento 0, onde as strings são analisadas e definidas como valores binários.

```
select query, step, starttime, endtime, tasknum, rows
from stl_parse
where slice=1 and segment=0;
```

```
 query | step |     starttime       |        endtime      | tasknum |  rows
-------+------+---------------------+---------------------+---------+--------
   669 |    1 | 2013-08-12 22:35:13 | 2013-08-12 22:35:17 |      32 | 192497
   696 |    1 | 2013-08-12 22:35:49 | 2013-08-12 22:35:49 |      32 |      0
   525 |    1 | 2013-08-12 22:32:03 | 2013-08-12 22:32:03 |      13 |  49990
   585 |    1 | 2013-08-12 22:33:18 | 2013-08-12 22:33:19 |      13 |    202
   621 |    1 | 2013-08-12 22:34:03 | 2013-08-12 22:34:03 |      27 |    365
   651 |    1 | 2013-08-12 22:34:47 | 2013-08-12 22:34:53 |      35 | 192497
```

```
   590 |    1 | 2013-08-12 22:33:28 | 2013-08-12 22:33:28 |      19 |      0
   599 |    1 | 2013-08-12 22:33:39 | 2013-08-12 22:33:39 |      31 |     11
   675 |    1 | 2013-08-12 22:35:26 | 2013-08-12 22:35:27 |      38 |   3766
   567 |    1 | 2013-08-12 22:32:47 | 2013-08-12 22:32:48 |      23 |  49990
   630 |    1 | 2013-08-12 22:34:17 | 2013-08-12 22:34:17 |      36 |      0
   572 |    1 | 2013-08-12 22:33:04 | 2013-08-12 22:33:04 |      29 |      0
   645 |    1 | 2013-08-12 22:34:37 | 2013-08-12 22:34:38 |      29 |   8798
   604 |    1 | 2013-08-12 22:33:47 | 2013-08-12 22:33:47 |      37 |      0
(14 rows)
```

STL_PLAN_INFO

Use a visualização STL_PLAN_INFO para examinar a saída EXPLAIN de uma consulta em termos de um conjunto de linhas. Essa é uma forma alternativa de ver os planos de consulta.

STL_PLAN_INFO permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_PLAN_INFO contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.

Nome da coluna	Tipo de dados	Descrição
nodeid	integer	O identificador do nó de plano, onde o nó mapeia para uma ou mais etapas na execução da consulta.
segment	integer	O número que identifica o segmento da consulta.
etapa	integer	O número que identifica a etapa da consulta.
locus	integer	Local onde a etapa é executada: 0 se estiver em um nó de computação e 1 se estiver no nó líder.
plannode	integer	O valor enumerado do nó de plano. Veja a tabela a seguir com os valores enumerados dos nós de plano. (A coluna PLANNODE na tabela STL_EXPLAIN contém o texto do nó de plano).
startupcost	double precision	O custo relativo estimado de retorno da primeira linha para esta etapa.
totalcost	double precision	O custo relativo estimado de execução da etapa.
rows	bigint	O número estimado de linhas que serão produzidas pela etapa.
bytes	bigint	O número estimado de bytes que serão produzidos pela etapa.

Consultas de exemplo

Os exemplos a seguir comparam os planos de consulta para uma consulta SELECT simples retornada usando o comando EXPLAIN e consultando a visualização STL_PLAN_INFO.

```
explain select * from category;
QUERY PLAN
------------------------------------------------------------
XN Seq Scan on category (cost=0.00..0.11 rows=11 width=49)
(1 row)
```

```
select * from category;
catid | catgroup | catname | catdesc
-------+----------+-----------+--------------------------------------------
1 | Sports | MLB | Major League Baseball
3 | Sports | NFL | National Football League
5 | Sports | MLS | Major League Soccer
...

select * from stl_plan_info where query=256;

query | nodeid | segment | step | locus | plannode | startupcost | totalcost
| rows | bytes
-------+--------+---------+------+-------+----------+-------------+-----------+------
+-------
256 | 1 | 0 | 1 | 0 | 104 | 0 | 0.11 | 11 | 539
256 | 1 | 0 | 0 | 0 | 104 | 0 | 0.11 | 11 | 539
(2 rows)
```

Neste exemplo, o PLANNODE 104 refere-se à varredura sequencial da tabela CATEGORY.

```
select distinct eventname from event order by 1;

eventname
----------------------------------------------------------------------------
.38 Special
3 Doors Down
70s Soul Jam
A Bronx Tale
...

explain select distinct eventname from event order by 1;

QUERY PLAN
------------------------------------------------------------------------------------------
XN Merge (cost=1000000000136.38..1000000000137.82 rows=576 width=17)
Merge Key: eventname
-> XN Network (cost=1000000000136.38..1000000000137.82 rows=576
width=17)
Send to leader
-> XN Sort (cost=1000000000136.38..1000000000137.82 rows=576
width=17)
Sort Key: eventname
-> XN Unique (cost=0.00..109.98 rows=576 width=17)
```

```
-> XN Seq Scan on event (cost=0.00..87.98 rows=8798
width=17)
(8 rows)

select * from stl_plan_info where query=240 order by nodeid desc;

query | nodeid | segment | step | locus | plannode | startupcost |
totalcost | rows | bytes
-------+--------+---------+------+-------+----------+-----------------
+------------------+------+--------
240 | 5 | 0 | 0 | 0 | 104 | 0                | 87.98   | 8798 | 149566
240 | 5 | 0 | 1 | 0 | 104 | 0                | 87.98   | 8798 | 149566
240 | 4 | 0 | 2 | 0 | 117 | 0                | 109.975 | 576  | 9792
240 | 4 | 0 | 3 | 0 | 117 | 0                | 109.975 | 576  | 9792
240 | 4 | 1 | 0 | 0 | 117 | 0                | 109.975 | 576  | 9792
240 | 4 | 1 | 1 | 0 | 117 | 0                | 109.975 | 576  | 9792
240 | 3 | 1 | 2 | 0 | 114 | 1000000000136.38 | 1000000000137.82 | 576 | 9792
240 | 3 | 2 | 0 | 0 | 114 | 1000000000136.38 | 1000000000137.82 | 576 | 9792
240 | 2 | 2 | 1 | 0 | 123 | 1000000000136.38 | 1000000000137.82 | 576 | 9792
240 | 1 | 3 | 0 | 0 | 122 | 1000000000136.38 | 1000000000137.82 | 576 | 9792
(10 rows)
```

STL_PROJECT

Contém as linhas das etapas de consulta que são usadas para avaliar expressões.

STL_PROJECT permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_PROJECT contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
slice	integer	O número que identifica a fatia em que a consulta estava sendo executada.
segment	integer	O número que identifica o segmento da consulta.
etapa	integer	Etapa da consulta que foi executada.
starttime	timestamp	O horário (em UTC) de início da consulta. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Horário em UTC em que a consulta foi finalizada. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
tasknum	integer	Número do processo de tarefa de consulta que foi atribuído para executar a etapa.
rows	bigint	O número total de linhas que foram processadas.
soma de verificação	bigint	Essas informações são somente para uso interno.

Consultas de exemplo

O exemplo a seguir retorna todas as linhas das etapas de consulta que foram usadas para avaliar expressões para a fatia 0 e o segmento 1.

```
select query, step, starttime, endtime, tasknum, rows
from stl_project
where slice=0 and segment=1;
```

```
 query  | step |         starttime   |          endtime    | tasknum | rows
--------+------+---------------------+---------------------+---------+------
  86399 |    2 | 2013-08-29 22:01:21 | 2013-08-29 22:01:21 |      25 |   -1
  86399 |    3 | 2013-08-29 22:01:21 | 2013-08-29 22:01:21 |      25 |   -1
    719 |    1 | 2013-08-12 22:38:33 | 2013-08-12 22:38:33 |       7 |   -1
  86383 |    1 | 2013-08-29 21:58:35 | 2013-08-29 21:58:35 |       7 |   -1
    714 |    1 | 2013-08-12 22:38:17 | 2013-08-12 22:38:17 |       2 |   -1
  86375 |    1 | 2013-08-29 21:57:59 | 2013-08-29 21:57:59 |       2 |   -1
  86397 |    2 | 2013-08-29 22:01:20 | 2013-08-29 22:01:20 |      19 |   -1
    627 |    1 | 2013-08-12 22:34:13 | 2013-08-12 22:34:13 |      34 |   -1
  86326 |    2 | 2013-08-29 21:45:28 | 2013-08-29 21:45:28 |      34 |   -1
  86326 |    3 | 2013-08-29 21:45:28 | 2013-08-29 21:45:28 |      34 |   -1
  86325 |    2 | 2013-08-29 21:45:27 | 2013-08-29 21:45:27 |      28 |   -1
  86371 |    1 | 2013-08-29 21:57:42 | 2013-08-29 21:57:42 |       4 |   -1
 111100 |    2 | 2013-09-03 19:04:45 | 2013-09-03 19:04:45 |      12 |   -1
    704 |    2 | 2013-08-12 22:36:34 | 2013-08-12 22:36:34 |      37 |   -1
    649 |    2 | 2013-08-12 22:34:47 | 2013-08-12 22:34:47 |      38 |   -1
    649 |    3 | 2013-08-12 22:34:47 | 2013-08-12 22:34:47 |      38 |   -1
    632 |    2 | 2013-08-12 22:34:22 | 2013-08-12 22:34:22 |      13 |   -1
    705 |    2 | 2013-08-12 22:36:48 | 2013-08-12 22:36:49 |      13 |   -1
    705 |    3 | 2013-08-12 22:36:48 | 2013-08-12 22:36:49 |      13 |   -1
      3 |    1 | 2013-08-12 20:07:40 | 2013-08-12 20:07:40 |       3 |   -1
  86373 |    1 | 2013-08-29 21:57:58 | 2013-08-29 21:57:58 |       3 |   -1
 107976 |    1 | 2013-09-03 04:05:12 | 2013-09-03 04:05:12 |       3 |   -1
  86381 |    1 | 2013-08-29 21:58:35 | 2013-08-29 21:58:35 |       8 |   -1
  86396 |    1 | 2013-08-29 22:01:20 | 2013-08-29 22:01:20 |      15 |   -1
    711 |    1 | 2013-08-12 22:37:10 | 2013-08-12 22:37:10 |      20 |   -1
  86324 |    1 | 2013-08-29 21:45:27 | 2013-08-29 21:45:27 |      24 |   -1
(26 rows)
```

STL_QUERY

Retorna as informações de execução de uma consulta de banco de dados.

> **Note**
>
> As visualizações STL_QUERY e STL_QUERYTEXT contêm apenas informações sobre consultas, não outro utilitário e comandos DDL. Para obter uma lista e informações sobre

todas as instruções executadas pelo Amazon Redshift, também é possível consultar as visualizações STL_DDLTEXT e STL_UTILITYTEXT. Para obter uma lista completa de todas as instruções executadas pelo Amazon Redshift, é possível consultar a visualização SVL_STATEMENTTEXT.

STL_QUERY permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_QUERY_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Observe que, quando o comprimento do texto da consulta é superior a 4 mil caracteres, STL_QUERY exibe somente dados truncados. Para obter o texto completo da consulta, você pode usar UNION no texto da consulta em todas as linhas.

Note

Para verificar se uma transação que contém a consulta executada foi confirmada com êxito, é necessário realizar uma operação de junção entre as tabelas do sistema e a tabela `sys_transaction_history`. Por exemplo:

```
SELECT
    stlq.xid AS transaction_id,
    stlq.query AS query_id,
    TRIM(stlq.querytxt) AS query_text,
    th.status AS transaction_status
FROM
    stl_query stlq
LEFT JOIN
    sys_transaction_history th ON stlq.xid = th.transaction_id;
```

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
rótulo	character(320)	O nome do arquivo usado para executar a consulta ou um rótulo definido com o comando SET QUERY_GRO UP. Se a consulta não for baseada em arquivos ou o parâmetro QUERY_GROUP não estiver definido, o valor deste campo será `default`.
xid	bigint	ID da transação.
pid	integer	ID do processo. Normalmente, todas as consultas de uma sessão são executadas no mesmo processo, portanto esse valor geralmente permanece constante se você executa uma série de consultas na mesma sessão. Após certos eventos internos, o Amazon Redshift pode reiniciar uma sessão ativa e atribuir um novo PID. Para obter mais informações, consulte STL_RESTA RTED_SESSIONS.
banco de dados	character(32)	O nome do banco de dados ao qual o usuário estava conectado quando a consulta foi enviada.
querytxt	character(4000)	O texto da consulta.
starttime	timestamp	O horário (em UTC) de início da consulta. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Horário em UTC em que a consulta foi finalizada. O tempo total inclui consultas e execução, com seis dígitos

Nome da coluna	Tipo de dados	Descrição
		de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
aborted	integer	Se uma consulta for interrompida pelo sistema ou cancelada pelo usuário, essa coluna terá o valor **`1`**. Se a consulta for executada até o final (inclusive retornando os resultados para o cliente), essa coluna conterá o valor **`0`**. Se o cliente desconectar antes de receber os resultado s, a consulta será marcada como cancelada (**`1`**), mesmo que ela tenha sido concluída com êxito no backend.
insert_pristine	integer	Se as consultas de gravação são/puderam ser executada s enquanto a consulta atual está/estava em execução. 1 = nenhuma consulta de gravação permitida. 0 = consultas de gravação permitidas. Essa coluna deve ser usada na depuração.
concurrency_scaling_status	integer	Indica se a consulta foi executada no cluster principal ou em um cluster de escalabilidade da simultaneidade. Os valores possíveis são: 0 - Executada no cluster principal 1 — Executada em um cluster de escalabilidade da simultaneidade Maior que 1 - Executada no cluster principal

Consultas de exemplo

A consulta a seguir lista as cinco consultas mais recentes.

```
select query, trim(querytxt) as sqlquery
from stl_query
order by query desc limit 5;
```

```
query |                                    sqlquery
------+-------------------------------------------------
129 | select query, trim(querytxt) from stl_query order by query;
128 | select node from stv_disk_read_speeds;
127 | select system_status from stv_gui_status
126 | select * from systable_topology order by slice
125 | load global dict registry
(5 rows)
```

A consulta a seguir retorna o tempo decorrido em ordem descendente para as consultas executadas no dia 15 de fevereiro de 2013.

```
select query, datediff(seconds, starttime, endtime),
trim(querytxt) as sqlquery
from stl_query
where starttime >= '2013-02-15 00:00' and endtime < '2013-02-16 00:00'
order by date_diff desc;

 query | date_diff |  sqlquery
-------+-----------+-----------------------------------------
 55    |       119 | padb_fetch_sample: select count(*) from category
121    |         9 | select * from svl_query_summary;
181    |         6 | select * from svl_query_summary where query in(179,178);
172    |         5 | select * from svl_query_summary where query=148;
...
(189 rows)
```

A consulta a seguir mostra o tempo da fila e o tempo de execução para consultas. Consultas com `concurrency_scaling_status = 1` executadas em um cluster de escalabilidade da simultaneidade. Todas as outras consultas executadas no cluster principal.

```
SELECT w.service_class AS queue
     , q.concurrency_scaling_status
     , COUNT( * ) AS queries
     , SUM( q.aborted )  AS aborted
     , SUM( ROUND( total_queue_time::NUMERIC / 1000000,2 ) ) AS queue_secs
     , SUM( ROUND( total_exec_time::NUMERIC / 1000000,2 ) )  AS exec_secs
FROM stl_query q
     JOIN stl_wlm_query w
          USING (userid,query)
WHERE q.userid > 1
  AND service_class > 5
```

```
  AND q.starttime > '2019-03-01 16:38:00'
  AND q.endtime   < '2019-03-01 17:40:00'
GROUP BY 1,2
ORDER BY 1,2;
```

STL_QUERY_METRICS

Contém as informações de métricas, como o número de linhas processadas, utilização da CPU, entrada/saída e utilização do disco, para consultas que concluíram a execução nas filas de consultas definidas pelo usuário (classes de serviço). Para visualizar as métricas das consultas ativas que estão em execução no momento, consulte a visualização do sistema STV_QUERY_METRICS.

Para as métricas de consulta, as amostras são feitas em intervalos de um segundo. Consequentemente, diferentes execuções da mesma consulta podem retornar tempos um pouco diferentes. Além disso, os segmentos das consultas que são executados em menos de 1 segundo podem não ser registrados.

A tabela STL_QUERY_METRICS rastreia e agrega as métricas no nível da consulta, do segmento e das etapas. Para obter informações sobre os segmentos e as etapas de uma consulta, acesse Planejamento de consulta e fluxo de trabalho de execução. Muitas métricas (como, por exemplo, `max_rows`, `cpu_time`, etc.) são resultado da soma obtida das fatias de nós. Para obter mais informações sobre as fatias de nós, consulte Arquitetura de sistema do data warehouse.

Para determinar o nível no qual uma linha relata suas métricas, examine as colunas `segment` e `step_type`.

- Se ambos `segment` e `step_type` são `-1`, a linha relata as métricas no nível da consulta.
- Se `segment` não é `-1` e `step_type` é `-1`, a linha relata as métricas no nível do segmento.
- Se ambos `segment` e `step_type` são diferentes de `-1`, a linha relata as métricas no nível da etapa.

As visualizações SVL_QUERY_METRICS e SVL_QUERY_METRICS_SUMMARY agregam os dados nesta visão e apresentam as informações de uma forma mais acessível.

STL_QUERY_METRICS permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que executou a consulta que gerou a entrada.
service_class	integer	ID da classe de serviço. As filas de consultas são definidas na configuração do WLM. As métricas são relatadas somente para as filas definidas pelo usuário.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
segment	integer	O número do segmento. Uma consulta consiste em vários segmentos e cada segmento consiste em uma ou mais etapas. Os segmentos de uma consulta podem ser executados em paralelo. Cada segmento executa em um único processo. Se o valor do segmento é -1, os valores das métricas do segmento são acumulados no nível de consulta.
step_type	integer	O tipo da etapa que foi executada. Para obter uma descrição dos tipos de etapas, consulte Tipos de etapas.
starttime	timestamp	O horário (em UTC) do início da execução da consulta, com 6 dígitos de precisão fracionária de segundos. Por exemplo: `2009-06-12 11:29:19.131358` .
slices	integer	O número de fatias do cluster.
max_rows	bigint	O número máximo de linhas geradas na saída de uma etapa, agregado de todas as fatias.

Nome da coluna	Tipo de dados	Descrição
rows	bigint	O número de linhas processadas por uma etapa.
max_cpu_time	bigint	O tempo máximo de CPU usado, em microssegundos. No nível de segmento, o tempo máximo de CPU usado pelo segmento em todas as fatias. No nível de consulta, o tempo máximo de CPU usado por qualquer um dos segmento da consulta.
cpu_time	bigint	O tempo de CPU usado, em microssegundos. No nível de segmento, o tempo total de CPU usado pelo segmento em todas as fatias. No nível de consulta, a soma dos tempos de CPU da consulta em todas as fatias e segmentos.
max_block s_read	integer	O número máximo de blocos de 1 MB lidos pelo segmento, agregado de todas as fatias. No nível de segmento, o número máximo de blocos de 1 MB lidos para o segmento em todas as fatias. No nível de consulta, o número máximo de blocos de 1 MB lidos por qualquer um dos segmentos da consulta.
blocks_read	bigint	O número de blocos de 1 MB lidos pela consulta ou segmento.
max_run_time	bigint	O tempo máximo decorrido para um segmento, em microsseg undos. No nível de segmento, o tempo máximo de execução para o segmento em todas as fatias. No nível de consulta, o tempo máximo de execução para qualquer um dos segmento da consulta.

Nome da coluna	Tipo de dados	Descrição
run_time	bigint	O tempo total de execução, somado de todas de fatias. O tempo de execução não inclui o tempo de espera. No nível de segmento, o tempo de execução para o segmento, somado de todas as fatias. No nível de consulta, o tempo de execução da consulta, somado de todas as fatias e segmentos. Como esse valor é uma soma, o tempo de execução não está relacionado ao tempo de execução de consulta.
max_block s_to_disk	bigint	A quantidade máxima de espaço em disco usada para gravar resultados intermediários, em blocos de MB. No nível de segmento, a quantidade máxima de espaço em disco usada pelo segmento em todas as fatias. No nível de consulta, a quantidade máxima de espaço em disco usada por qualquer um dos segmento da consulta.
blocks_to_disk	bigint	A quantidade de espaço em disco usada por uma consulta ou segmento para gravar resultados intermediários, em blocos de MB.
etapa	integer	Etapa da consulta que foi executada.
max_query _scan_size	bigint	A quantidade máxima de dados pesquisados em varredura por uma consulta, em MB. No nível de segmento, a quantidad e máxima de dados pesquisados em varredura por um segmento em todas as fatias. No nível de consulta, a quantidade máxima de dados pesquisados em varredura por qualquer um dos segmento da consulta.
query_scan_size	bigint	A quantidade de dados pesquisados em varredura por uma consulta, em MB.

Nome da coluna	Tipo de dados	Descrição
query_priority	integer	A prioridade da consulta. Os valores possíveis são -1, 0, 1, 2, 3 e 4, em que -1 significa que a prioridade da consulta não é compatível.
query_queue_time	bigint	O tempo em microssegundos que a consulta estava em fila.
service_class_name	character (64)	O nome da classe de serviços.

Consulta de exemplo

Para encontrar as consultas com alto nível de tempo de CPU (mais de 1.000 segundos), execute a consulta a seguir.

```
Select query, cpu_time / 1000000 as cpu_seconds
from stl_query_metrics where segment = -1 and cpu_time > 1000000000
order by cpu_time;

query | cpu_seconds
------+------------
25775 |        9540
```

Para encontrar as consultas ativas com uma junção de loop aninhado que retornaram mais de um milhão de linhas, execute a seguinte consulta.

```
select query, rows
from stl_query_metrics
where step_type = 15 and rows > 1000000
order by rows;

query | rows
------+-----------
25775 | 2621562702
```

Para encontrar as consultas ativas cujas execuções duraram mais de 60 segundos e que usaram menos de 10 segundos de tempo de CPU, execute a seguinte consulta.

```
select query, run_time/1000000 as run_time_seconds
from stl_query_metrics
where segment = -1 and run_time > 60000000 and cpu_time < 10000000;

query | run_time_seconds
------+-----------------
25775 |              114
```

STL_QUERYTEXT

Captura o texto da consulta para os comandos SQL.

Consulte a visualização STL_QUERYTEXT para capturar o SQL que foi registrado para as seguintes instruções:

- SELECT, SELECT INTO
- INSERT, UPDATE, DELETE
- COPY
- UNLOAD
- As consultas geradas executando VACUUM e ANALYZE
- CREATE TABLE AS (CTAS)

Para consultar a atividade dessas instruções em um determinado período de tempo, junte as visualizações STL_QUERYTEXT e STL_QUERY.

> **Note**
>
> As visualizações STL_QUERY e STL_QUERYTEXT contêm apenas informações sobre consultas, não outro utilitário e comandos DDL. Para obter uma lista e informações sobre todas as instruções executadas pelo Amazon Redshift, também é possível consultar as visualizações STL_DDLTEXT e STL_UTILITYTEXT. Para obter uma lista completa de todas as instruções executadas pelo Amazon Redshift, é possível consultar a visualização SVL_STATEMENTTEXT.

Consulte também STL_DDLTEXT, STL_UTILITYTEXT e SVL_STATEMENTTEXT.

STL_QUERYTEXT permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_QUERY_TEXT. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Observe que, quando o comprimento do texto da consulta é superior a 4000 caracteres, STL_QUERYTEXT exibe somente dados truncados. Para obter o texto completo da consulta, você pode usar UNION no texto da consulta em todas as linhas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
xid	bigint	ID da transação.
pid	integer	ID do processo. Normalmente, todas as consultas de uma sessão são executadas no mesmo processo, portanto esse valor geralmente permanece constante se você executa uma série de consultas na mesma sessão. Após certos eventos internos, o Amazon Redshift pode reiniciar uma sessão ativa e atribuir um novo PID. Para obter mais informações, consulte STL_RESTARTED_SESSIONS. Você pode usar esta coluna para ingressar na visualização STL_ERROR.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
sequence	integer	Quando uma única instrução contém mais de 200 caracteres, são registradas linhas adicionais para essa instrução. O valor 0 da sequência é a primeira linha, 1 é a segunda, e assim por diante.

Nome da coluna	Tipo de dados	Descrição
texto	character(200)	O texto em SQL, em incrementos de 200 caracteres. Esse campo pode conter caracteres especiais como barra invertida (\\) e nova linha (\n).
user_query_id	bigint	O identificador da consulta enviada pelo usuário, conforme registrado na coluna query_id de SYS_QUERY_TEXT.
child_query_sequence	integer	O número de sequência da consulta filha regravada na consulta do usuário, começando com 1.
query_uuid	character(36)	O identificador globalmente exclusivo (UUID) da consulta.

Consultas de exemplo

Você pode usar a função PG_BACKEND_PID() para recuperar informações da sessão atual. Por exemplo, a consulta a seguir retorna o ID de consulta e uma parte do texto da consulta para as consultas concluídas na sessão atual.

```
select query, substring(text,1,60)
from stl_querytext
where pid = pg_backend_pid()
order by query desc;

 query |                          substring
-------+-------------------------------------------------------------
 28262 | select query, substring(text,1,80) from stl_querytext where
 28252 | select query, substring(path,0,80) as path from stl_unload_l
 28248 | copy category from 's3://dw-tickit/manifest/category/1030_ma
 28247 | Count rows in target table
 28245 | unload ('select * from category') to 's3://dw-tickit/manifes
 28240 | select query, substring(text,1,40) from stl_querytext where
(6 rows)
```

Reconstrução de SQL armazenado

Para reconstruir o SQL armazenado na coluna `text` de STL_QUERYTEXT, execute uma instrução SELECT para criar SQL a partir de uma ou mais partes na coluna `text`. Antes de executar o SQL reconstruído, substitua os caracteres especiais por uma nova linha (\n). O resultado da instrução SELECT a seguir são linhas de SQL reconstruído no campo `query_statement`.

```
SELECT query, LISTAGG(CASE WHEN LEN(RTRIM(text)) = 0 THEN text ELSE RTRIM(text) END)
 WITHIN GROUP (ORDER BY sequence) as query_statement, COUNT(*) as row_count
FROM stl_querytext GROUP BY query ORDER BY query desc;
```

Por exemplo, a consulta a seguir seleciona três colunas. A consulta em si tem mais de 200 caracteres e é armazenada em partes em STL_QUERYTEXT.

```
select
1 AS a01234567890123456789012345678901234567890123456789012345678901234567890,
2 AS b01234567890123456789012345678901234567890123456789012345678901234567890,
3 AS b01234567890123456789012345678901234567890123456789012345678901234
FROM stl_querytext;
```

Nesse exemplo, a consulta é armazenada em duas partes (linhas) na coluna `text` de STL_QUERYTEXT.

```
select query, sequence, text
from stl_querytext where query=pg_last_query_id() order by query desc, sequence limit
 10;
```

```
query  | sequence |
                        text

-------+----------
+-----------------------------------------------------------------------------------------------------------------
    45 |        0 | select\n1 AS
 a01234567890123456789012345678901234567890123456789012345678901234567890,\n2 AS
 b01234567890123456789012345678901234567890123456789012345678901234567890,\n3 AS
 b01234567890123456789012345678901234567890123456789012345678901234
    45 |        1 | \nFROM stl_querytext;
```

Para reconstruir o SQL armazenado em STL_QUERYTEXT, execute o seguinte SQL.

```
select LISTAGG(CASE WHEN LEN(RTRIM(text)) = 0 THEN text ELSE RTRIM(text) END, '')
 within group (order by sequence) AS text
from stl_querytext where query=pg_last_query_id();
```

Para usar o SQL reconstruído resultante em seu cliente, substitua os caracteres especiais por uma nova linha (\n).

```
                    text

----------------------------------------------------------------------------------------------------
 select\n1 AS a01234567890123456789012345678901234567890123456789012345678901234567890,
\n2 AS b01234567890123456789012345678901234567890123456789012345678901234567890,\n3 AS
 b012345678901234567890123456789012345678901234\nFROM stl_querytext;
```

STL_REPLACEMENTS

Exibe um log que registra quando caracteres UTF-8 inválidos são substituídos pelo comando COPY com a opção ACCEPTINVCHARS. Uma entrada de log é adicionada à tabela STL_REPLACEMENTS para cada uma das 100 primeiras linhas em cada fatia de nó que exigiu pelo menos uma substituição.

STL_REPLACEMENTS permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_NESTLOOP contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_COPY_REPLACEMENTS. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
slice	integer	O número da fatia do nó onde a substituição ocorreu.
tbl	integer	ID da tabela.
starttime	timestamp	O horário de início (em UTC) do comando COPY.
sessão	integer	O ID de sessão para a sessão executando o comando COPY.
filename	character (256)	O caminho completo do arquivo de entrada para o comando COPY.
line_number	bigint	O número da linha no arquivo de dados de entrada que continha um caractere UTF-8 inválido. Um -1 indica que o número da linha não está disponível (como) ao copiar de um arquivo de dados colunares.
colname	character (127)	O primeiro campo que continha um caractere UTF-8 inválido.
raw_line	character (1024)	Os dados brutos de carga que continham um caractere UTF-8 inválido.
user_query_id	bigint	O identificador da consulta enviada pelo usuário, conforme registrado na coluna query_id de SYS_COPY_REPLACEMENTS.

Consultas de exemplo

O exemplo a seguir retorna as substituições da operação COPY mais recente.

```
select query, session, filename, line_number, colname
from stl_replacements
where query = pg_last_copy_id();

 query | session |   filename                                  | line_number | colname
 ------+---------+---------------------------------------------+-------------+--------
    96 |    6314 | s3://DOC-EXAMPLE-BUCKET/allusers_pipe.txt   |         251 | city
    96 |    6314 | s3://DOC-EXAMPLE-BUCKET/allusers_pipe.txt   |         317 | city
    96 |    6314 | s3://DOC-EXAMPLE-BUCKET/allusers_pipe.txt   |         569 | city
    96 |    6314 | s3://DOC-EXAMPLE-BUCKET/allusers_pipe.txt   |         623 | city
    96 |    6314 | s3://DOC-EXAMPLE-BUCKET/allusers_pipe.txt   |         694 | city
...
```

STL_RESTARTED_SESSIONS

Para manter a disponibilidade contínua após certos eventos internos, o Amazon Redshift pode reiniciar uma sessão ativa com um novo ID de processo (PID). Quando o Amazon Redshift reinicia uma sessão, a tabela STL_RESTARTED_SESSIONS registra o PID novo e o antigo.

Para obter mais informações, consulte os exemplos a seguir nesta seção.

STL_RESTARTED_SESSIONS permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_SESSION_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
currenttime	timestamp	O horário do evento.
dbname	character (50)	O nome do banco de dados associado à sessão.

Nome da coluna	Tipo de dados	Descrição
newpid	integer	O ID de processo da sessão reiniciada.
oldpid	integer	O ID de processo da sessão original.
username	character (50)	O nome do usuário associado à sessão.
remotehost	character (45)	O nome ou endereço IP do host remoto.
remoteport	character (32)	O número da porta do host remoto.
parkedtime	timestamp	Essas informações são somente para uso interno.
session_vars	character (2000)	Essas informações são somente para uso interno.

Consultas de exemplo

O exemplo a seguir une as tabelas STL_RESTARTED_SESSIONS e STL_SESSIONS para mostrar os nomes dos usuários das sessões que foram reiniciadas.

```
select process, stl_restarted_sessions.newpid, user_name
from stl_sessions
inner join stl_restarted_sessions on stl_sessions.process =
 stl_restarted_sessions.oldpid
order by process;

...
```

STL_RETURN

Contém detalhes das etapas de retorno nas consultas. Uma etapa de retorno obtém os resultados das consultas concluídas nos nós de computação para o nó líder. O nó de liderança, em seguida, mescla os dados e retorna os resultados ao cliente que fez a solicitação. Para as consultas concluídas em um nó líder, a etapa de retorno obtém os resultados para o cliente.

Uma consulta consiste em vários segmentos e cada segmento consiste em uma ou mais etapas. Para obter mais informações, consulte Processamento de consulta.

O STL_RETURNS é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_RETURN contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
slice	integer	O número que identifica a fatia em que a consulta estava sendo executada.
segment	integer	O número que identifica o segmento da consulta.
etapa	integer	Etapa da consulta que foi executada.
starttime	timestamp	O horário (em UTC) de início da consulta. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .

Nome da coluna	Tipo de dados	Descrição
endtime	timestamp	Horário em UTC em que a consulta foi finalizada. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
tasknum	integer	Número do processo de tarefa de consulta que foi atribuído para executar a etapa.
rows	bigint	O número total de linhas que foram processadas.
bytes	bigint	O tamanho, em bytes, de todas as linhas de saída da etapa.
packets	integer	O número total de pacotes enviados pela rede.
soma de verificação	bigint	Essas informações são somente para uso interno.

Consultas de exemplo

A consulta a seguir mostra quais etapas da consulta mais recente foram realizadas em cada fatia.

```
SELECT query, slice, segment, step, endtime, rows, packets
from stl_return where query = pg_last_query_id();

 query |  slice | segment | step |          endtime           | rows | packets
-------+--------+---------+------+----------------------------+------+---------
     4 |      2 |       3 |    2 | 2013-12-27 01:43:21.469043 |    3 |       0
     4 |      3 |       3 |    2 | 2013-12-27 01:43:21.473321 |    0 |       0
     4 |      0 |       3 |    2 | 2013-12-27 01:43:21.469118 |    2 |       0
     4 |      1 |       3 |    2 | 2013-12-27 01:43:21.474196 |    0 |       0
     4 |      4 |       3 |    2 | 2013-12-27 01:43:21.47704  |    2 |       0
     4 |      5 |       3 |    2 | 2013-12-27 01:43:21.478593 |    0 |       0
     4 |   12811|       4 |    1 | 2013-12-27 01:43:21.480755 |    0 |       0
(7 rows)
```

STL_S3CLIENT

Registra o tempo de transferência e outras métricas de performance.

Use a tabela STL_S3CLIENT para encontrar o tempo gasto na transferência de dados do Amazon S3.

STL_S3CLIENT permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
slice	integer	O número que identifica a fatia em que a consulta estava sendo executada.
recordtime	timestamp	O horário em que o registro é feito.
pid	integer	ID do processo. Todas as consultas em uma sessão são executadas no mesmo processo, portanto esse valor permanece constante se você executa uma série de consultas na mesma sessão.
http_method	character (64)	Nome do método HTTP correspondente à solicitação do Amazon S3.
bucket	character (64)	O nome do bucket do S3.
key	character (256)	A chave que corresponde ao objeto do Amazon S3.

Nome da coluna	Tipo de dados	Descrição
transfer_size	bigint	O número de bytes transferidos.
data_size	bigint	O número de bytes de dados. Este valor é o mesmo que o transfer_size para dados descompactados. Se a compactação foi usada, este é o tamanho dos dados descompactados.
start_time	bigint	O horário em que a transferência começou (em microsseg undos desde 1° de janeiro de 2000).
end_time	bigint	O horário em que a transferência terminou (em microsseg undos desde 1° de janeiro de 2000).
transfer_time	bigint	O tempo de duração da transferência (em microssegundos).
compression_time	bigint	A porção do tempo de transferência que foi gasta na descompactação dos dados (em microssegundos).
connect_time	bigint	O tempo decorrido desde o início até o encerramento da conexão com o servidor remoto (em microssegundos).
app_connect_time	bigint	O tempo decorrido desde o início até o encerramento da conexão SSL ou do handshake com o host remoto (em microssegundos).
retries	bigint	O número de novas tentativas de transferência.
request_id	char(32)	ID de solicitação da resposta do cabeçalho HTTP do Amazon S3
extended_ request_id	char(128)	ID de solicitação estendida da resposta do cabeçalho HTTP do Amazon S3 (x-amz-id-2).
ip_address	char(64)	O endereço IP do servidor (ip V4 ou V6).
is_parcial	integer	Valor que, se true (1), indica que o arquivo de entrada é dividido em intervalos durante uma operação COPY. Se esse valor for false (0), o arquivo de entrada não será dividido.

Nome da coluna	Tipo de dados	Descrição
start_offset	bigint	Valor que, se o arquivo de entrada for dividido durante uma operação COPY, indica o valor de deslocamento da divisão (em bytes). Se o arquivo não estiver dividido, esse valor será 0.
query_uuid	character (36)	O identificador globalmente exclusivo (UUID) da consulta.

Consulta de exemplo

A consulta a seguir retorna o tempo gasto para carregar arquivos usando o comando COPY.

```
select slice, key, transfer_time
from stl_s3client
where query = pg_last_copy_id();
```

Resultado

```
 slice |   key                        | transfer_time
 ------+------------------------------+---------------
     0 | listing10M0003_part_00       |    16626716
     1 | listing10M0001_part_00       |    12894494
     2 | listing10M0002_part_00       |    14320978
     3 | listing10M0000_part_00       |    11293439
  3371 | prefix=listing10M;marker=    |       99395
```

A consulta a seguir converte `start_time` e `end_time` em um carimbo de data e hora.

```
select userid,query,slice,pid,recordtime,start_time,end_time,
'2000-01-01'::timestamp + (start_time/1000000.0)* interval '1 second' as start_ts,
'2000-01-01'::timestamp + (end_time/1000000.0)* interval '1 second' as end_ts
from stl_s3client where query> -1 limit 5;
```

```
 userid | query | slice |  pid  |         recordtime         |   start_time    |
 end_time     |          start_ts          |          end_ts
```

```
--------+-------+-------+-------+---------------------------+----------------
+-----------------+---------------------------+---------------------------
      0 |     0 |     0 | 23449 | 2019-07-14 16:27:17.207839 | 616436837154256 |
 616436837207838 | 2019-07-14 16:27:17.154256 | 2019-07-14 16:27:17.207838
      0 |     0 |     0 | 23449 | 2019-07-14 16:27:17.252521 | 616436837208208 |
 616436837252520 | 2019-07-14 16:27:17.208208 | 2019-07-14 16:27:17.25252
      0 |     0 |     0 | 23449 | 2019-07-14 16:27:17.284376 | 616436837208460 |
 616436837284374 | 2019-07-14 16:27:17.20846  | 2019-07-14 16:27:17.284374
      0 |     0 |     0 | 23449 | 2019-07-14 16:27:17.285307 | 616436837208980 |
 616436837285306 | 2019-07-14 16:27:17.20898  | 2019-07-14 16:27:17.285306
      0 |     0 |     0 | 23449 | 2019-07-14 16:27:17.353853 | 616436837302216 |
 616436837353851 | 2019-07-14 16:27:17.302216 | 2019-07-14 16:27:17.353851
```

STL_S3CLIENT_ERROR

Registra erros encontrados por uma fatia ao carregar um arquivo do Amazon S3.

Use STL_S3CLIENT_ERROR para encontrar detalhes para erros encontrados durante a transferência de dados do Amazon S3 como parte de um comando COPY.

STL_S3CLIENT_ERROR permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema. O ID da consulta -1 é para uso interno.
sliceid	integer	O número que identifica a fatia em que a consulta estava sendo executada.
recordtime	timestamp	O horário em que o registro é feito.

Nome da coluna	Tipo de dados	Descrição
pid	integer	ID do processo. Todas as consultas em uma sessão são executadas no mesmo processo, portanto esse valor permanece constante se você executa uma série de consultas na mesma sessão.
http_method	character (64)	Nome do método HTTP correspondente à solicitação do Amazon S3.
bucket	character (64)	Nome do bucket do Amazon S3.
key	character (256)	A chave que corresponde ao objeto do Amazon S3.
erro	character (1024)	A mensagem de erro.
is_parcial	integer	Valor que, se true (1), indica que o arquivo de entrada é dividido em intervalos durante uma operação COPY. Se esse valor for false (0), o arquivo de entrada não será dividido.
start_offset	bigint	Valor que, se o arquivo de entrada for dividido durante uma operação COPY, indica o valor de deslocamento da divisão (em bytes). Se o arquivo não estiver dividido, esse valor será 0.

Observações de uso

Se você encontrar vários erros com "Connection timed out", pode haver um problema de rede. Se você estiver usando o Roteamento por VPC aprimorado, verifique se existe um caminho de rede válido entre a VPC do cluster e seus recursos de dados. Para obter mais informações, consulte Roteamento aprimorado de VPC do Amazon Redshift Spectrum.

Consulta de exemplo

A consulta a seguir retorna os erros dos comandos COPY concluídos durante a sessão atual.

```
select query, sliceid, substring(key from 1 for 20) as file,
substring(error from 1 for 35) as error
from stl_s3client_error
where pid = pg_backend_pid()
order by query desc;
```

Resultado

```
 query  | sliceid |     file           |              error

--------+---------+--------------------+-----------------------------------
 362228 |      12 | part.tbl.25.159.gz | transfer closed with 1947655 bytes
 362228 |      24 | part.tbl.15.577.gz | transfer closed with 1881910 bytes
 362228 |       7 | part.tbl.22.600.gz | transfer closed with 700143 bytes r
 362228 |      22 | part.tbl.3.34.gz   | transfer closed with 2334528 bytes
 362228 |      11 | part.tbl.30.274.gz | transfer closed with 699031 bytes r
 362228 |      30 | part.tbl.5.509.gz  | Unknown SSL protocol error in conne
 361999 |      10 | part.tbl.23.305.gz | transfer closed with 698959 bytes r
 361999 |      19 | part.tbl.26.582.gz | transfer closed with 1881458 bytes
 361999 |       4 | part.tbl.15.629.gz | transfer closed with 2275907 bytes
 361999 |      20 | part.tbl.6.456.gz  | transfer closed with 692162 bytes r
(10 rows)
```

STL_SAVE

Contém os detalhes das etapas de gravação nas consultas. Uma etapa de gravação salva o stream de entrada em uma tabela transitória. Uma tabela transitória é uma tabela temporária que armazena os resultados intermediários durante a execução da consulta.

Uma consulta consiste em vários segmentos e cada segmento consiste em uma ou mais etapas. Para obter mais informações, consulte Processamento de consulta.

STL_SAVE é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_SAVE contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em

namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
slice	integer	O número que identifica a fatia em que a consulta estava sendo executada.
segment	integer	O número que identifica o segmento da consulta.
etapa	integer	Etapa da consulta que foi executada.
starttime	timestamp	O horário (em UTC) de início da consulta. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Horário em UTC em que a consulta foi finalizada. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
tasknum	integer	Número do processo de tarefa de consulta que foi atribuído para executar a etapa.
rows	bigint	O número total de linhas que foram processadas.
bytes	bigint	O tamanho, em bytes, de todas as linhas de saída da etapa.

Nome da coluna	Tipo de dados	Descrição
tbl	integer	O ID da tabela transitória materializada.
is_diskbased	character(1)	Indica se esta etapa da consulta foi realizada como uma operação em disco: true (`t`) ou false (`f`).
workmem	bigint	O número de bytes da memória de trabalho atribuída à etapa.

Consultas de exemplo

A consulta a seguir mostra quais etapas de salvamento da consulta mais recente foram realizadas em cada fatia.

```
select query, slice, segment, step, tasknum, rows,  tbl
from stl_save where query = pg_last_query_id();

 query | slice | segment | step | tasknum | rows | tbl
-------+-------+---------+------+---------+------+-----
 52236 |     3 |       0 |    2 |      21 |    0 | 239
 52236 |     2 |       0 |    2 |      20 |    0 | 239
 52236 |     2 |       2 |    2 |      20 |    0 | 239
 52236 |     3 |       2 |    2 |      21 |    0 | 239
 52236 |     1 |       0 |    2 |      21 |    0 | 239
 52236 |     0 |       0 |    2 |      20 |    0 | 239
 52236 |     0 |       2 |    2 |      20 |    0 | 239
 52236 |     1 |       2 |    2 |      21 |    0 | 239
(8 rows)
```

STL_SCAN

Analisa as etapas de varredura de tabelas nas consultas. O número da etapa para as linhas dessa tabela é sempre 0, pois a varredura é a primeira etapa em um segmento.

STL_SCAN permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_SCAN contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
slice	integer	O número que identifica a fatia em que a consulta estava sendo executada.
segment	integer	O número que identifica o segmento da consulta.
etapa	integer	Etapa da consulta que foi executada.
starttime	timestamp	O horário (em UTC) de início da consulta. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Horário em UTC em que a consulta foi finalizada. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .

Nome da coluna	Tipo de dados	Descrição
tasknum	integer	Número do processo de tarefa de consulta que foi atribuído para executar a etapa.
rows	bigint	O número total de linhas que foram processadas.
bytes	bigint	O tamanho, em bytes, de todas as linhas de saída da etapa.
fetches	bigint	Essas informações são somente para uso interno.
type	integer	O ID do tipo de varredura. Para ver uma lista de valores válidos, consulte tabela a seguir.
tbl	integer	ID da tabela.
is_rrscan	character(1)	O valor true (t) indica que a varredura restrita ao intervalo foi utilizada na etapa.
is_delayed_scan	character(1)	Essas informações são somente para uso interno.
rows_pre_filter	bigint	Para varreduras de tabelas permanentes, o número total de linhas enviadas antes da filtragem das linhas marcadas para exclusão (linhas fantasma) e antes de aplicar os filtros de consulta definidos pelo usuário.
rows_pre_user_filter	bigint	Para varreduras de tabelas permanentes, o número total de linhas processadas após a filtragem das linhas marcadas para exclusão (linhas fantasma), mas antes de aplicar os filtros de consulta definidos pelo usuário.
perm_table_name	character(136)	Para varreduras de tabelas permanentes, o nome da tabela pesquisada na varredura.
is_rlf_scan	character(1)	O valor true (t) indica que o filtro de baixo nível foi utilizado na etapa.

Nome da coluna	Tipo de dados	Descrição
is_rlf_scan_reason	integer	Essas informações são somente para uso interno.
num_em_blocks	integer	Essas informações são somente para uso interno.
soma de verificação	bigint	Essas informações são somente para uso interno.
runtime_filtering	character(1)	Se true (t), indica que os filtros do tempo de execução são aplicados.
scan_region	integer	Essas informações são somente para uso interno.
num_sortkey_as_predicate	integer	Essas informações são somente para uso interno.
row_fetcher_state	integer	Essas informações são somente para uso interno.
consumed_scan_ranges	bigint	Essas informações são somente para uso interno.
work_stealing_reason	bigint	Essas informações são somente para uso interno.
is_vectorized_scan	character(1)	Essas informações são somente para uso interno.
is_vectorized_scan_reason	integer	Essas informações são somente para uso interno.

Nome da coluna	Tipo de dados	Descrição
row_fetcher_reason	bigint	Essas informações são somente para uso interno.
topology_signature	bigint	Essas informações são somente para uso interno.
use_tpm_partição	character(1)	Essas informações são somente para uso interno.
is_rrscan_expr	character(1)	Essas informações são somente para uso interno.
scanned_mega_value	character(1)	Essas informações são somente para uso interno. Essas informações mostram se a etapa de varredura fornecida digitalizou um valor grande. Um valor grande será armazenado em vários blocos. O tamanho do bloco é de 1 MB por padrão; um valor grande é maior que 1 MB em uma configuração padrão.

Tipos de varredura

ID do tipo	Descrição
1	Os dados da rede.
2	As tabelas de usuário permanentes em memória compartilhada compactada.
3	As tabelas transitórias lineares.
21	Carregar arquivos do Amazon S3.
22	Carregar tabelas do Amazon DynamoDB.
23	Os dados de carga de uma conexão SSH remota.

ID do tipo	Descrição
24	Os dados de carga do cluster remoto (região classificada). Esse tipo é usado para redimensionamento.
25	Os dados de carga do cluster remoto (região não classificada). Esse tipo é usado para redimensionamento.
28	Leia dados de uma exibição de série temporal com UNION ALL em várias tabelas.
29	Leia dados de tabelas externas do Amazon S3.
30	Leia as informações de partição de uma tabela externa do Amazon S3.
33	Leia dados de uma tabela remota do Postgres.
36	Leia dados de uma tabela remota do MySQL.
37	Leia dados de um fluxo remoto do Kinesis.

Observações de uso

Idealmente, o número de `rows` deve estar relativamente perto do número de `rows_pre_filter`. Uma diferença grande entre `rows` e `rows_pre_filter` indica que o mecanismo de execução está pesquisando linhas que serão descartadas mais tarde, o que é ineficiente. A diferença entre `rows_pre_filter` e `rows_pre_user_filter` é o número de linhas fantasma na varredura. Execute o comando VACUUM para remover as linhas marcadas para exclusão. A diferença entre `rows` e `rows_pre_user_filter` é o número de linhas filtradas pela consulta. Se muitas linhas estiverem sendo rejeitadas pelo filtro do usuário, reveja a escolha de coluna de classificação ou, se isso for devido a uma região extensa não classificada, execute uma limpeza.

Consultas de exemplo

O exemplo a seguir mostra que `rows_pre_filter` é maior do que `rows_pre_user_filter`, pois a tabela tem linhas excluídas que não foram limpadas (linhas fantasma).

```
SELECT query, slice, segment,step,rows, rows_pre_filter, rows_pre_user_filter
from stl_scan where query = pg_last_query_id();
```

```
 query |  slice | segment | step | rows  | rows_pre_filter | rows_pre_user_filter
-------+--------+---------+------+-------+-----------------+---------------------
 42915 |      0 |       0 |    0 | 43159 |           86318 |                43159
 42915 |      0 |       1 |    0 |     1 |               0 |                    0
 42915 |      1 |       0 |    0 | 43091 |           86182 |                43091
 42915 |      1 |       1 |    0 |     1 |               0 |                    0
 42915 |      2 |       0 |    0 | 42778 |           85556 |                42778
 42915 |      2 |       1 |    0 |     1 |               0 |                    0
 42915 |      3 |       0 |    0 | 43428 |           86856 |                43428
 42915 |      3 |       1 |    0 |     1 |               0 |                    0
 42915 |  10000 |       2 |    0 |     4 |               0 |                    0
(9 rows)
```

STL_SCHEMA_QUOTA_VIOLATIONS

Registra a ocorrência, o time stamp, o XID e outras informações úteis quando uma cota de esquema é excedida.

STL_SCHEMA_QUOTA_VIOLATIONS permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_SCHEMA_QUOTA_VIOLATIONS. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
ownerid	integer	O ID do proprietário do esquema.
xid	bigint	O ID da transação associada à instrução.
pid	integer	O ID do processo associado à instrução.
userid	integer	O ID do usuário que gerou a entrada.

Nome da coluna	Tipo de dados	Descrição
schema_id	integer	O ID do esquema ou namespace.
schema_name	character (128)	O nome do esquema ou namespace.
quota	integer	A quantidade de espaço em disco (em MB) que o esquema pode usar.
disk_usage	integer	O espaço em disco (em MB) atualmente usado pelo esquema.
disk_usage_pct	double precision	A porcentagem de espaço em disco usado atualmente pelo esquema fora da cota configurada.
timestamp	time stamp sem fuso horário	O horário de ocorrência da violação.

Consultas de exemplo

A consulta a seguir mostra o resultado da violação de cota:

```
SELECT userid, TRIM(SCHEMA_NAME) "schema_name", quota, disk_usage, disk_usage_pct,
 timestamp FROM
stl_schema_quota_violations WHERE SCHEMA_NAME = 'sales_schema' ORDER BY timestamp DESC;
```

Essa consulta retorna o seguinte exemplo de resultado para o esquema especificado:

```
userid | schema_name  | quota | disk_usage | disk_usage_pct |timestamp
-------+--------------+-------+------------+----------------
+----------------------------
104    | sales_schema | 2048  | 2798       |  136.62        | 2020-04-20
 20:09:25.494723
(1 row)
```

STL_SESSIONS

Retorna as informações sobre o histórico da sessão do usuário.

A diferença entre a STL_SESSIONS e a STV_SESSIONS é que a STL_SESSIONS contém o histórico das sessões enquanto a STV_SESSIONS contém as sessões atuais ativas.

STL_SESSIONS permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_SESSION_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
starttime	timestamp	O horário (em UTC) do início da sessão.
endtime	timestamp	O horário (em UTC) do término da sessão.
process	integer	O ID de processo da sessão.
user_name	character(50)	O nome do usuário associado à sessão.
db_name	character(50)	O nome do banco de dados associado à sessão.
timeout_sec	int	O tempo máximo em segundos em que uma sessão permanece inativa ou ociosa antes do tempo limite. 0 indica que nenhum tempo limite está definido.
timed_out	int	Um valor que indica por que a conexão foi encerrada. Ele pode ter os seguintes valores:

Nome da coluna	Tipo de dados	Descrição
		• 0: a conexão foi encerrada devido a um erro desconhecido. • 1: a conexão atingiu o tempo limite. • 2: o lado do cliente encerrou a conexão. • 3: um erro interno do back-end do Amazon Redshift encerrou a conexão.

Consultas de exemplo

Para visualizar o histórico da sessão do banco de dados TICKIT, digite a seguinte consulta:

```
select starttime, process, user_name, timeout_sec, timed_out
from stl_sessions
where db_name='tickit' order by starttime;
```

Essa consulta retorna os seguintes dados de saída de exemplo:

```
    starttime              | process |  user_name             | timeout_sec | timed_out
---------------------------+---------+------------------------+-------------
+-------------
2008-09-15 09:54:06.746705 |   32358 | dwuser                 | 120         | 1
2008-09-15 09:56:34.30275  |   32744 | dwuser                 | 60          | 1
2008-09-15 11:20:34.694837 |   14906 | dwuser                 | 0           | 0
2008-09-15 11:22:16.749818 |   15148 | dwuser                 | 0           | 0
2008-09-15 14:32:44.66112  |   14031 | dwuser                 | 0           | 0
2008-09-15 14:56:30.22161  |   18380 | dwuser                 | 0           | 0
2008-09-15 15:28:32.509354 |   24344 | dwuser                 | 0           | 0
2008-09-15 16:01:00.557326 |   30153 | dwuser                 | 120         | 1
2008-09-15 17:28:21.419858 |   12805 | dwuser                 | 0           | 0
2008-09-15 20:58:37.601937 |   14951 | dwuser                 | 60          | 1
2008-09-16 11:12:30.960564 |   27437 | dwuser                 | 60          | 1
2008-09-16 14:11:37.639092 |   23790 | dwuser                 | 3600        | 1
2008-09-16 15:13:46.02195  |    1355 | dwuser                 | 120         | 1
2008-09-16 15:22:36.515106 |    2878 | dwuser                 | 120         | 1
2008-09-16 15:44:39.194579 |    6470 | dwuser                 | 120         | 1
2008-09-16 16:50:27.02138  |   17254 | dwuser                 | 120         | 1
```

```
2008-09-17 12:05:02.157208 |    8439 | dwuser                | 3600        | 0
(17 rows)
```

STL_SORT

Exibe as etapas de execução da classificação nas consultas, como as etapas que usam o processamento de ORDER BY.

STL_SORT permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_SORT contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
slice	integer	O número que identifica a fatia em que a consulta estava sendo executada.
segment	integer	O número que identifica o segmento da consulta.
etapa	integer	Etapa da consulta que foi executada.

Nome da coluna	Tipo de dados	Descrição
starttime	timestamp	O horário (em UTC) de início da consulta. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **2009-06-12 11:29:19.131358** .
endtime	timestamp	Horário em UTC em que a consulta foi finalizada. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **2009-06-12 11:29:19.131358** .
tasknum	integer	Número do processo de tarefa de consulta que foi atribuído para executar a etapa.
rows	bigint	O número total de linhas que foram processadas.
bytes	bigint	O tamanho, em bytes, de todas as linhas de saída da etapa.
tbl	integer	ID da tabela.
is_diskbased	character(1)	Se o valor é true (t), a consulta foi realizada como uma operação em disco. Se o valor é false (f), a consulta foi realizada na memória.
workmem	bigint	O número total de bytes na memória de trabalho que foram atribuídos à etapa.
soma de verificação	bigint	Essas informações são somente para uso interno.

Consultas de exemplo

O exemplo a seguir retorna os resultados da classificação para a fatia 0 e o segmento 1.

```
select query, bytes, tbl, is_diskbased, workmem
from stl_sort
```

```
where slice=0 and segment=1;
```

```
 query |  bytes  | tbl | is_diskbased |  workmem
-------+---------+-----+--------------+-----------
   567 | 3126968 | 241 | f            | 383385600
   604 |    5292 | 242 | f            | 383385600
   675 |  104776 | 251 | f            | 383385600
   525 | 3126968 | 251 | f            | 383385600
   585 |    5068 | 241 | f            | 383385600
   630 |  204808 | 266 | f            | 383385600
   704 |       0 | 242 | f            |         0
   669 | 4606416 | 241 | f            | 383385600
   696 |  104776 | 241 | f            | 383385600
   651 | 4606416 | 254 | f            | 383385600
   632 |       0 | 256 | f            |         0
   599 |     396 | 241 | f            | 383385600
 86397 |       0 | 242 | f            |         0
   621 |    5292 | 241 | f            | 383385600
 86325 |       0 | 242 | f            |         0
   572 |    5068 | 242 | f            | 383385600
   645 |  204808 | 241 | f            | 383385600
   590 |     396 | 242 | f            | 383385600
(18 rows)
```

STL_SSHCLIENT_ERROR

Registra todos os erros vistos pelo cliente SSH.

STL_SSHCLIENT_ERROR permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.

Nome da coluna	Tipo de dados	Descrição
slice	integer	O número que identifica a fatia em que a consulta estava sendo executada.
recordtime	timestamp	O horário em que o erro foi registrado.
pid	integer	O processo que registrou o erro.
ssh_username	character (1024)	O nome do usuário do SSH.
endpoint	character (1024)	O endpoint do SSH.
command	character (4096)	O comando SSH completo.
erro	character (1024)	A mensagem de erro.

STL_STREAM_SEGS

Lista a relação entre os streams e os segmentos simultâneos.

Neste contexto, os fluxos são do Amazon Redshift. Essa visão do sistema não se aplica a Ingestão de streaming para uma visão materializada.

STL_SSHCLIENT_ERROR é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_STREAM_SEGS contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e

namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
transmissão	integer	O conjunto de segmentos simultâneos de uma consulta.
segment	integer	O número que identifica o segmento da consulta.
query_uuid	character (36)	O identificador globalmente exclusivo (UUID) da consulta.

Consultas de exemplo

Para visualizar a relação entre streams e segmentos simultâneos para a consulta mais recente, digite a seguinte consulta:

```
select *
from stl_stream_segs
where query = pg_last_query_id();

 query | stream | segment
-------+--------+---------
    10 |      1 |       2
    10 |      0 |       0
    10 |      2 |       4
    10 |      1 |       3
    10 |      0 |       1
```

```
(5 rows)
```

STL_TR_CONFLICT

Exibe informações para identificar e resolver conflitos de transações com tabelas de banco de dados.

Um conflito de transações ocorre quando dois ou mais usuários estão executando consultas e alterando linhas de dados de tabelas, de modo que suas transações não podem ser serializadas. A transação que executa uma instrução que poderia quebrar a serialização é interrompida e revertida. Sempre que ocorre um conflito de transação, o Amazon Redshift grava uma linha de dados na tabela de sistema STL_TR_CONFLICT contendo detalhes sobre a transação cancelada. Para obter mais informações, consulte Níveis de isolamento no Amazon Redshift.

STL_TR_CONFLICT só permanece visível para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_TRANSACTION_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
xact_id	bigint	O ID de transação para a transação revertida.
process_id	bigint	O processo associado à transação que foi revertida.
xact_start_ts	timestamp	O carimbo de data/hora (UTC) em que a transação começou.
abort_time	timestamp	O carimbo de data/hora (UTC) em que a transação foi interrompida.
table_id	bigint	O ID de tabela para a tabela onde o conflito ocorreu.

Consulta de exemplo

Para obter informações sobre os conflitos que envolvem uma tabela específica, execute uma consulta especificando o ID da tabela:

```
select * from stl_tr_conflict where table_id=100234
order by xact_start_ts;

xact_id|process_|      xact_start_ts       |        abort_time        |table_
       |id      |                          |                          |id
-------+--------+--------------------------+--------------------------+------
  1876 |  8551  |2010-03-30 09:19:15.852326|2010-03-30 09:20:17.582499|100234
  1928 | 15034  |2010-03-30 13:20:00.636045|2010-03-30 13:20:47.766817|100234
  1991 | 23753  |2010-04-01 13:05:01.220059|2010-04-01 13:06:06.94098 |100234
  2002 | 23679  |2010-04-01 13:17:05.173473|2010-04-01 13:18:27.898655|100234
(4 rows)
```

Você pode obter o ID da tabela na seção DETAIL da mensagem de erro das violações de serialização (erro 1023).

STL_UNDONE

Exibe informações sobre as transações que foram desfeitas.

STL_UNDONE permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_TRANSACTION_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.

Nome da coluna	Tipo de dados	Descrição
xact_id	bigint	O ID da transação de desfazer.
xact_id_undone	bigint	ID da transação que foi desfeita.
undo_start_ts	timestamp	O horário de início da transação de desfazer.
undo_end_ts	timestamp	O horário de término da transação de desfazer.
table_id	bigint	O ID da tabela que foi afetada pela transação de desfazer.

Consulta de exemplo

Para visualizar um log conciso de todas as transações de desfazer, digite o seguinte comando:

```
select xact_id, xact_id_undone, table_id from stl_undone;
```

Este comando retorna a seguinte saída de exemplo:

```
 xact_id | xact_id_undone | table_id
---------+----------------+----------
1344 |           1344 |   100192
1326 |           1326 |   100192
1551 |           1551 |   100192
(3 rows)
```

STL_UNIQUE

Analisa as etapas de execução que ocorrem quando uma função DISTINCT é usada na lista SELECT, ou quando duplicações são removidas de uma consulta UNION ou INTERSECT.

STL_UNIQUE permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_UNIQUE contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
slice	integer	O número que identifica a fatia em que a consulta estava sendo executada.
segment	integer	O número que identifica o segmento da consulta.
etapa	integer	Etapa da consulta que foi executada.
starttime	timestamp	O horário (em UTC) de início da consulta. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Horário em UTC em que a consulta foi finalizada. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .

Nome da coluna	Tipo de dados	Descrição
tasknum	integer	Número do processo de tarefa de consulta que foi atribuído para executar a etapa.
rows	bigint	O número total de linhas que foram processadas.
type	character(6)	O tipo da etapa. Os valores válidos são: • HASHED. Indica que a etapa usou uma agregação não classificada e agrupada. • PLAIN. Indica que a etapa usou uma agregação escalar e não agrupada. • SORTED. Indica que a etapa usou uma agregação classificada e agrupada.
is_diskbased	character(1)	Se o valor é true (t), a consulta foi realizada como uma operação em disco. Se o valor é false (f), a consulta foi realizada na memória.
slots	integer	O número total de buckets de hash.
workmem	bigint	O número total de bytes na memória de trabalho que foram atribuídos à etapa.
max_buffers_used	bigint	O número máximo de buffers usados na tabela de hash antes de acessar o disco.
resizes	integer	Essas informações são somente para uso interno.
occupied	integer	Essas informações são somente para uso interno.
flushable	integer	Essas informações são somente para uso interno.
used_unique_prefetching	character(1)	Essas informações são somente para uso interno.

Nome da coluna	Tipo de dados	Descrição
bytes	biginit	O número de bytes de todas as linhas de saída da etapa.

Consultas de exemplo

Suponha que você execute a seguinte consulta:

```
select distinct eventname
from event order by 1;
```

Assumindo que o ID da consulta anterior seja 6313, o exemplo a seguir mostra o número de linhas produzidas pela etapa exclusiva para cada fatia nos segmentos 0 e 1.

```
select query, slice, segment, step, datediff(msec, starttime, endtime) as msec,
 tasknum, rows
from stl_unique where query = 6313
order by query desc, slice, segment, step;
```

```
 query | slice | segment | step | msec | tasknum | rows
-------+-------+---------+------+------+---------+------
  6313 |     0 |       0 |    2 |    0 |      22 |  550
  6313 |     0 |       1 |    1 |  256 |      20 |  145
  6313 |     1 |       0 |    2 |    1 |      23 |  540
  6313 |     1 |       1 |    1 |   42 |      21 |  127
  6313 |     2 |       0 |    2 |    1 |      22 |  540
  6313 |     2 |       1 |    1 |  255 |      20 |  158
  6313 |     3 |       0 |    2 |    1 |      23 |  542
  6313 |     3 |       1 |    1 |   38 |      21 |  146
(8 rows)
```

STL_UNLOAD_LOG

Registra os detalhes de uma operação de descarregamento.

A tabela STL_UNLOAD_LOG registra uma linha para cada arquivo criado por uma instrução UNLOAD. Por exemplo, se um UNLOAD criar 12 arquivos, a STL_UNLOAD_LOG conterá 12 linhas correspondentes.

STL_UNLOAD_LOG permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_UNLOAD_LOG contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar as visualizações de monitoramento SYS SYS_UNLOAD_HISTORY e SYS_UNLOAD_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	O ID da consulta.
slice	integer	O número que identifica a fatia em que a consulta estava sendo executada.
pid	integer	O ID do processo associado à instrução da consulta.
caminho	character(1280)	O caminho completo do objeto Amazon S3 para o arquivo.
start_time	timestamp	O horário de início da transação.
end_time	timestamp	O horário de término da transação.
line_count	bigint	O número de linhas descarregadas no arquivo.
transfer_size	bigint	O número de bytes transferidos.

Nome da coluna	Tipo de dados	Descrição
file_format	character(10)	Formato de arquivo não carregado.
user_query_id	bigint	O identificador da consulta enviada pelo usuário, conforme registrado na coluna query_id de SYS_UNLOAD_DETAIL.
transaction_id	bigint	O identificador da transação associada à instrução.

Consulta de exemplo

Para obter uma lista dos arquivos que foram gravados no Amazon S3 por um comando UNLOAD, você pode chamar uma operação de lista do Amazon S3 após a conclusão do UNLOAD. Você também pode consultar STL_UNLOAD_LOG.

A consulta a seguir retorna o nome do caminho para os arquivos que foram criados por um UNLOAD para a última consulta concluída:

```
select query, substring(path,0,40) as path
from stl_unload_log
where query = pg_last_query_id()
order by path;
```

Este comando retorna a seguinte saída de exemplo:

```
 query |             path
-------+--------------------------------------
  2320 | s3://amzn-s3-demo-bucket/venue0000_part_00
  2320 | s3://amzn-s3-demo-bucket/venue0001_part_00
  2320 | s3://amzn-s3-demo-bucket/venue0002_part_00
  2320 | s3://amzn-s3-demo-bucket/venue0003_part_00
(4 rows)
```

STL_USAGE_CONTROL

A visualização STL_USAGE_CONTROL contém informações que são registradas quando um limite de uso é atingido. Para obter mais informações sobre limites de uso, consulte “Gerenciar limites de uso” no Guia de gerenciamento de clusters do Amazon Redshift.

STL_USAGE_CONTROL é visível somente para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
eventtime	timestamp	A hora (UTC) em que a consulta excedeu um limite de uso.
consultar	integer	O identificador da consulta. Este ID pode ser usado para unir várias outras tabelas e exibições do sistema.
xid	bigint	O identificador da transação.
pid	integer	O identificador de processo associado à consulta.
usage_limit_id	character(40)	Um identificador universalmente exclusivo (UUID) gerado pelo Amazon Redshift, por exemplo `25d9297e-3e7b-41c8-9f4d-c4b6eb731c09` .
feature_type	character(30)	O recurso cujo limite de uso foi excedido. Os valores possíveis incluem `CONCURRENCY_SCALING` e `SPECTRUM`.

Consulta de exemplo

O exemplo de SQL a seguir retorna algumas das informações registradas em log quando um limite de uso é atingido.

```
select query, pid, eventtime, feature_type
from stl_usage_control
order by eventtime desc
```

```
limit 5;
```

STL_USERLOG

Registra os detalhes das seguintes alterações de um usuário de banco de dados:

- Criar usuário
- Descartar usuário
- Alterar usuário (renomear)
- Alterar usuário (alterar as propriedades)

STL_USERLOG só permanece visível para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_USERLOG. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário afetado pelas alterações.
username	character(50)	O nome de usuário do usuário afetado pelas alterações.
oldusername	character(50)	Para uma ação de renomeação, o nome de usuário original. Para qualquer outra ação, este campo é vazio.
ação	character(10)	A ação ocorrida. Valores válidos: • Alter • Criar • Drop • Renomear

Nome da coluna	Tipo de dados	Descrição
usecreatedb	integer	Se for verdadeiro (1), indica que o usuário tem privilégios para criar um banco de dados.
usesuper	integer	Se for verdadeiro (1), indica que o usuário é um superusuário.
usecatupd	integer	Se for verdadeiro (1), indica que o usuário pode atualizar catálogos do sistema.
valuntil	timestamp	A data de expiração da senha.
pid	integer	ID do processo.
xid	bigint	ID da transação.
recordtime	timestamp	O horário (em UTC) de início da consulta.

Consultas de exemplo

O exemplo a seguir executa quatro ações do usuário e, em seguida, consulta a visualização STL_USERLOG.

```
create user userlog1 password 'Userlog1';
alter user userlog1 createdb createuser;
alter user userlog1 rename  to userlog2;
drop user userlog2;

select userid, username, oldusername, action, usecreatedb, usesuper from stl_userlog
 order by recordtime desc;
```

```
 userid |  username | oldusername |  action | usecreatedb | usesuper
--------+-----------+-------------+---------+-------------+----------
    108 | userlog2  |             | drop    |           1 |   1
    108 | userlog2  | userlog1    | rename  |           1 |   1
```

```
   108 | userlog1  |             | alter   |           1 |   1
   108 | userlog1  |             | create  |           0 |   0
(4 rows)
```

STL_UTILITYTEXT

Captura o texto de comandos não SELECT SQL executados no banco de dados.

Consulte a visualização STL_UTILITYTEXT para capturar o seguinte subconjunto de instruções SQL que foram executadas no sistema:

- ABORT, BEGIN, COMMIT, END, ROLLBACK
- ANALYZE
- CALL
- CANCEL
- COMMENT
- CREATE, ALTER, DROP DATABASE
- CREATE, ALTER, DROP USER
- EXPLAIN
- GRANT, REVOKE
- LOCK
- RESET
- SET
- SHOW
- TRUNCATE

Consulte também STL_DDLTEXT, STL_QUERYTEXT e SVL_STATEMENTTEXT.

Use as colunas STARTTIME e ENDTIME para saber quais instruções foram registradas em um determinado período. Os blocos longos de texto SQL são quebrados em linhas de 200 caracteres; a coluna SEQUENCE identifica os fragmentos de texto que pertencem a uma única instrução.

A tabela de sistema STL_UTILITYTEXT permite as operações MANUAL e AUTO REFRESH nas visões materializadas. Para identificar AUTO REFRESH em visões materializadas, encontre a coluna `label`. Todas as consultas AUTO REFRESH têm um rótulo com o valor `maintenance`.

STL_UTILITYTEXT permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_QUERY_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
xid	bigint	ID da transação.
pid	integer	O ID do processo associado à instrução da consulta.
rótulo	character(320)	O nome do arquivo usado para executar a consulta ou um rótulo definido com o comando SET QUERY_GROUP. Se a consulta não for baseada em arquivos ou o parâmetro QUERY_GROUP não estiver definido, o valor deste campo será branco.
starttime	timestamp	O horário (em UTC) de início da consulta. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **2009-06-12 11:29:19.131358** .
endtime	timestamp	Horário em UTC em que a consulta foi finalizada. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **2009-06-12 11:29:19.131358** .
sequence	integer	Quando uma única instrução contém mais de 200 caracteres, são registradas linhas adicionais para essa

Nome da coluna	Tipo de dados	Descrição
		instrução. O valor 0 da sequência é a primeira linha, 1 é a segunda, e assim por diante.
texto	character(200)	O texto em SQL, em incrementos de 200 caracteres. Esse campo pode conter caracteres especiais como barra invertida (\\) e nova linha (\n).

Consultas de exemplo

A consulta a seguir retorna o texto dos comandos de “utilitário” que foram executados em 26 de janeiro de 2012. Nesse caso, alguns comandos SET e um SHOW ALL foram executados:

```
select starttime, sequence, rtrim(text)
from stl_utilitytext
where starttime like '2012-01-26%'
order by starttime, sequence;

starttime          | sequence |              rtrim
---------------------------+-----+---------------------------------
2012-01-26 13:05:52.529235 |   0 | show all;
2012-01-26 13:20:31.660255 |   0 | SET query_group to ''
2012-01-26 13:20:54.956131 |   0 | SET query_group to 'soldunsold.sql'
...
```

Reconstrução de SQL armazenado

Para reconstruir o SQL armazenado na coluna `text` de STL_UTILITYTEXT, execute uma instrução SELECT para criar SQL a partir de uma ou mais partes na coluna `text`. Antes de executar o SQL reconstruído, substitua os caracteres especiais por uma nova linha (\n). O resultado da instrução SELECT a seguir são linhas de SQL reconstruído no campo `query_statement`.

```
SELECT LISTAGG(CASE WHEN LEN(RTRIM(text)) = 0 THEN text ELSE RTRIM(text) END) WITHIN
 GROUP (ORDER BY sequence) as query_statement
FROM stl_utilitytext GROUP BY xid order by xid;
```

Por exemplo, a consulta a seguir define query_group como uma string de zeros. A consulta em si tem mais de 200 caracteres e é armazenada em partes em STL_UTILITYTEXT.

```
set query_group to
 '000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000
             000000';
```

Nesse exemplo, a consulta é armazenada em duas partes (linhas) na coluna `text` de STL_UTILITYTEXT.

```
select query, sequence, text
from stl_utilitytext where query=pg_last_query_id() order by query desc, sequence limit
 10;
```

```
         starttime          | sequence |
                                                text

----------------------------+----------
+-----------------------------------------------------------------------------------------------
 2019-07-23 22:55:34.926198 |        0 | set query_group to
 '000000000000000000000000000000000000000000000000000000000000000000000000000000000000\n00000000000
 2019-07-23 22:55:34.926198 |        1 |                   000000';
```

Para reconstruir o SQL armazenado em STL_UTILITYTEXT, execute o seguinte SQL.

```
select LISTAGG(CASE WHEN LEN(RTRIM(text)) = 0 THEN text ELSE RTRIM(text) END, '')
 within group (order by sequence) AS query_statement
from stl_utilitytext where query=pg_last_query_id();
```

Para usar o SQL reconstruído resultante em seu cliente, substitua os caracteres especiais por uma nova linha (\n).

```
                                                query_statement


-----------------------------------------------------------------------------------------------
 set query_group to
 '000000000000000000000000000000000000000000000000000000000000000000000000000000000000\n00000000000
                  000000';
```

STL_VACUUM

Exibe as estatísticas relativas a linhas e blocos para as tabelas que foram limpadas.

A exibição mostra informações específicas de quando cada operação de vácuo começou e terminou e demonstra os benefícios de executar a operação. Para obter informações sobre os requisitos para executar este comando, consulte a descrição do comando VACUUM.

STL_VACUUM só permanece visível para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_VACUUM_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
xid	bigint	O ID da transação para a instrução VACUUM. Você pode associar esta tabela à visualização STL_QUERY para ver as instruções SQL individuais que são executadas para uma determinada transação VACUUM. Se você limpar o banco de dados inteiro, cada tabela é limpada em uma transação separada.
table_id	integer	O ID da tabela.
status	character(30)	O status da operação VACUUM para cada tabela. Os valores possíveis são os seguintes: • **`Started`** • **`Started Delete Only`**

Nome da coluna	Tipo de dados	Descrição
		• **Started Delete Only (Sorted >= nn%)** Apenas a fase de exclusão foi iniciada para um VACUUM FULL. A fase de classificação foi ignorada porque a tabela já foi classificada no limite ou acima do limite de classificação. • **Started Sort Only** • **Started Ranged Partition** • **Started Reindex** • **Finished** O horário em que a operação foi concluída para a tabela. Para saber quanto tempo uma operação de limpeza levou em uma determinada tabela, subtraia o horário de início do horário de término para um ID de transação e ID de tabela específicos. • **Skipped** A tabela foi ignorada porque já foi totalmente classific ada e não existem linhas marcadas para exclusão. • **Skipped (delete only)** A tabela foi ignorada porque a opção DELETE ONLY foi especificada e não existem linhas marcadas para exclusão. • **Skipped (sort only)** A tabela foi ignorada porque a opção SORT ONLY foi especificada e a tabela já foi totalmente classificada. • **Skipped (sort only, sorted>=xx%)** A tabela foi ignorada porque a opção SORT ONLY foi especificada e a tabela já foi classificada no limite ou acima do limite de classificação.

Nome da coluna	Tipo de dados	Descrição
		• **`Skipped (0 rows)`** A tabela foi ignorada porque estava vazia. • **`VacuumBG`** Uma operação de limpeza automática foi executada em segundo plano. Esse status é anexado a outros status quando eles são executados automaticamente. Por exemplo, uma limpeza somente de exclusão executada automaticamente teria uma linha inicial com o status `[VacuumBG] Started Delete Only .` Para obter mais informações sobre como configurar o limite de classificação do VACUUM, consulte VACUUM.
rows	bigint	O número real de linhas na tabela e mais todas as linhas excluídas que ainda estão armazenadas em disco (esperando para serem limpadas). Esta coluna mostra a contagem antes do início da limpeza para as linhas com status **`Started`** e a contagem após a limpeza para as linhas com status **`Finished`**.
sortedrows	integer	O número de linhas na tabela que estão classificadas. Esta coluna mostra a contagem antes do início da limpeza para as linhas definidas com **`Started`** na coluna Status e a contagem após a limpeza para as linhas definidas com **`Finished`** na coluna Status.
blocks	integer	O número total de blocos de dados usados para armazenar os dados da tabela antes da operação de limpeza (linhas com status **`Started`**) e após a operação de limpeza (coluna **`Finished`**). Cada bloco de dados usa 1 MB.

Nome da coluna	Tipo de dados	Descrição
max_merge_partitions	integer	Essa coluna é usada para a análise de performance e representa o número máximo de partições que a limpeza pode processar para a tabela por iteração de fase de mesclagem. (A limpeza classifica a região não classificada em uma ou mais partições classificadas. Dependendo do número de colunas na tabela e da configuração atual do Amazon Redshift, a fase de mesclagem pode processar um número máximo de partições em uma única iteração de mesclagem. A fase de mesclagem ainda pode prosseguir se o número de partições classificadas ultrapassar o número máximo de partições de mesclagem, mas serão necessárias mais iterações de mesclagem).
eventtime	timestamp	Quando a operação de limpeza começou ou terminou.
reclaimable_rows	bigint	O número de linhas recuperáveis para o cutoff_xid atual. Essa coluna mostra o número estimado de linhas recuperáveis do Redshift antes do início da limpeza para linhas com um status **`Started`** e o número real de linhas recuperáveis restantes após a limpeza para linhas com o status **`Finished`**.
reclaimable_space_mb	bigint	Espaço recuperável em MB para o cutoff_xid atual. Essa coluna mostra a quantidade estimada de linhas recuperáveis do Redshift antes do início da limpeza para linhas com o status **`Started`** e a quantidade real de espaço recuperável restante após a limpeza para linhas com o status **`Finished`**.
cutoff_xid	bigint	O ID da transação de limite da operação de VACUUM. Todas as transações após o limite não são incluídas na operação de VACUUM.

Nome da coluna	Tipo de dados	Descrição
is_recluster	integer	Se 1 (true), a operação de VACUUM executou o algoritmo de reagrupamento; se 0 (false), não executou.

Consultas de exemplo

A consulta a seguir relata as estatísticas de limpeza para a tabela 108313. A tabela foi limpada depois de uma série de inserções e exclusões.

```
select xid, table_id, status, rows, sortedrows, blocks, eventtime,
                reclaimable_rows, reclaimable_space_mb
from stl_vacuum where table_id=108313 order by eventtime;

 xid   | table_id | status                  | rows | sortedrows | blocks |  eventtime
         | reclaimable_rows | reclaimable_space_mb
-------+----------+-------------------------+------+------------+--------
+----------------------+------------------+----------------------
 14294 |   108313 | Started                 | 1950 |        408 |     28 |  2016-05-19
 17:36:01 |              984 |                   17
 14294 |   108313 | Finished                |  966 |        966 |     11 |  2016-05-19
 18:26:13 |                0 |                    0
 15126 |   108313 | Skipped(sorted>=95%)    |  966 |        966 |     11 |  2016-05-19
 18:26:38 |                0 |                    0
```

No início de VACUUM, a tabela continha 1.950 linhas armazenadas em 28 blocos de 1 MB. O Amazon Redshift estimou que poderia recuperar 984 ou 17 blocos de espaço em disco com uma operação de limpeza.

Na linha do status Finished (Concluído), a coluna ROWS mostra um valor de 966 e o valor da coluna BLOCKS é 11, abaixo de 28. A limpeza recuperou a quantidade estimada de espaço em disco, sem linhas ou espaço recuperáveis após a conclusão da operação de limpeza.

Na fase de classificação (transação 15126), a limpeza ignorou a tabela, pois as linhas foram inseridas na ordem da chave de classificação.

O exemplo a seguir mostra as estatísticas de uma limpeza com SORT ONLY na tabela SALES (tabela 110116 neste exemplo) após uma grande operação INSERT:

```
vacuum sort only sales;

select xid, table_id, status, rows, sortedrows, blocks, eventtime
from stl_vacuum order by xid, table_id, eventtime;

xid |table_id|     status      | rows  |sortedrows|blocks|     eventtime
----+--------+-----------------+-------+----------+------+-------------------
...
2925| 110116 |Started Sort Only|1379648|   172456 |  132 | 2011-02-24 16:25:21...
2925| 110116 |Finished         |1379648|  1379648 |  132 | 2011-02-24 16:26:28...
```

STL_WINDOW

Analisa as etapas da consulta que realizam funções da janela.

STL_WINDOW permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_WINDOW contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.

Nome da coluna	Tipo de dados	Descrição
slice	integer	O número que identifica a fatia em que a consulta estava sendo executada.
segment	integer	O número que identifica o segmento da consulta.
etapa	integer	Etapa da consulta que foi executada.
starttime	timestamp	O horário (em UTC) de início da consulta. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Horário em UTC em que a consulta foi finalizada. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .
tasknum	integer	Número do processo de tarefa de consulta que foi atribuído para executar a etapa.
rows	bigint	O número total de linhas que foram processadas.
is_diskbased	character(1)	Se o valor é true (t), a consulta foi realizada como uma operação em disco. Se o valor é false (f), a consulta foi realizada na memória.
workmem	bigint	O número total de bytes na memória de trabalho que foram atribuídos à etapa.

Consultas de exemplo

O exemplo a seguir retorna os resultados da função de janela para a fatia 0 e o segmento 3.

```
select query, tasknum, rows, is_diskbased, workmem
from stl_window
where slice=0 and segment=3;
```

```
 query | tasknum | rows | is_diskbased | workmem
-------+---------+------+--------------+----------
 86326 |      36 | 1857 | f            | 95256616
   705 |      15 | 1857 | f            | 95256616
 86399 |      27 | 1857 | f            | 95256616
   649 |      10 |    0 | f            | 95256616
(4 rows)
```

STL_WLM_ERROR

Registra todos os erros relacionados a WLM ocorridos.

STL_WLM_ERROR permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
recordtime	timestamp	O horário em que o erro ocorreu.
pid	integer	O ID do processo que gerou o erro.
error_string	character(256)	A descrição do erro.

STL_WLM_RULE_ACTION

Registra detalhes sobre as ações resultantes das regras de monitoramento de consultas de WLM associadas às filas definidas pelo usuário. Para obter mais informações, consulte Regras de monitoramento de consulta do WLM.

STL_WLM_RULE_ACTION permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O usuário que executou a consulta.
consultar	integer	ID da consulta.
service_class	integer	ID da classe de serviço. As filas de consultas são definidas na configuração do WLM. As classes de serviço maiores do que 5 são filas definidas pelo usuário.
regra	character(256)	O nome de uma regra de monitoramento de consulta.
ação	character(256)	A ação resultante. Os valores possíveis são: • log • hop(reassign) • hop(restart) • abort • change_query_priority • nenhuma O valor `none` indica que os predicados da regra foram satisfeitos mas a ação foi substituída por uma outra regra com uma ação de severidade maior.
recordtime	timestamp	O horário em que a ação foi registrada em UTC.
action_value	character(256)	Se `action` for `change_query_priority` , os valores possíveis serão `highest`, `high`, `normal`, `low` e `lowest`. Se `action` for `log`, `hop` ou `abort`, o valor será vazio.
service_class_name	character(64)	O nome da classe de serviços.

Consultas de exemplo

O exemplo a seguir encontra as consultas que foram interrompidas por uma regra de monitoramento de consulta.

```
Select query, rule
from stl_wlm_rule_action
where action = 'abort'
order by query;
```

STL_WLM_QUERY

Contém um registro de cada tentativa de execução de uma consulta em uma classe de serviço processada pelo WLM.

STL_WLM_QUERY permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_QUERY_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
xid	integer	O ID da transação da consulta ou subconsulta.
tarefa	integer	O ID usado para rastrear uma consulta no gerenciador de workload. Ele pode ser associado a vários IDs de consulta. Se uma consulta é reiniciada, a consulta recebe um novo ID de consulta mas não um novo ID de tarefa.

Nome da coluna	Tipo de dados	Descrição
consultar	integer	ID da consulta. Se uma consulta é reiniciada, a consulta recebe um novo ID de consulta mas não um novo ID de tarefa.
service_class	integer	ID da classe de serviço. Para obter uma lista dos IDs de classe de serviço, consulte IDs da classe de serviço do WLM.
slot_count	integer	Número de slots de consulta do WLM que uma consulta usa de acordo com o nível de simultane idade definido para a fila. O padrão é 1. Para obter mais informações, consulte wlm_query_slot_count.
service_class_star t_time	timestamp	O horário em que a consulta foi atribuída à classe de serviço. Este horário está no fuso horário UTC.
queue_start_time	timestamp	O horário em que a consulta entrou na fila da classe de serviço. Este horário está no fuso horário UTC.
queue_end_time	timestamp	O horário em que a consulta saiu da fila da classe de serviço. Este horário está no fuso horário UTC.
total_queue_time	bigint	O número total de microssegundos que a consulta passou na fila.
exec_start_time	timestamp	O horário em que a consulta iniciou a execução na classe de serviço. Este horário está no fuso horário UTC.
exec_end_time	timestamp	O horário em que a consulta concluiu a execução na classe de serviço. Este horário está no fuso horário UTC.
total_exec_time	bigint	O número de microssegundos que a consulta gastou na execução.

Nome da coluna	Tipo de dados	Descrição
service_class_end_time	timestamp	O horário em que a consulta saiu da classe de serviço. Este horário está no fuso horário UTC.
final_state	character(16)	Reservada para uso do sistema.
est_peak_mem	bigint	Reservada para uso do sistema.
query_priority	char(20)	A prioridade da consulta. Os valores possíveis são `n/a`, `lowest`, `low`, `normal`, `high` e `highest`, em que `n/a` significa que a prioridade da consulta não é compatível.
service_class_name	character(64)	O nome da classe de serviço. Para obter mais informações sobre classes de serviço, consulte Tabelas e visualizações do sistema WLM.

Consultas de exemplo

Visualização do tempo médio da consulta em filas e em execução

A consulta a seguir mostra a configuração atual das classes de serviço maiores do que 4. Para obter uma lista dos IDs de classe de serviço, consulte IDs da classe de serviço do WLM.

A consulta a seguir retorna o tempo médio (em microssegundos) que cada consulta passou em filas de consultas e em execução para cada classe de serviço.

```
select service_class as svc_class, count(*),
avg(datediff(microseconds, queue_start_time, queue_end_time)) as avg_queue_time,
avg(datediff(microseconds, exec_start_time, exec_end_time )) as avg_exec_time
from stl_wlm_query
where service_class > 4
group by service_class
order by service_class;
```

Essa consulta retorna os seguintes dados de saída de exemplo:

```
 svc_class | count | avg_queue_time | avg_exec_time
```

```
-----------+-------+----------------+---------------
         5 | 20103 |              0 |         80415
         5 |  3421 |          34015 |        234015
         6 |    42 |              0 |        944266
         7 |   196 |           6439 |       1364399
(4 rows)
```

Visualização do tempo máximo das consultas em filas e em execução

A consulta a seguir retorna o tempo máximo (em microssegundos) que uma consulta passou em todas as filas de consultas e em execução para cada classe de serviço.

```
select service_class as svc_class, count(*),
max(datediff(microseconds, queue_start_time, queue_end_time)) as max_queue_time,
max(datediff(microseconds, exec_start_time, exec_end_time )) as max_exec_time
from stl_wlm_query
where svc_class > 5
group by service_class
order by service_class;
```

```
 svc_class | count | max_queue_time | max_exec_time
-----------+-------+----------------+---------------
         6 |    42 |              0 |       3775896
         7 |   197 |          37947 |      16379473
(4 rows)
```

Tabelas STV para dados de snapshot

Essas tabelas de STV são tabelas virtuais do sistema que contêm snapshots dos dados atuais do sistema.

Tópicos

- STV_ACTIVE_CURSORS
- STV_BLOCKLIST
- STV_CURSOR_CONFIGURATION
- STV_DB_ISOLATION_LEVEL
- STV_EXEC_STATE
- STV_INFLIGHT

- STV_LOAD_STATE
- STV_LOCKS
- STV_ML_MODEL_INFO
- STV_MV_DEPS
- STV_MV_INFO
- STV_NODE_STORAGE_CAPACITY
- STV_PARTITIONS
- STV_QUERY_METRICS
- STV_RECENTS
- STV_SESSIONS
- STV_SLICES
- STV_STARTUP_RECOVERY_STATE
- STV_TBL_PERM
- STV_TBL_TRANS
- STV_WLM_CLASSIFICATION_CONFIG
- STV_WLM_QMR_CONFIG
- STV_WLM_QUERY_QUEUE_STATE
- STV_WLM_QUERY_STATE
- STV_WLM_QUERY_TASK_STATE
- STV_WLM_SERVICE_CLASS_CONFIG
- STV_WLM_SERVICE_CLASS_STATE
- STV_XRESTORE_ALTER_QUEUE_STATE

STV_ACTIVE_CURSORS

A tabela STV_ACTIVE_CURSORS exibe os detalhes dos cursores que estão abertos. Para obter mais informações, consulte DECLARE.

STV_ACTIVE_CURSORS é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações,

consulte Visibilidade de dados em tabelas e visualizações de sistema. Um usuário só pode visualizar cursores abertos por ele. Os superusuários podem visualizar todos os cursores.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
name	characte (256)	Nome do cursor.
xid	bigint	O contexto da transação.
pid	integer	O processo líder que executa a consulta.
starttime	timestam	O horário em que o cursor foi declarado.
row_count	bigint	O número de linhas no conjunto de resultados do cursor.
byte_count	bigint	O número de bytes no conjunto de resultados do cursor.
fetched_rows	bigint	O número de linhas que foram obtidas do conjunto de resultados do cursor.

STV_BLOCKLIST

A tabela STV_BLOCKLIST contém o número de blocos de disco de 1 MB que são usados por cada fatia, tabela ou coluna em um banco de dados.

Use consultas agregadas com a STV_BLOCKLIST, como mostram os exemplos a seguir, para determinar o número de blocos de disco de 1 MB alocados por banco de dados, tabela, fatia ou coluna. Você também pode usar a STV_PARTITIONS para visualizar informações resumidas sobre a utilização do disco.

STV_BLOCKLIST é visível somente aos superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STV_BLOCKLIST registra somente blocos pertencentes ao cluster provisionado ou namespace sem servidor. Se o banco de dados incluir blocos compartilhados de um produtor de unidade de compartilhamento de dados, esses blocos não serão incluídos em STV_BLOCKLIST. Para obter mais informações sobre unidades de compartilhamento de dados, acesse Compartilhamento de dados no Amazon Redshift.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
slice	integer	Fatia do nó.
col	integer	Um índice com base zero para a coluna. Toda tabela que é criada tem três colunas ocultas anexadas a ela: INSERT_XID, DELETE_XID e ROW_ID (OID). Uma tabela com 3 colunas definidas pelo usuário contém na realidade 6 colunas, e as colunas definidas pelo usuário são numeradas internamente com 0, 1 e 2. As colunas INSERT_XID, DELETE_XID e ROW_ID são numeradas com 3, 4 e 5, respectivamente, neste exemplo.
tbl	integer	ID da tabela para a tabela do banco de dados.
blocknum	integer	O ID do bloco de dados.
num_values	integer	O número de valores contidos no bloco.
extended_limits	integer	Para uso interno.
minvalue	bigint	O valor mínimo de um dado do bloco. Armazena os oito primeiros caracteres como inteiros de 64 bits para dados não numéricos. É usado para a varredura de discos.

Nome da coluna	Tipo de dados	Descrição
maxvalue	bigint	O valor máximo de um dado do bloco. Armazena os oito primeiros caracteres como inteiros de 64 bits para dados não numéricos. É usado para a varredura de discos.
sb_pos	integer	Identificador interno do Amazon Redshift para a posição do superbloco no disco.
pinned	integer	Se o bloco é fixado ou não na memória como parte do pré-carre gamento. 0 = false; 1 = true. O padrão é falso.
on_disk	integer	Se o bloco é ou não armazenado automaticamente no disco. 0 = false; 1 = true. O padrão é falso.
modified	integer	Se o bloco foi modificado ou não. 0 = false; 1 = true. O padrão é falso.
hdr_modif ied	integer	Se o cabeçalho do bloco foi modificado ou não. 0 = false; 1 = true. O padrão é falso.
unsorted	integer	Se um bloco está ou não desordenado. 0 = false; 1 = true. O padrão é true (verdadeiro).
tombstone	integer	Para uso interno.
preferred _diskno	integer	O número do disco onde o bloco se encontra, a menos que o disco esteja com uma falha. Depois que o disco for corrigido, o bloco voltará para esse disco.
temporary	integer	Se o bloco contém ou não dados temporários, como de uma tabela temporária ou resultados de consulta intermediários. 0 = false; 1 = true. O padrão é falso.
newblock	integer	Indica se um bloco é ou não novo (true) ou nunca foi confirmado no disco (false). 0 = false; 1 = true.

Nome da coluna	Tipo de dados	Descrição
num_readers	integer	O número de referências em cada bloco.
flags	integer	Sinalizadores internos do Amazon Redshift para o cabeçalho do bloco.

Consultas de exemplo

A tabela STV_BLOCKLIST possui uma linha para cada bloco de disco alocado, de maneira que uma consulta que selecione todas as linhas pode retornar um número muito grande de linhas. Recomendamos usar somente consultas agregadas com a STV_BLOCKLIST.

A exibição SVV_DISKUSAGE fornece informações semelhantes em um formato mais fácil de usar. Entretanto, o exemplo a seguir demonstra uma forma de utilizar a tabela STV_BLOCKLIST.

Para determinar o número de blocos de 1 MB usados para cada coluna na tabela VENUE, digite a seguinte consulta:

```
select col, count(*)
from stv_blocklist, stv_tbl_perm
where stv_blocklist.tbl = stv_tbl_perm.id
and stv_blocklist.slice = stv_tbl_perm.slice
and stv_tbl_perm.name = 'venue'
group by col
order by col;
```

Essa consulta retorna o número de blocos de 1 MB alocados para cada coluna na tabela VENUE, como mostram os seguintes dados de exemplo:

```
 col | count
-----+-------
   0 |  4
   1 |  4
   2 |  4
   3 |  4
   4 |  4
```

```
   5 |  4
   7 |  4
   8 |  4
(8 rows)
```

A consulta a seguir mostra se os dados da tabelas são distribuídos de fato por todas as fatias:

```
select trim(name) as table, stv_blocklist.slice, stv_tbl_perm.rows
from stv_blocklist,stv_tbl_perm
where stv_blocklist.tbl=stv_tbl_perm.id
and stv_tbl_perm.slice=stv_blocklist.slice
and stv_blocklist.id > 10000 and name not like '%#m%'
and name not like 'systable%'
group by name, stv_blocklist.slice, stv_tbl_perm.rows
order by 3 desc;
```

Essa consulta produz os seguintes dados de saída como exemplo, mostrando a distribuição de dados uniforme para a tabela com o maior número de linhas:

```
table   | slice | rows
----------+-------+-------
listing  |    13 | 10527
listing  |    14 | 10526
listing  |     8 | 10526
listing  |     9 | 10526
listing  |     7 | 10525
listing  |     4 | 10525
listing  |    17 | 10525
listing  |    11 | 10525
listing  |     5 | 10525
listing  |    18 | 10525
listing  |    12 | 10525
listing  |     3 | 10525
listing  |    10 | 10525
listing  |     2 | 10524
listing  |    15 | 10524
listing  |    16 | 10524
listing  |     6 | 10524
listing  |    19 | 10524
listing  |     1 | 10523
listing  |     0 | 10521
...
```

```
(180 rows)
```

A consulta a seguir determina se um bloco com marca de exclusão está confirmado no disco:

```
select slice, col, tbl, blocknum, newblock
from stv_blocklist
where  tombstone > 0;

slice | col |   tbl  | blocknum | newblock
-------+-----+--------+----------+----------
4     |  0  | 101285 |    0     |   1
4     |  2  | 101285 |    0     |   1
4     |  4  | 101285 |    1     |   1
5     |  2  | 101285 |    0     |   1
5     |  0  | 101285 |    0     |   1
5     |  1  | 101285 |    0     |   1
5     |  4  | 101285 |    1     |   1
...
(24 rows)
```

STV_CURSOR_CONFIGURATION

A tabela STV_CURSOR_CONFIGURATION exibe as restrições de configuração do cursor. Para obter mais informações, consulte Restrições de cursor.

STV_CURSOR_CONFIGURATION é visível somente aos superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
current_cursor_count	integer	O número de cursores abertos no momento.
max_diskspace_usable	integer	A quantidade de espaço em disco disponível para os cursores, em megabytes. Essa restrição é baseada no tamanho máximo do conjunto de resultados do cursor para o cluster.

Nome da coluna	Tipo de dados	Descrição
current_diskspace_used	integer	A quantidade de espaço em disco utilizada no momento pelos cursores, em megabytes.

STV_DB_ISOLATION_LEVEL

STV_DB_ISOLATION_LEVEL exibe o nível de isolamento atual para bancos de dados. Para obter mais informações sobre níveis de isolamento, consulte Níveis de isolamento no Amazon Redshift.

STV_DB_ISOLATION_LEVEL permanece visível para todos os superusuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
db_name	characte (128)	O nome do banco de dados.
isolation_level	characte (20)	O nível de isolamento do banco de dados. Os valores possíveis incluem `Serializable` e `Snapshot Isolation` .

STV_EXEC_STATE

Use a tabela STV_EXEC_STATE para encontrar informações sobre as consultas e as etapas de consulta em execução nos nós de computação.

Em geral, essas informações são usadas somente para resolver problemas de engenharia. As exibições SVV_QUERY_STATE e SVL_QUERY_SUMMARY extraem suas informações da tabela STV_EXEC_STATE.

STV_EXEC_STATE é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. Ele pode ser usado para unir várias outras tabelas e exibições do sistema.
slice	integer	A fatia do nó onde a etapa foi concluída.
segment	integer	O segmento da consulta que foi executado. Um segmento de consulta é uma série de etapas.
etapa	integer	A etapa do segmento da consulta que foi concluída. Uma etapa é a menor unidade que uma consulta realiza.
starttime	timestamp	O horário em que a etapa foi executada.
currenttime	timestamp	A hora atual.
tasknum	integer	O processo da tarefa de consulta que foi atribuído para concluir a etapa.
rows	bigint	O número de linhas processadas.
bytes	bigint	O número de bytes processados.
rótulo	char(256)	O rótulo da etapa, que consiste no nome da etapa da consulta e, se for aplicável, no ID e no nome da tabela (por exemplo, `scan tbl=100448 name =user`).

Nome da coluna	Tipo de dados	Descrição
		Os IDs de tabela com três dígitos geralmente indicam varreduras de tabelas transitórias. Quando você vê `tbl=0`, isso normalmente indica uma varredura de um valor constante.
is_diskbased	char(1)	Indica se esta etapa da consulta foi concluída como uma operação em disco: true (**t**) ou false (**f**). Somente algumas etapas, como hash, classificação e etapas de agregação podem ir para o disco. Muitos tipos de etapas são sempre concluídos na memória.
workmem	bigint	O número de bytes da memória de trabalho atribuída à etapa.
num_parts	integer	O número de partições em que a tabela hash é particionada durante uma etapa de hash. Um número positivo nessa coluna não significa que a etapa de hash foi executada como uma operação em disco. Verifique o valor na coluna IS_DISKBASED para ver se a etapa de hash foi executada em disco.
is_rrscan	char(1)	O valor true (**t**) indica que a varredura restrita ao intervalo foi utilizada na etapa. O padrão é falso (**f**).
is_delayed_scan	char(1)	O valor true (**t**) indica que a varredura com atraso foi utilizada na etapa. O padrão é falso (**f**).
query_uuid	character(36)	O identificador globalmente exclusivo (UUID) da consulta.

Consultas de exemplo

Em vez de consultar STV_EXEC_STATE diretamente, o Amazon Redshift recomenda consultar SVL_QUERY_SUMMARY ou SVV_QUERY_STATE para obter as informações em STV_EXEC_STATE em um formato mais amigável. Consulte a documentação das tabelas SVL_QUERY_SUMMARY ou SVV_QUERY_STATE para obter mais detalhes.

STV_INFLIGHT

Use a tabela STV_INFLIGHT para determinar quais consultas estão sendo executadas no cluster no momento. Se você estiver solucionando problemas, é útil verificar o status de consultas de longa duração.

A tabela STV_INFLIGHT não mostra as consultas executadas apenas nos nós de liderança. Para obter mais informações, consulte Função de apenas nó líder. STV_INFLIGHT é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_QUERY_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Solução de problemas com STV_INFLIGHT

Se você usa STV_INFLIGHT para solucionar problemas de performance de uma consulta ou de uma coleção de consultas, observe o seguinte:

- Transações abertas de longa duração geralmente aumentam a carga. Essas transações abertas podem resultar em tempos de execução mais longos para outras consultas.
- Os trabalhos de COPY e ETL de longa duração podem afetar outras consultas em execução no cluster, caso estejam consumindo muitos recursos de computação. Na maioria dos casos, mover esses trabalhos de longa duração para períodos de baixo uso aumenta a performance das workloads de relatórios ou análises.
- Há visualizações que fornecem informações relacionadas a STV_INFLIGHT. Isso inclui STL_QUERYTEXT, que captura o texto da consulta para comandos SQL, e SVV_QUERY_INFLIGHT, que une STV_INFLIGHT a STL_QUERYTEXT. Você também pode usar STV_RECENTS com STV_INFLIGHT para solução de problemas. Por exemplo, STV_RECENTS pode indicar se consultas específicas estão em um estado Em execução ou Concluído. A combinação dessas informações com os resultados de STV_INFLIGHT pode fornecer mais dados sobre as propriedades de uma consulta e o impacto nos recursos de computação.

Você também pode monitorar a execução de consultas usando o console do Amazon Redshift.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
slice	integer	A fatia onde a consulta está sendo executada.
consultar	integer	ID da consulta. Ele pode ser usado para unir várias outras tabelas e exibições do sistema.
rótulo	character(320)	O nome do arquivo usado para executar a consulta ou um rótulo definido com o comando SET QUERY_GROUP. Se a consulta não for baseada em arquivos ou o parâmetro QUERY_GROUP não estiver definido, o valor deste campo será branco.
xid	bigint	ID da transação.
pid	integer	ID do processo. Todas as consultas em uma sessão são executadas no mesmo processo, portanto esse valor permanece constante se você executa uma série de consultas na mesma sessão. Use esta coluna para unir a tabela STL_ERROR.
starttime	timestamp	O horário do início da consulta.
texto	character(100)	O texto da consulta, que é truncado após 100 caracteres se a instrução exceder esse limite.
suspended	integer	Se a consulta está suspensa ou não. 0 = false; 1 = true.
insert_pristine	integer	Se as consultas de gravação são/puderam ser executadas enquanto a consulta atual está/estava em execução. 1 = nenhuma consulta de gravação permitida. 0 = consultas de gravação permitidas. Essa coluna deve ser usada na depuração.

Nome da coluna	Tipo de dados	Descrição
concurrency_scaling_status	integer	Indica se a consulta foi executada no cluster principal ou em um cluster de escalabilidade da simultaneidade. Os valores possíveis são os seguintes: 0 - Executada no cluster principal 1 — Executada em um cluster de escalabilidade da simultaneidade

Consultas de exemplo

Para ver todas as consultas em execução no momento no banco de dados, digite a seguinte consulta:

```
select * from stv_inflight;
```

Os dados de saída de exemplo abaixo mostram que duas consultas estão em execução no momento, incluindo a própria consulta com a tabela STV_INFLIGHT e uma consulta que foi executada de um script chamado `avgwait.sql`:

```
select slice, query, trim(label) querylabel, pid,
starttime, substring(text,1,20) querytext
from stv_inflight;

slice|query|querylabel | pid |        starttime         |      querytext
-----+-----+-----------+-----+--------------------------+--------------------
1011 |  21 |           | 646 |2012-01-26 13:23:15.645503|select slice, query,
1011 |  20 |avgwait.sql| 499 |2012-01-26 13:23:14.159912|select avg(datediff(
(2 rows)
```

A consulta a seguir seleciona várias colunas, incluindo concurrency_scaling_status. Essa coluna indica se as consultas estão sendo enviadas ao cluster de escalabilidade simultânea. Se o valor for 1 para alguns resultados, é uma indicação de que recursos de computação de escalabilidade simultânea estão sendo usados. Para obter mais informações, consulte Escalabilidade simultânea..

```
select userid,
```

```
query,
pid,
starttime,
text,
suspended,
concurrency_scaling_status
 from STV_INFLIGHT;
```

O exemplo de saída mostra uma consulta sendo enviada ao cluster de escalabilidade simultânea.

```
 query  | pid     |        starttime           |   text                | suspended
 |  concurrency_scaling_status
--------+---------
+---------------------------|-----------------------|---------------|-----------------------
1234567 | 123456  | 2012-01-26 13:23:15.645503 | select userid, query...  0
    1
2345678 | 234567  | 2012-01-26 13:23:14.159912 | select avg(datediff(...  0
    0
(2 rows)
```

Para obter mais dicas sobre como solucionar problemas de performance de consultas, consulte Solução de problemas de consulta.

STV_LOAD_STATE

Use a tabela STV_LOAD_STATE para encontrar informações sobre o estado atual das instruções COPY em andamento.

O comando COPY atualiza essa tabela a cada um milhão de registros carregados.

STV_LOAD_STATE é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
sessão	integer	O PID da sessão do processo que executa o carregamento.

Nome da coluna	Tipo de dados	Descrição
consultar	integer	ID da consulta. Ele pode ser usado para unir várias outras tabelas e exibições do sistema.
slice	integer	O número da fatia do nó.
pid	integer	ID do processo. Todas as consultas em uma sessão são executadas no mesmo processo, portanto esse valor permanece constante se você executa uma série de consultas na mesma sessão.
recordtime	timestamp	O horário em que o registro é feito.
bytes_to_load	bigint	O número total de bytes a serem carregados por essa fatia. Este número é 0 quando os dados sendo carregados estão compactados
bytes_loaded	bigint	O número de bytes a serem carregados por essa fatia. Se os dados sendo carregados estão compactados, este é o número de bytes carregados depois que os dados são descompactados.
bytes_to_load_compressed	bigint	O número total de bytes de dados compactados a serem carregados por essa fatia. Este número é 0 quando os dados sendo carregados não estão compactados.
bytes_loaded_compressed	bigint	O número de bytes de dados compactados a serem carregados por essa fatia. Este número é 0 quando os dados sendo carregados não estão compactados.
lines	integer	O número de linhas a serem carregadas por essa fatia.
num_files	integer	O número de arquivos a serem carregados por essa fatia.
num_files_complete	integer	O número de arquivos carregados por essa fatia.

Nome da coluna	Tipo de dados	Descrição
current_file	character (256)	O nome do arquivo sendo carregado por essa fatia.
pct_complete	integer	A porcentagem de conclusão da carga de dados por essa fatia.

Consulta de exemplo

Para ver o andamento em cada fatia para um comando COPY, digite a consulta a seguir. Este exemplo usa a função PG_LAST_COPY_ID() para recuperar as informações do último comando COPY.

```
select slice , bytes_loaded, bytes_to_load , pct_complete from stv_load_state where
 query = pg_last_copy_id();

 slice | bytes_loaded | bytes_to_load | pct_complete
-------+--------------+---------------+--------------
     2 |            0 |             0 |            0
     3 |     12840898 |      39104640 |           32
(2 rows)
```

STV_LOCKS

Use a tabela STV_LOCKS para visualizar todas as atualizações nas tabelas do banco de dados.

O Amazon Redshift bloqueia tabelas para evitar que dois usuários atualizem a mesma tabela ao mesmo tempo. Enquanto a tabela STV_LOCKS mostra todas as atualizações do momento, você pode consultar a tabela STL_TR_CONFLICT para ver o log de conflitos dos bloqueios. Use a exibição SVV_TRANSACTIONS para identificar as transações abertas e os problemas de disputa de bloqueio.

STV_LOCKS é visível somente aos superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
table_id	bigint	O ID da tabela que está solicitando o bloqueio.
last_commit	timestamp	A data e a hora da última confirmação na tabela.
last_update	timestamp	A data e a hora da última atualização na tabela.
lock_owner	bigint	O ID da transação associado ao bloqueio.
lock_owner_pid	bigint	O ID do processo associado ao bloqueio.
lock_owner_start_ts	timestamp	A data e a hora de início da transação.
lock_owner_end_ts	timestamp	A data e a hora de término da transação.
lock_status	character (22)	O status do processo que espera ou que mantém um bloqueio.

Consulta de exemplo

Para ver todos os bloqueios que estão ocorrendo nas transações atuais, digite o seguinte comando:

```
select table_id, last_update, lock_owner, lock_owner_pid from stv_locks;
```

Esta consulta retorna os dados de saída de exemplo a seguir, que exibe três bloqueios em ação:

```
 table_id |        last_update         | lock_owner | lock_owner_pid
----------+----------------------------+------------+----------------
100004  | 2008-12-23 10:08:48.882319 |       1043 |           5656
100003  | 2008-12-23 10:08:48.779543 |       1043 |           5656
100140  | 2008-12-23 10:08:48.021576 |       1043 |           5656
(3 rows)
```

STV_ML_MODEL_INFO

O estado das informações sobre o estado atual do modelo de Machine Learning.

STV_ML_MODEL_INFO é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
schema_name	char(128)	O namespace do modelo.
user_name	char(128)	O proprietário do modelo.
model_name	char(128)	O nome do modelo.
life_cycle	char(20)	O status do ciclo de vida do modelo.
is_refreshable	integer	O estado do modelo se ele é atualizável se as tabelas e colunas originais na consulta de treinamento ainda existirem e o usuário ainda tiver as permissões para elas. Os valores possíveis são: 1 (atualizável) e 0 (não atualizáv el).
model_state	char(128)	O estado atual do modelo.

Consulta de exemplo

A consulta a seguir exibe o estado atual dos modelos de Machine Learning.

```
SELECT schema_name, model_name, model_state
FROM stv_ml_model_info;

 schema_name |        model_name             |             model_state
-------------+-------------------------------+-------------------------------------
 public      | customer_churn_auto_model     | Train Model On SageMaker In Progress
 public      | customer_churn_xgboost_model | Model is Ready
```

```
(2 row)
```

STV_MV_DEPS

A tabela STV_MV_DEPS mostra as dependências de visualizações materializadas em outras visualizações materializadas no Amazon Redshift.

Para obter mais informações sobre visões materializadas, consulte Visões materializadas no Amazon Redshift.

STV_MV_DEPS é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários comuns só podem listar visões materializadas que residem nos esquemas aos quais eles têm acesso. Para ter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
db_name	char(128)	O banco de dados que contém a visualização materializada especificada.
schema	char(128)	O esquema da visão materializada.
name	char(128)	O nome da visualização materializada.
ref_schema	char(128)	O esquema de visualização materializada do qual essa visualização materializada depende.
ref_name	char(128)	O nome da visualização materializada da qual essa visualização materializada depende.
ref_database_name	char(128)	O nome do banco de dados do qual essa visão materializada depende.

Consulta de exemplo

A consulta a seguir retorna uma linha de saída que indica que a visualização materializada `mv_over_foo` usa a visualização materializada `mv_foo` em sua definição como uma dependência.

```
CREATE SCHEMA test_ivm_setup;
CREATE TABLE test_ivm_setup.foo(a INT);
CREATE MATERIALIZED VIEW test_ivm_setup.mv_foo AS SELECT * FROM test_ivm_setup.foo;
CREATE MATERIALIZED VIEW test_ivm_setup.mv_over_foo AS SELECT * FROM
 test_ivm_setup.mv_foo;

SELECT * FROM stv_mv_deps;

 db_name | schema          | name        |   ref_schema   | ref_name |
 ref_database_name
---------+-----------------+-------------+----------------+----------
+------------------
 dev     | test_ivm_setup  | mv_over_foo | test_ivm_setup | mv_foo   | dev
```

STV_MV_INFO

A tabela STV_MV_INFO contém uma linha para cada visualização materializada, se os dados estão obsoletos e informações de estado.

Para obter mais informações sobre visões materializadas, consulte Visões materializadas no Amazon Redshift.

STV_MV_INFO é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários comuns só podem listar visões materializadas que residem nos esquemas aos quais eles têm acesso. Para ter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
db_name	char(128)	O banco de dados que contém a visão materiali zada.
schema	char(128)	O esquema do banco de dados.
name	char(128)	O nome da visualização materializada.
updated_u pto_xid	bigint	Reservado para uso interno.

Nome da coluna	Tipo de dados	Descrição
is_stale	char(1)	Um `t` indica que a visualização materializada está obsoleta. Uma visualização materializada obsoleta é aquela que não foi atualizada nas tabelas de base atualizadas. Um `f` indica que a visão materializada está atualizada com as alterações da tabela base. Um valor u indica uma desatualização desconhecida, por exemplo, no caso de MVs em tabelas externas simples. É possível que as informações não sejam precisas se uma atualização não tiver sido executada desde a última reinicialização. A coluna `is_stale` é sempre definida como `t` se a visão materializada depende de uma função mutável. Uma função mutável retorna um resultado diferente quando recebe o mesmo argumento ou argumentos. Por exemplo, as funções que retornam uma data ou data e hora são, em sua maioria, funções mutáveis.
owner_user_name	char(128)	O usuário que possui a visualização materializada.

Nome da coluna	Tipo de dados	Descrição
estado	integer	O estado da visualização materializada da seguinte forma: • 0 - A visão materializada é totalmente recomputada quando atualizada. • 1 - A visão materializada é incremental. • 101 - A visão materializada não pode ser atualizada devido a uma coluna eliminada. Essa restrição é aplicável mesmo que a coluna não seja usada na visualização materializada. • 102 - A visão materializada não pode ser atualizada devido a um tipo de coluna alterado. Essa restrição é aplicável mesmo que a coluna não seja usada na visualização materializada. • 103 - A visão materializada não pode ser atualizada devido a uma tabela renomeada. • 104 - A visão materializada não pode ser atualizada devido a uma coluna renomeada. Essa restrição é aplicável mesmo que a coluna não seja usada na visualização materializada. • 105 - A visão materializada não pode ser atualizada devido a um esquema renomeado.
autorewrite	char(1)	Um `t` indica que a visualização materializada é elegível para reescrita automática das consultas.
autorefresh	char(1)	Um `t` indica que a visualização materializada pode ser atualizada automaticamente.

Consulta de exemplo

Para exibir o estado de todas as visões materializadas, execute a consulta a seguir.

```
select * from stv_mv_info;
```

Essa consulta retorna os seguintes dados de saída de exemplo.

```
 db_name |       schema       |   name   | updated_upto_xid | is_stale | owner_user_name
 | state | autorefresh | autorewrite
---------+--------------------+---------+------------------+----------
+-----------------+-------+------------+------------
 dev     | test_ivm_setup     | mv       |             1031 | f        | catch-22
 |     1 |           1 |           0
 dev     | test_ivm_setup     | old_mv   |              988 | t        | lotr
 |     1 |           0 |           1
```

STV_NODE_STORAGE_CAPACITY

A tabela STV_NODE_STORAGE_CAPACITY mostra detalhes da capacidade total de armazenamento e da capacidade total usada para cada nó em um cluster. Ele contém uma linha para cada nó.

STV_NODE_STORAGE_CAPACITY só é visível para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
nó	integer	O número de nós.
used	integer	O número de blocos de disco de 1 MB atualmente em uso no nó. Para tipos de nó RG e RA3, os blocos usados incluem blocos armazenados em cache localmente e blocos persistentes no Amazon S3.
capacity	integer	A capacidade total de armazenamento provisionada para o nó em blocos de 1 MB. A capacidade inclui espaço reservado pelo Amazon Redshift em tipos de nó DC2 para uso interno. A capacidade é maior do que a capacidade nominal do nó, que é a quantidade de espaço do nó disponível para os dados do usuário. Para tipos de nó RG e RA3,

Nome da coluna	Tipo de dados	Descrição
		essa capacidade é a mesma que a cota total de armazenamento gerenciado para o cluster. Para obter mais informações sobre a capacidade por tipo de nó, consulte “Detalhes do tipo de nó” no Guia de gerenciamento de clusters do Amazon Redshift.

Consultas de exemplo

> **Note**
>
> Os resultados dos exemplos a seguir variam com base nas especificações de nó do seu cluster. Adicione a coluna `capacity` ao seu SQL SELECT para recuperar a capacidade do cluster.

A consulta a seguir retorna espaço usado e capacidade total em blocos de 1 MB no disco. Este exemplo foi executado em um cluster dc2.8xlarge de dois nós.

```
select node, used from stv_node_storage_capacity order by node;
```

Essa consulta retorna os seguintes dados de saída de exemplo.

```
 node | used
------+-------
    0 | 30597
    1 | 27089
```

A consulta a seguir retorna espaço usado e capacidade total em blocos de 1 MB no disco. Este exemplo foi executado em um cluster ra3.16xlarge de dois nós.

```
select node, used from stv_node_storage_capacity order by node;
```

Essa consulta retorna os seguintes dados de saída de exemplo.

```
 node | used
------+-------
    0 | 30591
    1 | 27103
```

STV_PARTITIONS

Use a tabela STV_PARTITIONS para descobrir a performance da velocidade do disco e a utilização do disco para Amazon Redshift.

STV_PARTITIONS contém uma linha por nó por volume lógico do disco.

STV_PARTITIONS é visível somente aos superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
proprietário	integer	O nó do disco que é proprietário da partição.
host	integer	O nó que está fisicamente vinculado à partição.
diskno	integer	O disco que contém a partição.
part_begin	bigint	O deslocamento da partição. Os dispositivos não processad os são particionados logicamente para abrir espaço para blocos espelho.
part_end	bigint	Final da partição.
used	integer	O número de blocos de 1 MB do disco em uso na partição no momento.
tossed	integer	O número de blocos que estão prontos para serem excluídos mas não foram removidos ainda porque não é seguro liberar seus endereços de disco. Se os endereços fossem liberados imediatamente, uma transação pendente poderia

Nome da coluna	Tipo de dados	Descrição
		gravar no mesmo local do disco. Portanto, esses blocos descartados são liberados somente a partir da próxima confirmação. Os blocos de disco podem ser descartados, por exemplo, quando uma coluna da tabela é descartada durante operações de INSERT ou de consultas no disco.
capacity	integer	A capacidade total da partição em blocos de 1 MB no disco.
reads	bigint	O número de operações de leitura ocorridas desde a última vez que o cluster foi reiniciado.
writes	bigint	O número de operações de gravação ocorridas desde a última vez que o cluster foi reiniciado.
seek_forward	integer	O número de vezes que uma solicitação não é feita para o endereço subsequente em relação a um dado endereço anterior.
seek_back	integer	O número de vezes que uma solicitação não é feita para o endereço anterior em relação a um dado endereço subsequente.
is_san	integer	Indica se a partição pertence a um SAN. Os valores válidos são **0** (false) ou **1** (true).
falha	integer	Essa coluna está suspensa.
mbps	integer	A velocidade do disco em megabytes por segundo.
mount	character (256)	O caminho de diretório para o dispositivo.

Consulta de exemplo

A consulta a seguir retorna o espaço usado e a capacidade do disco, em blocos de 1 MB, e calcula a utilização do disco como uma porcentagem do espaço do disco não processado. O espaço em

disco bruto inclui o espaço reservado pelo Amazon Redshift para uso interno, portanto, é maior do que a capacidade nominal do disco, que é a quantidade de espaço em disco disponível para o usuário. A métrica Porcentagem de espaço em disco usado na guia Performance do Console de Gerenciamento do Amazon Redshift relata a porcentagem da capacidade nominal do disco usada por seu cluster. Recomendamos que você monitore a métrica Percentage of Disk Space Used para manter a utilização dentro dos limites de capacidade nominal do disco do cluster.

> **Important**
>
> É altamente recomendável que você não exceda a capacidade nominal de disco do cluster. Embora seja tecnicamente possível exceder a capacidade nominal de disco em determinadas circunstâncias, isso diminui a tolerância a falhas do cluster e aumenta o risco de perda de dados.

Este exemplo foi executado em um cluster de dois nós com seis partições lógicas de disco por nó. O espaço está sendo usado com bastante uniformidade pelos discos, com aproximadamente 25% de utilização por disco.

```
select owner, host, diskno, used, capacity,
(used-tossed)/capacity::numeric *100 as pctused
from stv_partitions order by owner;

 owner | host | diskno |  used  | capacity | pctused
-------+------+--------+--------+----------+---------
   0   |  0   |    0   | 236480 |  949954  | 24.9
   0   |  0   |    1   | 236420 |  949954  | 24.9
   0   |  0   |    2   | 236440 |  949954  | 24.9
   0   |  1   |    2   | 235150 |  949954  | 24.8
   0   |  1   |    1   | 237100 |  949954  | 25.0
   0   |  1   |    0   | 237090 |  949954  | 25.0
   1   |  1   |    0   | 236310 |  949954  | 24.9
   1   |  1   |    1   | 236300 |  949954  | 24.9
   1   |  1   |    2   | 236320 |  949954  | 24.9
   1   |  0   |    2   | 237910 |  949954  | 25.0
   1   |  0   |    1   | 235640 |  949954  | 24.8
   1   |  0   |    0   | 235380 |  949954  | 24.8

(12 rows)
```

STV_QUERY_METRICS

Contém informações de métricas, como o número de linhas processadas, utilização da CPU, entrada/saída e utilização do disco, para consultas ativas em execução nas filas de consultas definidas pelo usuário (classes de serviço). Para visualizar as métricas para as consultas que foram concluídas, consulte a tabela do sistema STL_QUERY_METRICS.

Para as métricas de consulta, as amostras são feitas em intervalos de um segundo. Consequentemente, diferentes execuções da mesma consulta podem retornar tempos um pouco diferentes. Além disso, os segmentos de consultas executados em menos de 1 segundo podem não ser registrados.

A tabela STV_QUERY_METRICS rastreia e agrega as métricas no nível da consulta, do segmento e das etapas. Para obter informações sobre os segmentos e as etapas de uma consulta, acesse Planejamento de consulta e fluxo de trabalho de execução. Muitas métricas (como, por exemplo, `max_rows`, `cpu_time`, etc.) são resultado da soma obtida das fatias de nós. Para obter mais informações sobre as fatias de nós, consulte Arquitetura de sistema do data warehouse.

Para determinar o nível no qual uma linha relata suas métricas, examine as colunas `segment` e `step_type`:

- Se ambos `segment` e `step_type` são `-1`, a linha relata as métricas no nível da consulta.
- Se `segment` não é `-1` e `step_type` é `-1`, a linha relata as métricas no nível do segmento.
- Se ambos `segment` e `step_type` são diferentes de `-1`, a linha relata as métricas no nível da etapa.

STV_QUERY_METRICS permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que executou a consulta que gerou a entrada.
service_class	integer	O ID da fila de consultas do WLM (classe de serviço). As filas de consultas são definidas na configuração do WLM. As métricas são relatadas somente para as filas definidas pelo usuário.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
starttime	timestamp	O horário (em UTC) do início da execução da consulta, com 6 dígitos de precisão fracionária de segundos. Por exemplo: `2009-06-12 11:29:19.131358` .
slices	integer	O número de fatias do cluster.
segment	integer	O número do segmento. Uma consulta consiste em vários segmentos e cada segmento consiste em uma ou mais etapas. Os segmentos de uma consulta podem ser executados em paralelo. Cada segmento executa em um único processo. Se o valor do segmento é -1, os valores das métricas do segmento são acumulados no nível de consulta.
step_type	integer	O tipo da etapa que foi executada. Para obter uma descrição dos tipos de etapas, consulte Tipos de etapas.
rows	bigint	O número de linhas processadas por uma etapa.
max_rows	bigint	O número máximo de linhas geradas na saída de uma etapa, agregado de todas as fatias.
cpu_time	bigint	O tempo de CPU usado, em microssegundos. No nível de segmento, o tempo total de CPU usado pelo segmento em

Nome da coluna	Tipo de dados	Descrição
		todas as fatias. No nível de consulta, a soma dos tempos de CPU da consulta em todas as fatias e segmentos.
max_cpu_time	bigint	O tempo máximo de CPU usado, em microssegundos. No nível de segmento, o tempo máximo de CPU usado pelo segmento em todas as fatias. No nível de consulta, o tempo máximo de CPU usado por qualquer um dos segmento da consulta.
blocks_read	bigint	O número de blocos de 1 MB lidos pela consulta ou segmento.
max_block s_read	bigint	O número máximo de blocos de 1 MB lidos pelo segmento, agregado de todas as fatias. No nível de segmento, o número máximo de blocos de 1 MB lidos para o segmento em todas as fatias. No nível de consulta, o número máximo de blocos de 1 MB lidos por qualquer um dos segmentos da consulta.
run_time	bigint	O tempo total de execução, somado de todas de fatias. O tempo de execução não inclui o tempo de espera. No nível de segmento, o tempo de execução para o segmento, somado de todas as fatias. No nível de consulta, o tempo de execução da consulta, somado de todas as fatias e segmentos. Como esse valor é uma soma, o tempo de execução não está relacionado ao tempo de execução de consulta.
max_run_time	bigint	O tempo máximo decorrido para um segmento, em microsseg undos. No nível de segmento, o tempo máximo de execução para o segmento em todas as fatias. No nível de consulta, o tempo máximo de execução para qualquer um dos segmento da consulta.

Nome da coluna	Tipo de dados	Descrição
max_blocks_to_disk	bigint	A quantidade máxima de espaço em disco usada para gravar resultados intermediários, em blocos de 1 MB. No nível de segmento, a quantidade máxima de espaço em disco usada pelo segmento em todas as fatias. No nível de consulta, a quantidade máxima de espaço em disco usada por qualquer um dos segmento da consulta.
blocks_to_disk	bigint	A quantidade de espaço em disco usada por uma consulta ou segmento para gravar resultados intermediários, em blocos de 1 MB.
etapa	integer	Etapa da consulta que foi executada.
max_query_scan_size	bigint	A quantidade máxima de dados pesquisados em varredura por uma consulta, em MB. No nível de segmento, a quantidade máxima de dados pesquisados em varredura por um segmento em todas as fatias. No nível de consulta, a quantidade máxima de dados pesquisados em varredura por qualquer um dos segmento da consulta.
query_scan_size	bigint	A quantidade de dados pesquisados em varredura por uma consulta, em MB.
query_priority	integer	A prioridade da consulta. Os valores possíveis são `-1`, `0`, `1`, `2`, `3` e `4`, em que `-1` significa que a prioridade da consulta não é compatível.
query_queue_time	bigint	O tempo em microssegundos que a consulta estava em fila.

Tipos de etapas

A tabela a seguir lista os tipos de etapas relevantes para os usuários de banco de dados. A tabela não lista os tipos de etapas que são somente para uso interno. Se o tipo da etapa é -1, a métrica não é relatada no nível da etapa.

Step type (Tipo de etapa)	Descrição
1	Varredura de tabela
2	Inserir linhas
3	Agregar linhas
6	Etapa de classificação
7	Etapa de mesclagem
8	Etapa de distribuição
9	Etapa de distribuição de transmissão
10	Junção de hash
11	Junção de mesclagem
12	Etapa de gravação
14	Hash
15	Junção de loop aninhado
16	Campos e expressões de projeto
17	Limitar o número de linhas retornadas
18	Exclusiva
20	Excluir linhas
26	Limitar o número de linhas classificadas retornadas
29	Computar uma função de janela
32	UDF
33	Exclusivo

Step type (Tipo de etapa)	Descrição
37	Retornar as linhas do nó de liderança para o cliente
38	Retornar as linhas dos nós de computação para o nó de liderança
40	Varredura de espectro.

Consulta de exemplo

Para encontrar as consultas ativas com alto nível de tempo de CPU (mais de 1.000 segundos), execute a consulta a seguir.

```
select query, cpu_time / 1000000 as cpu_seconds
from stv_query_metrics where segment = -1 and cpu_time > 1000000000
order by cpu_time;

query | cpu_seconds
------+------------
25775 |        9540
```

Para encontrar as consultas ativas com uma junção de loop aninhado que retornaram mais de um milhão de linhas, execute a seguinte consulta.

```
select query, rows
from stv_query_metrics
where step_type = 15 and rows > 1000000
order by rows;

query | rows
------+-----------
25775 | 1580225854
```

Para encontrar as consultas ativas cujas execuções duraram mais de 60 segundos e que usaram menos de 10 segundos de tempo de CPU, execute a seguinte consulta.

```
select query, run_time/1000000 as run_time_seconds
from stv_query_metrics
where segment = -1 and run_time > 60000000 and cpu_time < 10000000;
```

```
query | run_time_seconds
------+-----------------
25775 |              114
```

STV_RECENTS

Use a tabela STV_RECENTS para encontrar informações sobre as consultas que estão ativas e as consultas que foram executadas recentemente em um banco de dados.

STV_RECENTS é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_QUERY_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Solução de problemas com STV_RECENTS

STV_RECENTS é particularmente útil para determinar se uma consulta ou coleção de consultas está sendo executada ou concluída no momento. Também mostra a duração em que uma consulta está sendo executada. Isso é útil para ter uma ideia de quais consultas são de longa duração.

Você pode unir STV_RECENTS a outras visualizações do sistema, como STV_INFLIGHT, para coletar metadados adicionais sobre a execução de consultas. (Há um exemplo que mostra como fazer isso na seção de exemplos de consultas.) Também é possível usar os registros retornados dessa visualização com os atributos de monitoramento no console do Amazon Redshift para solucionar problemas em tempo real.

As visualizações do sistema que complementam STV_RECENTS incluem STL_QUERYTEXT, que recupera o texto da consulta para comandos SQL, e SVV_QUERY_INFLIGHT, que une STV_INFLIGHT a STL_QUERYTEXT.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.

Nome da coluna	Tipo de dados	Descrição
status	character(20)	Status da consulta. Os valores válidos são **`Running`**, **`Done`**.
starttime	timestamp	O horário do início da consulta.
duration	integer	O número de microssegundos transcorridos desde o início da sessão.
user_name	character(50)	O nome do usuário que executou o processo.
db_name	character(50)	O nome do banco de dados.
consultar	character(600)	O texto da consulta, com até 600 caracteres. Todos os caracteres adicionais são truncados.
pid	integer	O ID do processo para a sessão associada à consulta, que é sempre -1 para as consultas que foram concluídas.

Consultas de exemplo

Para determinar quais consultas estão em execução no momento no banco de dados, execute a seguinte consulta:

```
select user_name, db_name, pid, query
from stv_recents
where status = 'Running';
```

Os dados de saída do exemplo mostram uma única consulta sendo executada no banco de dados TICKIT:

```
user_name | db_name |   pid   | query
----------+---------+---------+-------------
dwuser    | tickit  |  19996  |select venuename, venueseats from
venue where venueseats > 50000 order by venueseats desc;
```

O exemplo a seguir retorna uma lista de consultas (se houver) em execução ou em uma fila de espera para execução:

```
select * from stv_recents where status<>'Done';

status |    starttime        | duration |user_name|db_name| query     | pid
-------+---------------------+----------+---------+-------+-----------+------
Running| 2010-04-21 16:11... | 281566454| dwuser  |tickit | select ...| 23347
```

Essa consulta não retorna resultados, a menos que você esteja executando um número de consultas simultâneas e que algumas dessas consultas estejam em uma fila.

O exemplo a seguir amplia o exemplo anterior. Neste caso, as consultas que estão verdadeiramente “em movimento” (em execução, não aguardando) são excluídas do resultado:

```
select * from stv_recents where status<>'Done'
and pid not in (select pid from stv_inflight);
...
```

Para obter mais dicas sobre como solucionar problemas de performance de consultas, consulte Solução de problemas de consulta.

STV_SESSIONS

Use a tabela STV_SESSIONS para visualizar informações sobre as sessões de usuário ativas para Amazon Redshift.

Para visualizar o histórico da sessão, use a tabela STL_SESSIONS em vez de STV_SESSIONS.

STV_SESSIONS permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_SESSION_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
starttime	timestamp	O horário de início da sessão.
process	integer	O ID de processo da sessão.
user_name	character(50)	O usuário associado à sessão.
db_name	character(50)	O nome do banco de dados associado à sessão.
timeout_sec	int	O tempo máximo em segundos em que uma sessão permanece inativa ou ociosa antes do tempo limite. 0 indica que nenhum tempo limite está definido.

Consultas de exemplo

Para realizar uma verificação rápida para ver se algum outro usuário está conectado no Amazon Redshift, digite a seguinte consulta:

```
select count(*)
from stv_sessions;
```

Se o resultado for maior que um, pelo menos um outro usuário está atualmente conectado ao banco de dados.

Para visualizar todas as sessões ativas do Amazon Redshift, digite a seguinte consulta:

```
select *
from stv_sessions;
```

O seguinte resultado mostra quatro sessões ativas em execução no Amazon Redshift:

```
       starttime         | process |user_name                 | db_name
      | timeout_sec
------------------------+---------+--------------------------
+---------------------------+-------------
```

```
  2018-08-06 08:44:07.50 |   13779 | IAMA:aws_admin:admin_grp   | dev
      | 0
  2008-08-06 08:54:20.50 |   19829 | dwuser                     | dev
      | 120
  2008-08-06 08:56:34.50 |   20279 | dwuser                     | dev
      | 120
  2008-08-06 08:55:00.50 |   19996 | dwuser                     | tickit
      | 0
(3 rows)
```

O nome do usuário prefixado com IAMA indica que o usuário se conectou com autenticação única federada. Para obter mais informações, consulte Uso da autenticação do IAM para gerar credenciais do usuário do banco de dados.

STV_SLICES

Use a tabela STV_SLICES para visualizar o mapeamento atual de uma fatia para um nó.

As informações da STV_SLICES são usadas principalmente para fins de investigação.

STV_SLICES é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
nó	integer	O nó do cluster onde a fatia se encontra.
slice	integer	Fatia do nó.
localslice	integer	Essas informações são somente para uso interno.
type	character (1)	Essas informações são somente para uso interno.

Consulta de exemplo

Para ver que nós do cluster estão controlando quais fatias, digite a seguinte consulta:

```
select node, slice from stv_slices;
```

Essa consulta retorna os seguintes dados de saída de exemplo:

```
 node | slice
------+-------
    0 |     2
    0 |     3
    0 |     1
    0 |     0
(4 rows)
```

STV_STARTUP_RECOVERY_STATE

Registra o estado das tabelas que estão temporariamente bloqueadas durante as operações de reinício do cluster. O Amazon Redshift aplica um bloqueio temporário nas tabelas, enquanto elas estão sendo processadas, para resolver transações velhas depois que um cluster é reiniciado.

STV_STARTUP_RECOVERY_STATE permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
db_id	integer	ID do banco de dados.
table_id	integer	ID da tabela.
table_name	character(137)	O nome da tabela.

Consultas de exemplo

Para monitorar quais tabelas estão temporariamente bloqueadas, execute a consulta a seguir depois que o cluster for reiniciado.

```
select * from STV_STARTUP_RECOVERY_STATE;

  db_id | tbl_id | table_name
--------+--------+------------
 100044 | 100058 | lineorder
 100044 | 100068 | part
 100044 | 100072 | customer
 100044 | 100192 | supplier
(4 rows)
```

STV_TBL_PERM

A tabela STV_TBL_PERM contém informações sobre as tabelas permanentes no Amazon Redshift, incluindo tabelas temporárias criadas por um usuário para a sessão atual. A tabela STV_TBL_PERM contém informações para todas as tabelas em todos os bancos de dados.

Essa tabela é diferente da STV_TBL_TRANS, que contém informações sobre as tabelas transitórias de banco de dados que o sistema cria durante o processamento de consultas.

STV_TBL_PERM é visível somente aos superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
slice	integer	A fatia do nó alocada para a tabela.
id	integer	ID da tabela.
name	character (72)	O nome da tabela.
rows	bigint	O número de linhas de dados em uma fatia.
sorted_rows	bigint	O número de linhas em uma fatia que já estão classificadas no disco. Se este número não é igual ao número de ROWS, é preciso limpar a tabela para reclassificar as linhas.
temp	integer	Se a mesa é temporária ou não. 0 = false; 1 = true.

Nome da coluna	Tipo de dados	Descrição
db_id	integer	O ID do banco de dados onde a tabela foi criada.
insert_pristine	integer	Para uso interno.
delete_pristine	integer	Para uso interno.
backup	integer	Valor que indica se a tabela está incluída em instantâneos de cluster. 0 = não; 1 = sim. Para obter mais informações, consulte o parâmetro BACKUP para o comando CREATE TABLE.
dist_style	integer	Estilo de distribuição da tabela à qual a fatia pertence. Para obter informações sobre os valores, consulte Visualização dos estilos de distribuição. Para obter informações sobre os estilos de distribuição, consulte Estilos de distribuição.
block_count	integer	Número de blocos usados pela fatia. O valor será -1 quando a contagem de blocos não puder ser calculada.

Consultas de exemplo

A consulta a seguir retorna uma lista de IDs de tabelas e nomes distintos:

```
select distinct id, name
from stv_tbl_perm order by name;

   id   |          name
--------+------------------------
 100571 | category
 100575 | date
 100580 | event
 100596 | listing
 100003 | padb_config_harvest
 100612 | sales
...
```

As outras tabelas do sistema usam IDs de tabelas e, portanto, saber o ID de uma determinada tabela pode ser muito útil. Neste exemplo, o comando SELECT DISTINCT é usado para remover as duplicatas (as tabelas são distribuídas por diversas fatias).

Para determinar o número de blocos usados por cada coluna na tabela VENUE, digite a seguinte consulta:

```
select col, count(*)
from stv_blocklist, stv_tbl_perm
where stv_blocklist.tbl = stv_tbl_perm.id
and stv_blocklist.slice = stv_tbl_perm.slice
and stv_tbl_perm.name = 'venue'
group by col
order by col;

 col | count
-----+-------
   0 |     8
   1 |     8
   2 |     8
   3 |     8
   4 |     8
   5 |     8
   6 |     8
   7 |     8
(8 rows)
```

Observações de uso

A coluna ROWS inclui a contagem das linhas excluídas que não foram limpadas (ou foram eliminadas mas com a opção de SORT ONLY). Portanto, a soma da coluna ROWS na tabela STV_TBL_PERM pode não ser igual ao resultado de COUNT(*) quando você consulta uma tabela específica diretamente. Por exemplo, se 2 linhas são excluídas da tabela VENUE, o resultado de COUNT(*) é 200, mas o resultado de SUM(ROWS) ainda é 202:

```
delete from venue
where venueid in (1,2);

select count(*) from venue;
count
-------
200
```

```
(1 row)

select trim(name) tablename, sum(rows)
from stv_tbl_perm where name='venue' group by name;

tablename | sum
-----------+-----
venue     | 202
(1 row)
```

Para sincronizar os dados na tabela STV_TBL_PERM, execute uma limpeza total da tabela VENUE.

```
vacuum venue;

select trim(name) tablename, sum(rows)
from stv_tbl_perm
where name='venue'
group by name;

tablename | sum
-----------+-----
venue     | 200
(1 row)
```

STV_TBL_TRANS

Use a tabela STV_TBL_TRANS para obter informações sobre as tabelas transitórias de banco de dados que estão na memória no momento.

As tabelas transitórias normalmente são conjuntos temporários de linhas que são usados como resultados intermediários durante a execução de uma consulta. A diferença entre a tabela STV_TBL_TRANS e a STV_TBL_PERM é que a STV_TBL_PERM contém informações sobre tabelas permanentes de banco de dados.

STV_TBL_TRANS é visível somente aos superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
slice	integer	A fatia do nó alocada para a tabela.
id	integer	ID da tabela.
rows	bigint	O número de linhas de dados na tabela.
tamanho	bigint	O número de bytes alocados para a tabela.
query_id	bigint	ID da consulta.
ref_cnt	integer	O número de referências.
from_suspended	integer	Indica se a tabela foi criada durante uma consulta que está suspensa no momento.
prep_swap	integer	Indica se a tabela transitória está preparada para mudar para o disco, caso seja necessário. (A mudança ocorrerá somente caso o espaço disponível na memória esteja baixo).

Consultas de exemplo

Para visualizar as informações da tabela transitória para uma consulta com o ID de consulta de 90, digite o seguinte comando:

```
select slice, id, rows, size, query_id, ref_cnt
from stv_tbl_trans
where query_id = 90;
```

Esta consulta retorna as informações da tabela transitória para a consulta 90, como mostram os seguintes dados de saída de exemplo:

```
slice | id | rows | size | query_ | ref_ | from_     | prep_
      |    |      |      | id     | cnt  | suspended | swap
```

```
------+----+------+------+--------+------+-----------+-------
 1013 | 95 |    0 |    0 |     90 |    4 |         0 |   0
    7 | 96 |    0 |    0 |     90 |    4 |         0 |   0
   10 | 96 |    0 |    0 |     90 |    4 |         0 |   0
   17 | 96 |    0 |    0 |     90 |    4 |         0 |   0
   14 | 96 |    0 |    0 |     90 |    4 |         0 |   0
    3 | 96 |    0 |    0 |     90 |    4 |         0 |   0
 1013 | 99 |    0 |    0 |     90 |    4 |         0 |   0
    9 | 96 |    0 |    0 |     90 |    4 |         0 |   0
    5 | 96 |    0 |    0 |     90 |    4 |         0 |   0
   19 | 96 |    0 |    0 |     90 |    4 |         0 |   0
    2 | 96 |    0 |    0 |     90 |    4 |         0 |   0
 1013 | 98 |    0 |    0 |     90 |    4 |         0 |   0
   13 | 96 |    0 |    0 |     90 |    4 |         0 |   0
    1 | 96 |    0 |    0 |     90 |    4 |         0 |   0
 1013 | 96 |    0 |    0 |     90 |    4 |         0 |   0
    6 | 96 |    0 |    0 |     90 |    4 |         0 |   0
   11 | 96 |    0 |    0 |     90 |    4 |         0 |   0
   15 | 96 |    0 |    0 |     90 |    4 |         0 |   0
   18 | 96 |    0 |    0 |     90 |    4 |         0 |   0
```

Neste exemplo, você pode ver que os dados da consulta envolvem as tabelas 95, 96 e 98. Como foram alocados zero bytes para essa tabela, a consulta pode ser executada na memória.

STV_WLM_CLASSIFICATION_CONFIG

Contém as regras de classificação atuais para o WLM.

STV_WLM_CLASSIFICATION_CONFIG é visível somente aos superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
id	integer	ID da classe de serviço. Para obter uma lista dos IDs de classe de serviço, consulte IDs da classe de serviço do WLM.
condição	character(128)	As condições da consulta.

Nome da coluna	Tipo de dados	Descrição
action_seq	integer	Reservada para uso do sistema.
ação	character(64)	Reservada para uso do sistema.
action_service_class	integer	A classe de serviço onde a ação ocorreu.

Consulta de exemplo

```
select * from STV_WLM_CLASSIFICATION_CONFIG;

id | condition                                    | action_seq | action |
 action_service_class
---+----------------------------------------------+------------+--------
+---------------------
 1 | (system user) and (query group: health)     |          0 | assign |
     1
 2 | (system user) and (query group: metrics)    |          0 | assign |
     2
 3 | (system user) and (query group: cmstats)    |          0 | assign |
     3
 4 | (system user)                               |          0 | assign |
     4
 5 | (super user) and (query group: superuser)   |          0 | assign |
     5
 6 | (query group: querygroup1)                  |          0 | assign |
     6
 7 | (user group: usergroup1)                    |          0 | assign |
     6
 8 | (user group: usergroup2)                    |          0 | assign |
     7
 9 | (query group: querygroup3)                  |          0 | assign |
     8
10 | (query group: querygroup4)                  |          0 | assign |
     9
11 | (user group: usergroup4)                    |          0 | assign |
     9
```

```
12 | (query group: querygroup*)                    |          0 | assign |
    10
13 | (user group: usergroup*)                      |          0 | assign |
    10
14 | (querytype: any)                              |          0 | assign |
    11
(4 rows)
```

STV_WLM_QMR_CONFIG

Registra a configuração das regras de monitoramento de consultas (QMR) para o WLM. Para obter mais informações, consulte Regras de monitoramento de consulta do WLM.

STV_WLM_QMR_CONFIG é visível somente aos superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
service_class	integer	O ID da fila de consultas do WLM (classe de serviço). As filas de consultas são definidas na configuração do WLM. As regras só podem ser definidas para filas definidas pelo usuário. Para obter uma lista dos IDs de classe de serviço, consulte IDs da classe de serviço do WLM.
rule_name	character(256)	O nome da regra de monitoramento de consultas.
ação	character(256)	A ação da regra. Os valores possíveis são `log`, `hop`, `abort` e `change_query_priority` .
metric_name	character(256)	O nome da métrica.
metric_operator	character(256)	O operador da métrica. Os valores possíveis são >, =, <.
metric_value	double	O valor limite para a métrica especificada que dispara uma ação.
action_value	character(256)	Se `action` for `change_query_priority` , os valores possíveis serão `highest`, `high`, `normal`, `low` e `lowest`.

Nome da coluna	Tipo de dados	Descrição
		Se `action` for `log`, `hop` ou `abort`, o valor será vazio.

Consulta de exemplo

Para visualizar as definições da regra de QMR para todas as classes de serviço maiores do que 5 (que incluem filas definidas pelo usuário), execute a consulta a seguir. Para obter uma lista dos IDs de classe de serviço, consulte IDs da classe de serviço do WLM.

```
Select *
from stv_wlm_qmr_config
where service_class > 5
order by service_class;
```

STV_WLM_QUERY_QUEUE_STATE

Registra o estado atual das filas de consultas para as classes de serviço.

STV_WLM_QUERY_QUEUE_STATE é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_QUERY_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
service_class	integer	ID da classe de serviço. Para obter uma lista dos IDs de classe de serviço, consulte IDs da classe de serviço do WLM.

Nome da coluna	Tipo de dados	Descrição
position	integer	A posição da consulta na fila. A consulta com o menor valor de **`position`** é executada em seguida.
tarefa	integer	O ID usado para rastrear uma consulta no gerenciador de workload. Ele pode ser associado a vários IDs de consulta. Se uma consulta é reiniciada, a consulta recebe um novo ID de consulta mas não um novo ID de tarefa.
consultar	integer	ID da consulta. Se uma consulta é reiniciada, a consulta recebe um novo ID de consulta mas não um novo ID de tarefa.
slot_count	integer	O número de slots da consulta de WLM.
start_time	timestamp	O horário em que a consulta entrou na fila.
queue_time	bigint	O número de microssegundos transcorridos desde que a consulta entrou na fila.

Consulta de exemplo

A consulta a seguir mostra as consultas na fila das classes de serviço maiores do que 4.

```
select * from stv_wlm_query_queue_state
where service_class > 4
order by service_class;
```

Essa consulta retorna os seguintes dados de saída de exemplo.

```
 service_class | position | task | query | slot_count |        start_time          |
 queue_time
---------------+----------+------+-------+------------+----------------------------
+------------
             5 |        0 |  455 |   476 |          5 | 2010-10-06 13:18:24.065838 |
 20937257
```

```
             6 |        1 |  456 |   478 |         5 | 2010-10-06 13:18:26.652906 |
 18350191
(2 rows)
```

STV_WLM_QUERY_STATE

Registra o estado atual das consultas que estão sendo rastreadas pelo WLM.

STV_WLM_QUERY_STATE é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_QUERY_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
xid	integer	O ID da transação da consulta ou subconsulta.
tarefa	integer	O ID usado para rastrear uma consulta no gerenciador de workload. Ele pode ser associado a vários IDs de consulta. Se uma consulta é reiniciada, a consulta recebe um novo ID de consulta mas não um novo ID de tarefa.
consultar	integer	ID da consulta. Se uma consulta é reiniciada, a consulta recebe um novo ID de consulta mas não um novo ID de tarefa.
service_class	integer	ID da classe de serviço. Para obter uma lista dos IDs de classe de serviço, consulte IDs da classe de serviço do WLM.
slot_count	integer	O número de slots da consulta de WLM.

Nome da coluna	Tipo de dados	Descrição
wlm_start_time	timestamp	O horário em que a consulta entrou na fila das tabelas do sistema ou na fila breve de consultas.
estado	character(16)	O estado atual da consulta ou subconsulta. Os valores possíveis são os seguintes: • `Classified` : a consulta foi atribuída a uma classe de serviço. • `Completed` : a consulta concluiu a execução. A consulta foi executada corretamente ou foi cancelada . Para o estado final, verifique os resultados de STL_QUERY. • `Dequeued`: somente para uso interno. • `Evicted`: a consulta foi removida da classe de serviço para reinicialização. • `Evicting`: a consulta está sendo removida da classe de serviço para reinicialização. • `Initialized` : somente para uso interno. • `Invalid`: somente para uso interno. • `Queued`: a consulta foi enviada para a fila de consultas porque não havia slots disponíveis para executá-la. • `QueuedWaiting` : a consulta está aguardando na fila de consultas. • `Rejected`: somente para uso interno. • `Returning` : a consulta está retornando os resultados ao cliente. • `Running`: a consulta está em execução. • `TaskAssigned` : somente para uso interno.
queue_time	bigint	O tempo, em número de microssegundos, que a consulta passou na fila.

Nome da coluna	Tipo de dados	Descrição
exec_time	bigint	Número de microssegundos transcorridos desde que a consulta foi executada.
query_priority	char(20)	A prioridade da consulta. Os valores possíveis são `n/a`, `lowest`, `low`, `normal`, `high` e `highest`, em que `n/a` significa que a prioridade da consulta não é compatível.

Consulta de exemplo

A consulta a seguir mostra todas as consultas em execução no momento nas classes de serviço maiores do que 4. Para obter uma lista dos IDs de classe de serviço, consulte IDs da classe de serviço do WLM.

```
select xid, query, trim(state) as state, queue_time, exec_time
from stv_wlm_query_state
where service_class > 4;
```

Essa consulta retorna os seguintes dados de saída de exemplo:

```
xid    | query | state   | queue_time | exec_time
-------+-------+---------+------------+-----------
100813 | 25942 | Running |          0 |    1369029
100074 | 25775 | Running |          0 | 2221589242
```

STV_WLM_QUERY_TASK_STATE

Contém o estado atual das tarefas de consulta da classe de serviços.

STV_WLM_QUERY_TASK_STATE é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
service_class	integer	ID da classe de serviço. Para obter uma lista dos IDs de classe de serviço, consulte IDs da classe de serviço do WLM.
tarefa	integer	O ID usado para rastrear uma consulta no gerenciador de workload. Ele pode ser associado a vários IDs de consulta. Se uma consulta é reiniciada, a consulta recebe um novo ID de consulta mas não um novo ID de tarefa.
consultar	integer	ID da consulta. Se uma consulta é reiniciada, a consulta recebe um novo ID de consulta mas não um novo ID de tarefa.
slot_count	integer	O número de slots da consulta de WLM.
start_time	timestamp	O horário de início da execução da consulta.
exec_time	bigint	O número de microssegundos transcorridos desde que a consulta entrou em execução.

Consulta de exemplo

A consulta a seguir mostra o estado atual das consultas nas classes de serviço maiores do que 4. Para obter uma lista dos IDs de classe de serviço, consulte IDs da classe de serviço do WLM.

```
select * from stv_wlm_query_task_state
where service_class > 4;
```

Essa consulta retorna os seguintes dados de saída de exemplo:

```
service_class | task | query |         start_time         | exec_time
--------------+------+-------+----------------------------+-----------
    5         |  466 |   491 | 2010-10-06 13:29:23.063787 | 357618748
```

```
(1 row)
```

STV_WLM_SERVICE_CLASS_CONFIG

Registra as configurações da classe de serviço para o WLM.

STV_WLM_SERVICE_CLASS_CONFIG é visível somente aos superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
service_class	integer	ID da classe de serviço. Para obter uma lista dos IDs de classe de serviço, consulte IDs da classe de serviço do WLM.
queueing_strategy	character(32)	Reservada para uso do sistema.
num_query_tasks	integer	O nível de simultaneidade atual da classe de serviço. Se `num_query_tasks` e `target_num_query_tasks` são diferentes, uma transição dinâmica do WLM está em andamento. Um valor de `-1` indica que Auto WLM (WLM automático) está configurado.
target_num_query_tasks	integer	O nível de simultaneidade definido pela alteração de configuração mais recente do WLM.
evictable	character(8)	Reservada para uso do sistema.
eviction_threshold	bigint	Reservada para uso do sistema.
query_working_mem	integer	A quantidade atual de memória de trabalho, em MB por slot, por nó, atribuída à classe de serviço. Se `query_working_mem` e `target_query_working_mem` são diferentes, uma transição dinâmica do WLM está em andamento. Um valor de `-1` indica que Auto WLM (WLM automático) está configurado.

Nome da coluna	Tipo de dados	Descrição
target_query_working_mem	integer	A quantidade de memória de trabalho, em MB por slot, por nó, definida pela alteração de configuração mais recente do WLM.
min_step_mem	integer	Reservada para uso do sistema.
name	character(64)	O nome da classe de serviços.
max_execution_time	bigint	O tempo, em número de milissegundos, durante o qual uma consulta pode ser executada antes de ser encerrada.
user_group_wild_card	Booleano	Se esta coluna for definida como TRUE, a fila do WLM tratará o asterisco (*) como um caractere curinga em strings de grupos de usuários na configuração do WLM.
query_group_wild_card	Booleano	Se esta coluna for definida como TRUE, a fila do WLM tratará o asterisco (*) como um caractere curinga em strings de grupos de consultas na configuração do WLM.
concurrency_scaling	character(20)	Descreve se a escalabilidade de simultaneidade é on ou off.
query_priority	character(20)	O valor da prioridade da consulta.
user_role_wild_card	Booleano	Se esta coluna for definida como TRUE, a fila do WLM tratará o asterisco (*) como um caractere curinga em strings de perfil de usuário na configuração do WLM.

Consulta de exemplo

A primeira classe de serviço definida pelo usuário é a classe 6, que é chamada de Classe de serviço #1. A consulta a seguir mostra a configuração atual das classes de serviço maiores do que 4. Para obter uma lista dos IDs de classe de serviço, consulte IDs da classe de serviço do WLM.

```
select rtrim(name) as name,
num_query_tasks as slots,
```

```
query_working_mem as mem,
max_execution_time as max_time,
user_group_wild_card as user_wildcard,
query_group_wild_card as query_wildcard
from stv_wlm_service_class_config
where service_class > 4;

name                          | slots | mem | max_time | user_wildcard | query_wildcard
------------------------------+-------+-----+----------+---------------+---------------
Service class for super user |     1 | 535 |        0 | false         | false
Queue 1                       |     5 | 125 |        0 | false         | false
Queue 2                       |     5 | 125 |        0 | false         | false
Queue 3                       |     5 | 125 |        0 | false         | false
Queue 4                       |     5 | 627 |        0 | false         | false
Queue 5                       |     5 | 125 |        0 | true          | true
Default queue                 |     5 | 125 |        0 | false         | false
```

A consulta a seguir mostra o status de uma transição de WLM dinâmica. Quando a transição está em andamento, `num_query_tasks` e `target_query_working_mem` são atualizados até que eles se igualem aos valores de destino. Para obter mais informações, consulte Propriedades de configuração dinâmicas e estáticas do WLM.

```
select rtrim(name) as name,
num_query_tasks as slots,
target_num_query_tasks as target_slots,
query_working_mem as memory,
target_query_working_mem as target_memory
from stv_wlm_service_class_config
where num_query_tasks > target_num_query_tasks
or query_working_mem > target_query_working_mem
and service_class > 5;

 name             | slots | target_slots | memory | target_mem
------------------+-------+--------------+--------+------------
 Queue 3          |     5 |           15 |    125 |       375
 Queue 5          |    10 |            5 |    250 |       125
 (2 rows)
```

STV_WLM_SERVICE_CLASS_STATE

Contém o estado atual das classes de serviço.

STV_WLM_SERVICE_CLASS_STATE é visível somente aos superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
service_class	integer	ID da classe de serviço. Para obter uma lista dos IDs de classe de serviço, consulte IDs da classe de serviço do WLM.
num_queued_queries	integer	O número de consultas na fila no momento.
num_executing_queries	integer	O número de consultas em execução no momento.
num_serviced_queries	integer	O número de consultas que já estiveram na classe de serviço.
num_executed_queries	integer	O número de consultas que foram executadas desde que o Amazon Redshift foi reiniciado.
num_evicted_queries	integer	O número de consultas removidas desde que o Amazon Redshift foi reiniciado. Alguns motivos para uma consulta removida incluem um tempo limite do WLM, uma ação de salto do QMR e uma consulta com falha em um cluster de escalabilidade da simultaneidade.
num_concurrency_scaling_queries	integer	Número de consultas executadas em um cluster de escalabilidade de simultaneidade desde que o Amazon Redshift foi reiniciado.

Consulta de exemplo

A consulta a seguir mostra o estado das classes de serviço maiores do que 5. Para obter uma lista dos IDs de classe de serviço, consulte IDs da classe de serviço do WLM.

```
select service_class, num_executing_queries,
```

```
num_executed_queries
from stv_wlm_service_class_state
where service_class > 5
order by service_class;
```

```
 service_class | num_executing_queries | num_executed_queries
---------------+-----------------------+----------------------
             6 |                     1 |                  222
             7 |                     0 |                  135
             8 |                     1 |                   39
(3 rows)
```

STV_XRESTORE_ALTER_QUEUE_STATE

Use STV_XRESTORE_ALTER_QUEUE_STATE para monitorar o andamento da migração de cada tabela durante um redimensionamento clássico. Isso se aplica especificamente quando o tipo de nó de destino é RG ou RA3. Para obter mais informações sobre o redimensionamento clássico para nós RG ou RA3, consulte Redimensionamento clássico.

STV_XRESTORE_ALTER_QUEUE_STATE é visível somente aos superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_RESTORE_STATE. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que iniciou o redimensionamento.
db_id	integer	O ID do banco de dados.
schema	char(128)	O nome do esquema.
table_name	char(128)	O nome da tabela.

Nome da coluna	Tipo de dados	Descrição
tbl	integer	O ID da tabela.
status	char(64)	O status do andamento da migração da tabela. Veja a seguir os valores possíveis. • `Waiting`: aguardando o início da redistribuição • `Applying`: realizando a redistribuição • `Finished`: redistribuição concluída
task_type	integer	O tipo de redistribuição da tabela. Veja a seguir os valores possíveis. • 1: KEY • 2: EVEN Para obter mais informações sobre estilos de distribuição, consulte Estilos de distribuição.

Consulta de exemplo

A consulta a seguir mostra o número de tabelas em um banco de dados que estão aguardando para serem redimensionadas, estão sendo redimensionadas no momento e concluíram o redimensionamento.

```
select db_id, status, count(*)
from stv_xrestore_alter_queue_state
group by 1,2 order by 3 desc

db_id  |   status    | count
-------+-------------+------
694325 | Waiting     |   323
694325 | Finished    |    60
694325 | Applying    |     1
```

Visualizações SVCS para o cluster principal e clusters de escalabilidade de simultaneidade

As visualizações do sistema SVCS com o prefixo SVCS fornecem detalhes sobre consultas no cluster principal e nos clusters de escalabilidade de simultaneidade. As visualizações são semelhantes às tabelas com o prefixo STL, exceto que as tabelas STL fornecem informações somente para consultas executadas no cluster principal.

Tópicos

- SVCS_ALERT_EVENT_LOG
- SVCS_COMPILE
- SVCS_CONCURRENCY_SCALING_USAGE
- SVCS_EXPLAIN
- SVCS_PLAN_INFO
- SVCS_QUERY_SUMMARY
- SVCS_S3LIST
- SVCS_S3LOG
- SVCS_S3PARTITION_SUMMARY
- SVCS_S3QUERY_SUMMARY
- SVCS_STREAM_SEGS
- SVCS_UNLOAD_LOG

SVCS_ALERT_EVENT_LOG

Registra um alerta quando o otimizador de consultas identifica as condições que podem indicar problemas de performance. Essa visualização é derivada da tabela do sistema STL_ALERT_EVENT_LOG, mas não mostra o nível de fatia para consultas executadas em um cluster de escalabilidade da simultaneidade. Use a tabela SVCS_ALERT_EVENT_LOG para identificar meios de melhorar a performance da consulta.

Uma consulta consiste em vários segmentos e cada segmento consiste em uma ou mais etapas. Para obter mais informações, consulte Processamento de consulta.

> **Note**
>
> Visualizações do sistema com o prefixo SVCS fornecem detalhes sobre consultas nos clusters principal e de escalabilidade da simultaneidade. As visualizações são semelhantes às tabelas com o prefixo STL, exceto que as tabelas STL fornecem informações somente para consultas executadas no cluster principal.

SVCS_ALERT_EVENT_LOG é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
segment	integer	O número que identifica o segmento da consulta.
etapa	integer	Etapa da consulta que foi executada.
pid	integer	O ID do processo associado à instrução e à fatia. A mesma consulta poderá ter vários PIDs, se for executada em várias fatias.
xid	bigint	O ID da transação associada à instrução.
event	character (1024)	A descrição do evento de alerta.
solução	character (1024)	A solução recomendada.

Nome da coluna	Tipo de dados	Descrição
event_time	timestamp	O horário (em UTC) de início da consulta. O tempo total inclui consultas e execução, com seis dígitos de precisão para segundos fracionários. Por exemplo: **`2009-06-12 11:29:19.131358`** .

Observações de uso

Você pode usar o SVCS_ALERT_EVENT_LOG para identificar problemas potenciais nas consultas, depois siga as práticas recomendadas em Ajustar o desempenho da consulta para otimizar o design do banco de dados e reescrever as consultas. A tabela SVCS_ALERT_EVENT_LOG registra os seguintes alertas:

- Ausência de estatísticas

  Não há estatísticas. Execute o ANALYZE depois de carregar dados ou de atualizações significativas e use o STATUPDATE com as operações COPY. Para obter mais informações, consulte Práticas recomendadas do Amazon Redshift para criar consultas.

- Loop aninhado

  Um loop aninhado é geralmente um produto cartesiano. Avalie sua consulta para garantir que todas as tabelas participantes sejam unidas de forma eficiente.

- Filtro muito seletivo

  A taxa de linhas retornadas em relação às linhas pesquisadas na varredura é menor do que 0.05. As linhas pesquisadas na varredura são representadas pelo valor de `rows_pre_user_filter` e as linhas retornadas são representadas pelo valor de STL_SCAN na tabela do sistema. Indica que a consulta está fazendo a varredura em um número excepcionalmente grande de linhas para determinar o conjunto de resultados. Isso pode ser causado pela falta ou por erros de chaves de classificação. Para obter mais informações, consulte Chaves de classificação.

- Excesso de linhas fantasma

  Uma varredura ignorou um número relativamente grande de linhas que estão marcadas como excluídas, mas não como limpadas, ou de linhas que foram inseridas, mas não confirmadas. Para obter mais informações, consulte Vacuum de tabelas.

- Grande distribuição

Mais de 1.000.000 de linhas foram redistribuídas para uma junção hash ou agregação. Para obter mais informações, consulte Distribuição de dados para otimização de consultas.

- Grande transmissão

  Mais de 1.000.000 de linhas foram transmitidas para uma junção hash. Para obter mais informações, consulte Distribuição de dados para otimização de consultas.

- Execução em série

  Um estilo de redistribuição DS_DIST_ALL_INNER foi indicado no plano de consulta, o que força uma execução em série, pois toda a tabela interna foi redistribuída para um único nó. Para obter mais informações, consulte Distribuição de dados para otimização de consultas.

Consultas de exemplo

As consultas a seguir mostram eventos de alerta para quatro consultas.

```
SELECT query, substring(event,0,25) as event,
substring(solution,0,25) as solution,
trim(event_time) as event_time from svcs_alert_event_log order by query;

 query |             event             |          solution            |     event_time

-------+-------------------------------+-------------------------------
+---------------------
  6567 | Missing query planner statist | Run the ANALYZE command      | 2014-01-03
 18:20:58
  7450 | Scanned a large number of del | Run the VACUUM command to rec| 2014-01-03
 21:19:31
  8406 | Nested Loop Join in the query | Review the join predicates to| 2014-01-04
 00:34:22
 29512 | Very selective query filter:r | Review the choice of sort key| 2014-01-06
 22:00:00

(4 rows)
```

SVCS_COMPILE

Registra o local e o tempo de compilação para cada segmento de consulta das consultas, incluindo aquelas executadas em um cluster de escalabilidade da simultaneidade e no cluster principal.

> **Note**
>
> Visualizações do sistema com o prefixo SVCS fornecem detalhes sobre consultas nos clusters principal e de escalabilidade da simultaneidade. As visualizações são semelhantes às visualizações com o prefixo SVL, exceto que as visualizações SVL fornecem informações somente para consultas executadas no cluster principal.

A exibição SVCS_COMPILE é visível por todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Para obter informações sobre SCL_COMPILE, consulte SVL_COMPILE.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
xid	bigint	O ID da transação associada à instrução.
pid	integer	O ID do processo associado à instrução.
consultar	integer	O ID da consulta. Este ID pode ser usado para unir várias outras tabelas e exibições do sistema.
segment	integer	O segmento de consulta a ser compilado.
locus	integer	O local onde o segmento é executado: **1** se for em um nó de computação e **2** se for no nó líder.
starttime	timestamp	O horário, no Tempo Universal Coordenado (UTC), do início da compilação.
endtime	timestamp	O horário, em UTC, do término da compilação.
compile	integer	Um valor que é **0** se a compilação foi reutilizada e **1** se o segmento foi compilado.

Consultas de exemplo

Neste exemplo, as consultas 35878 e 35879 executaram a mesma instrução SQL. A coluna de compilação da consulta 35878 mostra o valor 1 para quatro segmentos da consulta, indicando que os segmentos foram compilados. A consulta 35879 mostra o valor `0` na coluna de compilação para todos os segmentos, indicando que os segmentos não precisaram ser compilados novamente.

```
select userid, xid,  pid, query, segment, locus,
datediff(ms, starttime, endtime) as duration, compile
from svcs_compile
where query = 35878 or query = 35879
order by query, segment;

 userid |  xid   |  pid  | query | segment | locus | duration | compile
--------+--------+-------+-------+---------+-------+----------+---------
    100 | 112780 | 23028 | 35878 |       0 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       1 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       2 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       3 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       4 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       5 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       6 |     1 |     1380 |       1
    100 | 112780 | 23028 | 35878 |       7 |     1 |     1085 |       1
    100 | 112780 | 23028 | 35878 |       8 |     1 |     1197 |       1
    100 | 112780 | 23028 | 35878 |       9 |     2 |      905 |       1
    100 | 112782 | 23028 | 35879 |       0 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       1 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       2 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       3 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       4 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       5 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       6 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       7 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       8 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       9 |     2 |        0 |       0
(20 rows)
```

SVCS_CONCURRENCY_SCALING_USAGE

Registra os períodos de uso para escalabilidade da simultaneidade. Cada período de uso é uma duração consecutiva em que um cluster individual de escalabilidade da simultaneidade está processando consultas ativamente.

SVCS_CONCURRENCY_SCALING_USAGE Esta tabela permanece visível para superusuários. O superusuário do banco de dados pode optar por abri-la para todos os usuários.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
start_time	time stamp sem fuso horário	Quando o período de uso é começa.
end_time	time stamp sem fuso horário	Quando o período de uso termina.
queries	bigint	Número de consultas executadas durante esse período de uso.
usage_in_seconds	numeric(27,0)	Total de segundos nesse período de uso.

Consultas de exemplo

Para visualizar a duração do uso em segundos para um período especificado, insira a consulta a seguir:

```
select * from svcs_concurrency_scaling_usage order by start_time;

start_time | end_time | queries | usage_in_seconds
---------------------------+---------------------------+---------+-----------------
2019-02-14 18:43:53.01063 | 2019-02-14 19:16:49.781649 | 48 | 1977
```

SVCS_EXPLAIN

Exibe o plano EXPLAIN para uma consulta que foi enviada para execução.

> **Note**
>
> Visualizações do sistema com o prefixo SVCS fornecem detalhes sobre consultas nos clusters principal e de escalabilidade da simultaneidade. As visualizações são semelhantes às tabelas com o prefixo STL, exceto que as tabelas STL fornecem informações somente para consultas executadas no cluster principal.

SVCS_EXPLAIN permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
nodeid	integer	O identificador do nó de plano, onde o nó mapeia para uma ou mais etapas na execução da consulta.
parentid	integer	O identificador do nó de plano para um nó pai. Um nó pai tem um certo número de nós filho. Por exemplo, uma junção de mesclagem é o pai das varreduras em tabelas unidas.
plannode	character(400)	O texto do nó de saída de EXPLAIN. Os nós de plano que se referem à execução em nós de computação recebem o prefixo **XN** na saída EXPLAIN.
info	character(400)	As informações sobre o qualificador e o filtro do nó de plano. Por exemplo, as condições de junção e as restrições da cláusula WHERE estão incluídas nesta coluna.

Consultas de exemplo

Considere a seguinte saída de EXPLAIN para uma consulta de junção agregada:

```
explain select avg(datediff(day, listtime, saletime)) as avgwait
from sales, listing where sales.listid = listing.listid;
                                QUERY PLAN

------------------------------------------------------------------------------
 XN Aggregate  (cost=6350.30..6350.31 rows=1 width=16)
  ->  XN Hash Join DS_DIST_NONE  (cost=47.08..6340.89 rows=3766 width=16)
        Hash Cond: ("outer".listid = "inner".listid)
        -> XN Seq Scan on listing  (cost=0.00..1924.97 rows=192497 width=12)
        -> XN Hash  (cost=37.66..37.66 rows=3766 width=12)
              -> XN Seq Scan on sales  (cost=0.00..37.66 rows=3766 width=12)
(6 rows)
```

Se você executar essa consulta e o ID de consulta for 10, use a tabela SVCS_EXPLAIN para visualizar os mesmos tipos de informações que o comando EXPLAIN retorna:

```
select query,nodeid,parentid,substring(plannode from 1 for 30),
substring(info from 1 for 20) from svcs_explain
where query=10 order by 1,2;

query| nodeid |parentid|           substring            |    substring
-----+--------+--------+--------------------------------+-------------------
10   |      1 |      0 |XN Aggregate  (cost=6717.61..6  |
10   |      2 |      1 |  -> XN Merge Join DS_DIST_NO| Merge Cond:("outer"
10   |      3 |      2 |       -> XN Seq Scan on lis |
10   |      4 |      2 |       -> XN Seq Scan on sal |
(4 rows)
```

Considere a seguinte consulta:

```
select event.eventid, sum(pricepaid)
from event, sales
where event.eventid=sales.eventid
group by event.eventid order by 2 desc;

eventid |   sum
--------+----------
    289 | 51846.00
```

```
   7895 | 51049.00
   1602 | 50301.00
    851 | 49956.00
   7315 | 49823.00
...
```

Se o ID da consulta é 15, a consulta de tabela do sistema a seguir retorna os nós de plano que foram executados. Nesse caso, a ordem dos nós é invertida para mostrar a ordem real de execução:

```
select query,nodeid,parentid,substring(plannode from 1 for 56)
from svcs_explain where query=15 order by 1, 2 desc;

query|nodeid|parentid|                            substring
-----+------+--------+-----------------------------------------------------------
15   |    8 |      7 |                                  -> XN Seq Scan on eve
15   |    7 |      5 |                            -> XN Hash(cost=87.98..87.9
15   |    6 |      5 |                            -> XN Seq Scan on sales(cos
15   |    5 |      4 |                       -> XN Hash Join DS_DIST_OUTER(cos
15   |    4 |      3 |                 -> XN HashAggregate(cost=862286577.07..
15   |    3 |      2 |           -> XN Sort(cost=1000862287175.47..10008622871
15   |    2 |      1 |  -> XN Network(cost=1000862287175.47..1000862287197.
15   |    1 |      0 |XN Merge(cost=1000862287175.47..1000862287197.46 rows=87
(8 rows)
```

A consulta a seguir recupera os IDs de consulta de todos os planos de consulta que contêm uma função de janela:

```
select query, trim(plannode) from svcs_explain
where plannode like '%Window%';

query|                                    btrim
-----+-------------------------------------------------------------------------
26   | -> XN Window(cost=1000985348268.57..1000985351256.98 rows=170 width=33)
27   | -> XN Window(cost=1000985348268.57..1000985351256.98 rows=170 width=33)
(2 rows)
```

SVCS_PLAN_INFO

Use a tabela SVCS_PLAN_INFO para ver a saída EXPLAIN para uma consulta em termos de um conjunto de linhas. Essa é uma forma alternativa de ver os planos de consulta.

> **Note**
>
> Visualizações do sistema com o prefixo SVCS fornecem detalhes sobre consultas nos clusters principal e de escalabilidade da simultaneidade. As visualizações são semelhantes às tabelas com o prefixo STL, exceto que as tabelas STL fornecem informações somente para consultas executadas no cluster principal.

SVCS_PLAN_INFO permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
nodeid	integer	O identificador do nó de plano, onde o nó mapeia para uma ou mais etapas na execução da consulta.
segment	integer	O número que identifica o segmento da consulta.
etapa	integer	O número que identifica a etapa da consulta.
locus	integer	Local onde a etapa é executada: 0 se estiver em um nó de computação e 1 se estiver no nó líder.
plannode	integer	O valor enumerado do nó de plano. Veja a tabela a seguir com os valores enumerados dos nós de plano. (A coluna PLANNODE na tabela SVCS_EXPLAIN contém o texto do nó de plano).
startupcost	double precision	O custo relativo estimado de retorno da primeira linha para esta etapa.

Nome da coluna	Tipo de dados	Descrição
totalcost	double precision	O custo relativo estimado de execução da etapa.
rows	bigint	O número estimado de linhas que serão produzidas pela etapa.
bytes	bigint	O número estimado de bytes que serão produzidos pela etapa.

Consultas de exemplo

Os exemplos a seguir comparam os planos de consulta retornados de uma consulta SELECT simples usando o comando EXPLAIN e consultando a tabela SVCS_PLAN_INFO.

```
explain select * from category;
QUERY PLAN
------------------------------------------------------------
XN Seq Scan on category (cost=0.00..0.11 rows=11 width=49)
(1 row)

select * from category;
catid | catgroup | catname | catdesc
-------+----------+-----------+--------------------------------------------
1 | Sports | MLB | Major League Baseball
3 | Sports | NFL | National Football League
5 | Sports | MLS | Major League Soccer
...

select * from svcs_plan_info where query=256;

query | nodeid | segment | step | locus | plannode | startupcost | totalcost
| rows | bytes
-------+--------+---------+------+-------+----------+-------------+-----------+------
+-------
256 | 1 | 0 | 1 | 0 | 104 | 0 | 0.11 | 11 | 539
256 | 1 | 0 | 0 | 0 | 104 | 0 | 0.11 | 11 | 539
(2 rows)
```

Neste exemplo, o PLANNODE 104 refere-se à varredura sequencial da tabela CATEGORY.

```
select distinct eventname from event order by 1;

eventname
----------------------------------------------------------------------------
.38 Special
3 Doors Down
70s Soul Jam
A Bronx Tale
...

explain select distinct eventname from event order by 1;

QUERY PLAN
------------------------------------------------------------------------------------------
XN Merge (cost=1000000000136.38..1000000000137.82 rows=576 width=17)
Merge Key: eventname
-> XN Network (cost=1000000000136.38..1000000000137.82 rows=576
width=17)
Send to leader
-> XN Sort (cost=1000000000136.38..1000000000137.82 rows=576
width=17)
Sort Key: eventname
-> XN Unique (cost=0.00..109.98 rows=576 width=17)
-> XN Seq Scan on event (cost=0.00..87.98 rows=8798
width=17)
(8 rows)

select * from svcs_plan_info where query=240 order by nodeid desc;

query | nodeid | segment | step | locus | plannode | startupcost |
totalcost | rows | bytes
-------+--------+---------+------+-------+----------+-----------------
+------------------+------+--------
240 | 5 | 0 | 0 | 0 | 104 | 0               | 87.98   | 8798 | 149566
240 | 5 | 0 | 1 | 0 | 104 | 0               | 87.98   | 8798 | 149566
240 | 4 | 0 | 2 | 0 | 117 | 0               | 109.975 | 576  | 9792
240 | 4 | 0 | 3 | 0 | 117 | 0               | 109.975 | 576  | 9792
240 | 4 | 1 | 0 | 0 | 117 | 0               | 109.975 | 576  | 9792
240 | 4 | 1 | 1 | 0 | 117 | 0               | 109.975 | 576  | 9792
240 | 3 | 1 | 2 | 0 | 114 | 1000000000136.38 | 1000000000137.82 | 576 | 9792
240 | 3 | 2 | 0 | 0 | 114 | 1000000000136.38 | 1000000000137.82 | 576 | 9792
240 | 2 | 2 | 1 | 0 | 123 | 1000000000136.38 | 1000000000137.82 | 576 | 9792
240 | 1 | 3 | 0 | 0 | 122 | 1000000000136.38 | 1000000000137.82 | 576 | 9792
```

```
(10 rows)
```

SVCS_QUERY_SUMMARY

Use a exibição SVCS_QUERY_SUMMARY para encontrar informações gerais sobre a execução de uma consulta.

Observe que as informações em SVCS_QUERY_SUMMARY são agregadas de todos os nós.

> **Note**
>
> A visualização SVCS_QUERY_SUMMARY contém apenas informações sobre as consultas concluídas pelo Amazon Redshift, e não por outros comandos de utilitário e DDL. Para obter uma lista completa e informações sobre todas as instruções concluídas pelo Amazon Redshift, inclusive comandos de DDL e utilitário, você pode consultar a visualização SVL_STATEMENTTEXT.
> Visualizações do sistema com o prefixo SVCS fornecem detalhes sobre consultas nos clusters principal e de escalabilidade da simultaneidade. As visualizações são semelhantes às visualizações com o prefixo SVL, exceto que as visualizações SVL fornecem informações somente para consultas executadas no cluster principal.

SVCS_QUERY_SUMMARY é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Para obter informações sobre SVL_QUERY_SUMMARY, consulte SVL_QUERY_SUMMARY.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.

Nome da coluna	Tipo de dados	Descrição
consultar	integer	ID da consulta. Ele pode ser usado para unir várias outras tabelas e exibições do sistema.
stm	integer	Stream: um conjunto de segmentos simultâneos em uma consulta. Uma consulta tem um ou mais streams.
seg	integer	O número do segmento. Uma consulta consiste em vários segmentos e cada segmento consiste em uma ou mais etapas. Os segmentos de uma consulta podem ser executados em paralelo. Cada segmento executa em um único processo.
etapa	integer	Etapa da consulta que foi executada.
maxtime	bigint	O tempo máximo de execução para esta etapa (em microsseg undos).
avgtime	bigint	O tempo médio de execução para esta etapa (em microsseg undos).
rows	bigint	O número de linhas de dados envolvidas na etapa da consulta.
bytes	bigint	O número de bytes de dados envolvidos na etapa da consulta.
rate_row	double precision	A taxa de execução da consulta por linha.
rate_byte	double precision	A taxa de execução da consulta por byte.
rótulo	texto	O rótulo da etapa, que consiste no nome da etapa da consulta e, se for aplicável, no ID e no nome da tabela (por exemplo, scan tbl=100448 name =user). Os IDs de tabela com três dígitos geralmente indicam varreduras de tabelas transitórias. Quando você vê `tbl=0`, isso normalmente indica uma varredura de um valor constante.

Nome da coluna	Tipo de dados	Descrição
is_diskbased	character(1)	Indica se esta etapa da consulta foi realizada como uma operação em disco em qualquer nó do cluster: true (**t**) ou false (**f**). Somente algumas etapas, como hash, classificação e etapas de agregação podem ir para o disco. Muitos tipos de etapas são sempre executadas na memória.
workmem	bigint	A quantidade de memória de trabalho (em bytes) atribuída à etapa da consulta.
is_rrscan	character(1)	O valor true (**t**) indica que a varredura restrita ao intervalo foi utilizada na etapa. O padrão é falso (**f**).
is_delayed_scan	character(1)	O valor true (**t**) indica que a varredura com atraso foi utilizada na etapa. O padrão é false (**f**).
rows_pre_filter	bigint	Para varreduras de tabelas permanentes, o número total de linhas enviadas antes da filtragem das linhas marcadas para exclusão (linhas fantasma).

Consultas de exemplo

Visualização de informações de processamento para uma etapa da consulta

A consulta a seguir mostra as informações básicas de processamento para cada etapa da consulta 87:

```
select query, stm, seg, step, rows, bytes
from svcs_query_summary
where query = 87
order by query, seg, step;
```

Esta consulta recupera as informações do processamento da consulta 87, como mostram os seguintes dados de saída de exemplo:

```
 query | stm | seg | step |  rows  |  bytes
-------+-----+-----+------+--------+---------
```

```
87     |   0 |   0 |    0 |     90 |    1890
87     |   0 |   0 |    2 |     90 |     360
87     |   0 |   1 |    0 |     90 |     360
87     |   0 |   1 |    2 |     90 |    1440
87     |   1 |   2 |    0 | 210494 | 4209880
87     |   1 |   2 |    3 |  89500 |       0
87     |   1 |   2 |    6 |      4 |      96
87     |   2 |   3 |    0 |      4 |      96
87     |   2 |   3 |    1 |      4 |      96
87     |   2 |   4 |    0 |      4 |      96
87     |   2 |   4 |    1 |      1 |      24
87     |   3 |   5 |    0 |      1 |      24
87     |   3 |   5 |    4 |      0 |       0
(13 rows)
```

Determinar se houve transbordamento das etapas de consulta para o disco

A consulta a seguir mostra se houve transbordamento para o disco em alguma das etapas da consulta de ID 1025 (veja a exibição SVL_QLOG para saber como obter o ID de uma consulta) ou se a consulta foi totalmente executada na memória:

```
select query, step, rows, workmem, label, is_diskbased
from svcs_query_summary
where query = 1025
order by workmem desc;
```

Essa consulta retorna os seguintes dados de saída de exemplo:

```
query| step|  rows  |  workmem   |  label        | is_diskbased
-----+-----+--------+-----------+---------------+--------------
1025 |  0  |16000000|  141557760 |scan tbl=9     | f
1025 |  2  |16000000|  135266304 |hash tbl=142   | t
1025 |  0  |16000000|  128974848 |scan tbl=116536| f
1025 |  2  |16000000|  122683392 |dist           | f
(4 rows)
```

Ao fazer a varredura dos valores pelo IS_DISKBASED, você pode ver quais etapas de consulta foram executadas no disco. Para a consulta 1025, a etapa de hash foi executada em disco. As etapas que podem ser executadas em disco incluem a hash, a aggr e as etapas de classificação. Para visualizar apenas a etapas da consulta que são baseadas em disco, adicione a cláusula **`and is_diskbased = 't'`** na instrução de SQL no exemplo acima.

SVCS_S3LIST

Use a visualização SVCS_S3LIST para obter detalhes sobre as consultas do Amazon Redshift Spectrum no nível do segmento. Um segmento pode executar uma varredura de tabela externa. Essa visualização é derivada da visualização do sistema SVL_S3LIST, mas não mostra o nível de fatia para consultas executadas em um cluster de escalabilidade da simultaneidade.

> **Note**
>
> Visualizações do sistema com o prefixo SVCS fornecem detalhes sobre consultas nos clusters principal e de escalabilidade da simultaneidade. As visualizações são semelhantes às visualizações com o prefixo SVL, exceto que as visualizações SVL fornecem informações somente para consultas executadas no cluster principal.

SVCS_S3LIST é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Para obter informações sobre SVL_S3LIST, consulte SVL_S3LIST.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
consultar	integer	O ID da consulta.
segment	integer	O número do segmento. Uma consulta consiste em vários segmentos.
nó	integer	O número de nós.
eventtime	timestamp	O horário em UTC em que o evento é gravado.
bucket	char(256)	O nome do bucket do Amazon S3.
prefix	char(256)	O prefixo do local do bucket do Amazon S3.
recursive	char(1)	Se há varredura recursiva para subpastas.

Nome da coluna	Tipo de dados	Descrição
retrieved_files	integer	O número de arquivos listados.
max_file_size	bigint	O tamanho máximo de arquivo entre os arquivos listados.
avg_file_size	double precision	O tamanho médio de arquivo entre os arquivos listados.
generated_splits	integer	O número de divisões de arquivos.
avg_split_length	double precision	O tamanho médio das divisões de arquivos em bytes.
duration	bigint	A duração da listagem, em microssegundos.

Consulta de exemplo

O exemplo a seguir consulta SVCS_S3LIST para verificar a última consulta realizada.

```
select *
from svcs_s3list
where query = pg_last_query_id()
order by query,segment;
```

SVCS_S3LOG

Use a visualização SVCS_S3LOG para obter detalhes de solução de problemas sobre as consultas de data lake no nível de segmento. Um segmento pode executar uma varredura de tabela externa. Essa visualização é derivada da visualização do sistema SVL_S3LOG, mas não mostra o nível de fatia para consultas executadas em um cluster de escalabilidade da simultaneidade.

> **Note**
>
> Visualizações do sistema com o prefixo SVCS fornecem detalhes sobre consultas nos clusters principal e de escalabilidade da simultaneidade. As visualizações são semelhantes às visualizações com o prefixo SVL, exceto que as visualizações SVL fornecem informações somente para consultas executadas no cluster principal.

SVCS_S3LOG é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Para obter informações sobre SVL_S3LOG, consulte SVL_S3LOG.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
pid	integer	O ID do processo.
consultar	integer	O ID da consulta.
segment	integer	O número do segmento. Uma consulta consiste em vários segmentos e cada segmento consiste em uma ou mais etapas.
etapa	integer	A etapa da consulta que foi executada.
nó	integer	O número de nós.
eventtime	timestamp	O horário em UTC em que o evento é gravado.
message	char(512)	A mensagem da entrada de log.

Consulta de exemplo

O exemplo a seguir consulta SVCS_S3LOG para verificar a última consulta executada.

```
select *
```

```
from svcs_s3log
where query = pg_last_query_id()
order by query,segment;
```

SVCS_S3PARTITION_SUMMARY

Use a visualização SVCS_S3PARTITION_SUMMARY para obter um resumo do processamento da partição de consultas do Redshift Spectrum no nível do segmento. Um segmento pode executar uma varredura de tabela externa.

> **Note**
>
> Visualizações do sistema com o prefixo SVCS fornecem detalhes sobre consultas nos clusters principal e de escalabilidade da simultaneidade. As visualizações são semelhantes às visualizações com o prefixo SVL, exceto que as visualizações SVL fornecem informações somente para consultas executadas no cluster principal.

SVCS_S3PARTITION_SUMMARY é visível por todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Para obter informações sobre SVL_S3PARTITION, consulte SVL_S3PARTITION.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
consultar	integer	O ID da consulta. Este valor pode ser usado para unir várias outras tabelas e exibições do sistema.
segment	integer	O número do segmento. Uma consulta consiste em vários segmentos.
assignment	char(1)	O tipo de atribuição de partição entre os nós.
min_starttime	timestamp	O horário em UTC em que o processamento de partição foi iniciado.

Nome da coluna	Tipo de dados	Descrição
max_endtime	timestamp	O horário em UTC em que o processamento de partição foi concluído.
min_duration	bigint	O tempo mínimo de processamento de partição usado por um nó para esta consulta (em microssegundos).
max_duration	bigint	O tempo máximo de processamento de partição usado por um nó para esta consulta (em microssegundos).
avg_duration	bigint	O tempo médio de processamento de partição usado por um nó para esta consulta (em microssegundos).
total_partitions	integer	O número total de partições em uma tabela externa.
qualified_partitions	integer	O número total de partições qualificadas.
min_assigned_partitions	integer	O número mínimo de partições atribuídas em um nó.
max_assigned_partitions	integer	O número máximo de partições atribuídas em um nó.
avg_assigned_partitions	bigint	O número médio de partições atribuídas em um nó.

Consulta de exemplo

O exemplo a seguir obtém os detalhes da varredura de partições da última consulta realizada.

```
select query, segment, assignment, min_starttime, max_endtime, min_duration,
 avg_duration
```

```
from svcs_s3partition_summary
where query = pg_last_query_id()
order by query,segment;
```

SVCS_S3QUERY_SUMMARY

Use a visualização SVCS_S3QUERY_SUMMARY para obter um resumo de todas as consultas de data lake que foram executadas no sistema. Um segmento pode executar uma varredura de tabela externa.

> **Note**
>
> Visualizações do sistema com o prefixo SVCS fornecem detalhes sobre consultas nos clusters principal e de escalabilidade da simultaneidade. As visualizações são semelhantes às visualizações com o prefixo SVL, exceto que as visualizações SVL fornecem informações somente para consultas executadas no cluster principal.

SVCS_S3QUERY_SUMMARY é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Para obter informações sobre SVL_S3QUERY, consulte SVL_S3QUERY.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou determinada entrada.
consultar	integer	O ID da consulta. Este valor pode ser usado para unir várias outras tabelas e exibições do sistema.
xid	bigint	O ID da transação.
pid	integer	O ID do processo.

Nome da coluna	Tipo de dados	Descrição
segment	integer	O número do segmento. Uma consulta consiste em vários segmentos e cada segmento consiste em uma ou mais etapas.
etapa	integer	A etapa da consulta que foi executada.
starttime	timestamp	A hora em UTC em que a consulta do Redshift Spectrum neste segmento começou a ser executada. Um segmento pode ter uma varredura de tabela externa.
endtime	timestamp	A hora em UTC que a consulta do Redshift Spectrum neste segmento foi concluída. Um segmento pode ter uma varredura de tabela externa.
elapsed	integer	O tempo que a consulta do Redshift Spectrum levou para ser executada neste segmento (em microssegundos).
aborted	integer	Se uma consulta for interrompida pelo sistema ou cancelada pelo usuário, essa coluna terá o valor **1**. Se a consulta foi executada até o final, essa coluna conterá **0**.
external_table_name	char(136)	O formato interno do nome externo da tabela para a varredura da tabela externa.
file_format	character(16)	O formato de arquivo dos dados da tabela externa.
is_partitioned	char(1)	O valor true (**t**) indica que a tabela externa está particionada.
is_rrscan	char(1)	O valor true (**t**) indica que uma varredura restrita ao intervalo foi aplicada.
is_nested	varchar(1)	O valor true (**t**) indica que o tipo de dados da coluna aninhada é acessado.

Nome da coluna	Tipo de dados	Descrição
s3_scanned_rows	bigint	O número de linhas digitalizadas do Amazon S3 e enviadas para a camada do Redshift Spectrum. Em clusters RG provisionados, esse é o número total de linhas digitalizadas diretamente do Amazon S3 pelo leitor nativo do cluster em todas as fatias, antes do envio do filtro.
s3_scanned_bytes	bigint	O número de bytes verificados do Amazon S3 e enviados para a camada do Redshift Spectrum, com base em dados compactados. Em clusters RG provisionados, esse é o número total de bytes digitalizados diretamente do Amazon S3 pelo leitor nativo do cluster em todas as fatias, antes do envio do filtro.
s3query_returned_rows	bigint	O número de linhas retornadas da camada do Redshift Spectrum para o cluster. Em clusters RG provisionados, esse é o número total de linhas retornadas pelo leitor nativo do cluster em todas as fatias, antes do envio do filtro.
s3query_returned_bytes	bigint	O número de bytes retornados da camada do Redshift Spectrum para o cluster. Uma grande quantidade de dados retornados ao Amazon Redshift pode afetar a performance do sistema. Em clusters RG provisionados, esse é o número total de bytes retornados pelo leitor nativo do cluster em todas as fatias, antes do envio do filtro.
files	integer	O número de arquivos que foram processados para esta consulta do data lake. Um número pequeno de arquivos limita os benefícios do processamento paralelo.
files_max	integer	O número máximo de arquivos processados em uma fatia.
files_avg	integer	O número médio de arquivos processados em uma fatia.

Nome da coluna	Tipo de dados	Descrição
splits	bigint	O número de divisões processadas para este segmento. O número de divisões processadas nesta fatia. Com arquivos de dados grandes que podem ser divididos, por exemplo, arquivos de dados maiores do que 512 MB, o Redshift Spectrum tenta dividir os arquivos em várias solicitações do S3 para o processamento paralelo.
splits_max	integer	O número máximo de divisões processadas nesta fatia.
splits_avg	bigint	O número médio de divisões processadas nesta fatia.
total_split_size	bigint	O tamanho total de todas as divisões processadas.
max_split_size	bigint	O tamanho máximo da divisão processada, em bytes.
avg_split_size	bigint	O tamanho médio da divisão processada, em bytes.
total_retries	bigint	O número total de novas tentativas para a consulta do Redshift Spectrum neste segmento. Em clusters do tipo de nó RG, essa coluna está obsoleta e contém -1. Para novas tentativas do cliente do Amazon S3 no RG, consulte `STL_S3CLIENT` .
max_retries	integer	O número máximo de novas tentativas para um arquivo individual processado. Em clusters do tipo de nó RG, essa coluna está obsoleta e contém -1. Para novas tentativas do cliente do Amazon S3 no RG, consulte `STL_S3CLIENT` .
max_request_duration	bigint	A duração máxima de uma solicitação de arquivo individual (em microssegundos). As consultas de longa duração podem indicar um gargalo. Em clusters do tipo de nó RG, essa coluna está obsoleta e contém -1.

Nome da coluna	Tipo de dados	Descrição
avg_request_duration	bigint	A duração média das solicitações de arquivos (em microssegundos). Em clusters do tipo de nó RG, essa coluna está obsoleta e contém -1.
max_request_parallelism	integer	O número máximo de solicitações paralelas em uma fatia para esta consulta do Redshift Spectrum. Em clusters do tipo de nó RG, essa coluna está obsoleta e contém -1.
avg_request_parallelism	double precision	O número médio de solicitações paralelas em uma fatia para esta consulta do Redshift Spectrum. Em clusters do tipo de nó RG, essa coluna está obsoleta e contém -1.
total_slowdown_count	bigint	O número total de solicitações do Amazon S3 com um erro de desaceleração ocorrido durante a varredura da tabela externa. Em clusters do tipo de nó RG, essa coluna está obsoleta e contém -1.
max_slowdown_count	integer	O número máximo de solicitações do Amazon S3 com um erro de desaceleração que ocorreu durante a varredura de tabela externa em um slice. Em clusters do tipo de nó RG, essa coluna está obsoleta e contém -1.

Consulta de exemplo

O exemplo a seguir obtém os detalhes da etapa de varredura da última consulta executada.

```
select query, segment, elapsed, s3_scanned_rows, s3_scanned_bytes,
 s3query_returned_rows, s3query_returned_bytes, files
from svcs_s3query_summary
where query = pg_last_query_id()
order by query,segment;
```

```
query | segment | elapsed | s3_scanned_rows | s3_scanned_bytes | s3query_returned_rows
 | s3query_returned_bytes | files
------+---------+---------+-----------------+------------------+----------------------
+------------------------+------
```

```
4587 |       2 |   67811 |               0 |               0 |                    0
|                    0 |     0
4587 |       2 |  591568 |          172462 |          11260097 |                 8513
|                170260 |     1
4587 |       2 |  216849 |               0 |               0 |                    0
|                    0 |     0
4587 |       2 |  216671 |               0 |               0 |                    0
|                    0 |     0
```

SVCS_STREAM_SEGS

Lista a relação entre os streams e os segmentos simultâneos.

> **Note**
>
> Visualizações do sistema com o prefixo SVCS fornecem detalhes sobre consultas nos clusters principal e de escalabilidade da simultaneidade. As visualizações são semelhantes às tabelas com o prefixo STL, exceto que as tabelas STL fornecem informações somente para consultas executadas no cluster principal.

SVCS_STREAM_SEGS permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
transmissão	integer	O conjunto de segmentos simultâneos de uma consulta.
segment	integer	O número que identifica o segmento da consulta.

Consultas de exemplo

Para visualizar a relação entre streams e segmentos simultâneos para a consulta mais recente, digite a seguinte consulta:

```
select *
from svcs_stream_segs
where query = pg_last_query_id();

 query | stream | segment
-------+--------+---------
    10 |      1 |       2
    10 |      0 |       0
    10 |      2 |       4
    10 |      1 |       3
    10 |      0 |       1
(5 rows)
```

SVCS_UNLOAD_LOG

Use SVCS_UNLOAD_LOG para obter detalhes de operações UNLOAD.

SVCS_UNLOAD_LOG registra uma linha para cada arquivo criado por uma instrução UNLOAD. Por exemplo, se uma operação UNLOAD criar 12 arquivos, SVCS_UNLOAD_LOG conterá 12 linhas correspondentes. Essa visualização é derivada da tabela do sistema STL_UNLOAD_LOG, mas não mostra o nível de fatia para consultas executadas em um cluster de escalabilidade da simultaneidade.

> **Note**
>
> Visualizações do sistema com o prefixo SVCS fornecem detalhes sobre consultas nos clusters principal e de escalabilidade da simultaneidade. As visualizações são semelhantes às tabelas com o prefixo STL, exceto que as tabelas STL fornecem informações somente para consultas executadas no cluster principal.

SVCS_UNLOAD_LOG permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	O ID da consulta.
pid	integer	O ID do processo associado à instrução da consulta.
caminho	character(1280)	O caminho completo do objeto Amazon S3 para o arquivo.
start_time	timestamp	O horário de início da operação UNLOAD.
end_time	timestamp	O horário de término da operação UNLOAD.
line_count	bigint	O número de linhas descarregadas no arquivo.
transfer_size	bigint	O número de bytes transferidos.
file_format	character(10)	O formato do arquivo não carregado.

Consulta de exemplo

Para obter uma lista dos arquivos que foram gravados no Amazon S3 por um comando UNLOAD, você pode chamar uma operação de lista do Amazon S3 após a conclusão do UNLOAD; no entanto, dependendo da rapidez com que você emite a chamada, a lista pode estar incompleta porque uma operação de lista do Amazon S3 é eventualmente consistente. Para obter uma lista completa e confiável imediatamente, consulte SVCS_UNLOAD_LOG.

A consulta a seguir retorna o nome do caminho para os arquivos que foram criados por uma operação UNLOAD para a última consulta concluída:

```
select query, substring(path,0,40) as path
from svcs_unload_log
where query = pg_last_query_id()
order by path;
```

Este comando retorna a seguinte saída de exemplo:

```
 query |             path
 ------+-----------------------------------------
  2320 | s3://amzn-s3-demo-bucket/venue0000_part_00
  2320 | s3://amzn-s3-demo-bucket/venue0001_part_00
  2320 | s3://amzn-s3-demo-bucket/venue0002_part_00
  2320 | s3://amzn-s3-demo-bucket/venue0003_part_00
(4 rows)
```

Visualizações SVL para o cluster principal

As visualizações SVL são visualizações do sistema no Amazon Redshift que contêm referências a tabelas e logs STL para informações mais detalhadas.

Essas exibições fornecem um acesso mais rápido e mais fácil aos dados que são consultados com frequência nessas tabelas.

> **Note**
>
> A visualização SVL_QUERY_SUMMARY contém apenas informações sobre as consultas executadas pelo Amazon Redshift, e não por outros comandos de utilitário e DDL.
> Para obter uma lista completa e informações sobre todas as declarações executadas pelo Amazon Redshift, incluindo comandos de DDL e utilitário, consulte a visualização SVL_STATEMENTTEXT

Tópicos

- SVL_AUTO_WORKER_ACTION
- SVL_COMPILE
- SVL_DATASHARE_CHANGE_LOG
- SVL_DATASHARE_CROSS_REGION_USAGE
- SVL_DATASHARE_USAGE_CONSUMER
- SVL_DATASHARE_USAGE_PRODUCER
- SVL_FEDERATED_QUERY
- SVL_MULTI_STATEMENT_VIOLATIONS

- SVL_MV_REFRESH_STATUS
- SVL_QERROR
- SVL_QLOG
- SVL_QUERY_METRICS
- SVL_QUERY_METRICS_SUMMARY
- SVL_QUERY_QUEUE_INFO
- SVL_QUERY_REPORT
- SVL_QUERY_SUMMARY
- SVL_RESTORE_ALTER_TABLE_PROGRESS
- SVL_S3LIST
- SVL_S3LOG
- SVL_S3PARTITION
- SVL_S3PARTITION_SUMMARY
- SVL_S3QUERY
- SVL_S3QUERY_SUMMARY
- SVL_S3RETRIES
- SVL_SPATIAL_SIMPLIFY
- SVL_SPECTRUM_SCAN_ERROR
- SVL_STATEMENTTEXT
- SVL_STORED_PROC_CALL
- SVL_STORED_PROC_MESSAGES
- SVL_TERMINATE
- SVL_UDF_LOG
- SVL_USER_INFO
- SVL_VACUUM_PERCENTAGE

SVL_AUTO_WORKER_ACTION

Regista ações automatizadas executadas pelo Amazon Redshift em tabelas definidas para otimização automática.

SVL_AUTO_WORKER_ACTION está visível apenas para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
table_id	integer	O identificador da tabela.
type	character (32)	O tipo de recomendação. Os valores possíveis são distkey e sortkey.
status	character (128)	O status da conclusão da recomendação. Os valores possíveis são Iniciar, Concluir, Ignorado, Abortar, Ponto de Verificação e Falha.
eventtime	timestamp	O timestamp da coluna `status`.
sequence	integer	O número de sequência de um valor truncado `previous_state` . Quando um único `previous_state` contém mais de 200 caracteres, linhas adicionais são registradas para esse valor. A sequência 0 é a primeira linha, 1 é a segunda e assim por diante.
previous_state	character (200)	O estilo de distribuição anterior e as chaves de classificação da tabela antes de aplicar a recomendação. O valor é truncado em incrementos de 200 caracteres.

Alguns exemplos de valores da coluna `status` são os seguintes:

- Ignorado: tabela não encontrada.
- Ignorado: a recomendação está vazia.
- Ignorado: aplicar recomendação de chave de classificação está desativada.
- Ignorado: repetir excede o limite máximo de uma tabela.
- Ignorado: coluna tabela foi alterada.
- Abortar: esta tabela não é AUTO.
- Abortar: esta tabela foi convertida recentemente.
- Abortar: esta tabela excede o limite de tamanho da tabela.

- Abortar: esta tabela já é o estilo recomendado.
- Ponto de verificação: progresso *21.9963*%.

Consultas de exemplo

No exemplo a seguir, as linhas no resultado mostram ações executadas pelo Amazon Redshift.

```
select table_id, type, status, eventtime, sequence, previous_state
from SVL_AUTO_WORKER_ACTION;
```

```
 table_id |  type   |                        status                         |
 eventtime          | sequence | previous_state
----------+---------+-------------------------------------------------------
+----------------------------+----------+---------------
   118082 | sortkey | Start                                                 | 2020-08-22
 19:42:20.727049 | 0        |
   118078 | sortkey | Start                                                 | 2020-08-22
 19:43:54.728819 | 0        |
   118082 | sortkey | Start                                                 | 2020-08-22
 19:42:52.690264 | 0        |
   118072 | sortkey | Start                                                 | 2020-08-22
 19:44:14.793572 | 0        |
   118082 | sortkey | Failed                                                | 2020-08-22
 19:42:20.728917 | 0        |
   118078 | sortkey | Complete                                              | 2020-08-22
 19:43:54.792705 | 0        | SORTKEY: None;
   118086 | sortkey | Complete                                              | 2020-08-22
 19:42:00.72635  | 0        | SORTKEY: None;
   118082 | sortkey | Complete                                              | 2020-08-22
 19:43:34.728144 | 0        | SORTKEY: None;
   118072 | sortkey | Skipped:Retry exceeds the maximum limit for a table. | 2020-08-22
 19:44:46.706155 | 0        |
   118086 | sortkey | Start                                                 | 2020-08-22
 19:42:00.685255 | 0        |
   118082 | sortkey | Start                                                 | 2020-08-22
 19:43:34.69531  | 0        |
   118072 | sortkey | Start                                                 | 2020-08-22
 19:44:46.703331 | 0        |
   118082 | sortkey | Checkpoint: progress 14.755079%                       | 2020-08-22
 19:42:52.692828 | 0        |
```

```
  118072 | sortkey | Failed                                                | 2020-08-22
19:44:14.796071 | 0       |
  116723 | sortkey | Abort:This table is not AUTO.                         | 2020-10-28
05:12:58.479233 | 0       |
  110203 | distkey | Abort:This table is not AUTO.                         | 2020-10-28
05:45:54.67259  | 0       |
```

SVL_COMPILE

Registra a hora e o local da compilação para cada segmento de consulta das consultas.

A exibição SVL_COMPILE é visível por todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> STL_WINDOW contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_QUERY_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Para obter informações sobre SVCS_COMPILE, consulte SVCS_COMPILE.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
xid	bigint	O ID da transação associada à instrução.
pid	integer	O ID do processo associado à instrução.

Nome da coluna	Tipo de dados	Descrição
consultar	integer	ID da consulta. Ele pode ser usado para unir várias outras tabelas e exibições do sistema.
segment	integer	O segmento de consulta a ser compilado.
locus	integer	O local onde o segmento é executado; **1** se for em um nó de computação e **2** se for no nó líder.
starttime	timestamp	O horário (em UTC) do início da compilação.
endtime	timestamp	O horário (em UTC) do término da compilação.
compile	integer	**0** se a compilação foi reutilizada, **1** se o segmento foi compilado.

Consultas de exemplo

Neste exemplo, as consultas 35878 e 35879 executaram a mesma instrução SQL. A coluna de compilação da consulta 35878 mostra o valor 1 para quatro segmentos da consulta, indicando que os segmentos foram compilados. A consulta 35879 mostra o valor 0 na coluna de compilação para todos os segmentos, indicando que os segmentos não precisaram ser compilados novamente.

```
select userid, xid,  pid, query, segment, locus,
datediff(ms, starttime, endtime) as duration, compile
from svl_compile
where query = 35878 or query = 35879
order by query, segment;

 userid |  xid   |  pid  | query | segment | locus | duration | compile
--------+--------+-------+-------+---------+-------+----------+---------
    100 | 112780 | 23028 | 35878 |       0 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       1 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       2 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       3 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       4 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       5 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       6 |     1 |     1380 |       1
    100 | 112780 | 23028 | 35878 |       7 |     1 |     1085 |       1
    100 | 112780 | 23028 | 35878 |       8 |     1 |     1197 |       1
```

```
    100 | 112780 | 23028 | 35878 |       9 |     2 |      905 |       1
    100 | 112782 | 23028 | 35879 |       0 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       1 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       2 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       3 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       4 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       5 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       6 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       7 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       8 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       9 |     2 |        0 |       0
(20 rows)
```

SVL_DATASHARE_CHANGE_LOG

Regista a exibição consolidada para rastrear alterações nos conjuntos de dados em clusters de produtores e consumidores.

SVL_DATASHARE_CHANGE_LOG permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_DATASHARE_CHANGE_LOG. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que executa a ação.
username	varchar(128)	O nome do usuário que está realizando a ação.
pid	integer	O ID do processo.
xid	bigint	O ID da transação.

Nome da coluna	Tipo de dados	Descrição
share_id	integer	O ID do datashare afetado.
share_name	varchar(128)	O nome do datashare.
source_database_id	integer	O ID do banco de dados ao qual o datashare pertence.
source_database_name	varchar(128)	O nome do banco de dados ao qual o datashare pertence.
consumer_database_id	integer	O ID do banco de dados importado do datashare.
consumer_database_name	varchar(128)	O nome do banco de dados importado do datashare.
arn	varchar(192)	O ARN do recurso por trás do banco de dados importado.
recordtime	timestamp	O timestamp da ação.
ação	varchar(128)	A ação que está sendo executada. Os valores possíveis são CREATE DATASHARE, DROP DATASHARE, GRANT ALTER, REVOKE ALTER, GRANT SHARE, REVOKE SHARE, ALTER ADD, ALTER REMOVE, ALTER SET, GRANT USAGE, REVOKE USAGE, CREATE DATABASE, GRANT ou REVOKE USAGE em um banco de dados compartilhado, DROP SHARED DATABASE e ALTER SHARED DATABASE.
status	integer	O status da ação. Os valores possíveis são SUCESS e ERROR-ERROR CODE.

Nome da coluna	Tipo de dados	Descrição
share_object_type	varchar(64)	O tipo de objeto de banco de dados que foi adicionado ou removido do datashare. Os valores possíveis são esquemas, tabelas, colunas, funções e exibições. Este é um campo para o cluster produtor.
share_object_id	integer	O ID do objeto de banco de dados que foi adicionado ou removido do datashare. Este é um campo para o cluster produtor.
share_object_name	varchar(128)	O nome do objeto de banco de dados que foi adicionado ou removido do datashare. Este é um campo para o cluster produtor.
target_user_type	varchar(16)	O tipo de usuários ou grupos aos quais um privilégio foi concedido. Este é um campo para o cluster do produtor e do consumidor.
target_userid	integer	O ID de usuários ou grupos aos quais um privilégio foi concedido. Este é um campo para o cluster do produtor e do consumidor.
target_username	varchar(128)	O nome dos usuários ou grupos aos quais um privilégio foi concedido. Este é um campo para o cluster do produtor e do consumidor.
consumer_account	varchar(16)	O ID da conta do consumidor de dados. Este é um campo para o cluster produtor.
consumer_namespace	varchar(64)	O namespace da conta de consumidor de dados. Este é um campo para o cluster produtor.
producer_account	varchar(16)	O ID da conta do produtor à qual o datashare pertence. Este é um campo para o cluster do consumidor.
producer_namespace	varchar(64)	O namespace da conta do produto ao qual o datashare pertence. Este é um campo para o cluster do consumidor.
attribute_name	varchar(64)	O nome de um atributo do datashare ou do banco de dados compartilhado.
attribute_value	varchar(128)	O valor de um atributo do datashare ou do banco de dados compartilhado.

Nome da coluna	Tipo de dados	Descrição
message	varchar(512)	A mensagem de erro quando uma ação falha.

Consultas de exemplo

O exemplo a seguir mostra uma visualização SVL_DATAASHARE_CHANGE_LOG.

```
SELECT DISTINCT action
FROM svl_datashare_change_log
WHERE share_object_name LIKE 'tickit%';

         action
 ----------------------
  "ALTER DATASHARE ADD"
```

SVL_DATASHARE_CROSS_REGION_USAGE

Use a visão SVL_DATASHARE_CROSS_REGION_USAGE para obter um resumo do uso transferido de dados entre regiões causado por uma consulta de compartilhamento de dados entre regiões. SVL_DATASHARE_CROSS_REGION_USAGE agrega detalhes no nível do segmento.

SVL_DATASHARE_CROSS_REGION_USAGE permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_DATASHARE_CROSS_REGION_USAGE. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
consultar	integer	A ID da consulta. Use esse valor para unir várias outras tabelas e visões do sistema.
segment	bigint	O número do segmento. Uma consulta consiste em vários segmentos e cada segmento consiste em uma ou mais etapas.
start_time	horário	O horário em UTC do início da transferência dos dados.
end_time	horário	O horário em UTC do término da transferência dos dados.
transferred_data	bigint	O número de bytes de dados transferidos de uma região produtora para uma região consumidora.
source_region	char(25)	A região produtora da qual a consulta transferiu os dados.
recordtime	timestamp	A hora em que a ação é registrada.

Consultas de exemplo

O exemplo a seguir mostra uma visão SVL_DATASHARE_CROSS_REGION_USAGE.

```
SELECT query, segment, transferred_data, source_region
from svl_datashare_cross_region_usage
where query = pg_last_query_id()
order by query,segment;

  query | segment | transferred_data | source_region
--------+---------+------------------+---------------
 200048 |       2 |          4194304 |    us-west-1
 200048 |       2 |          4194304 |    us-east-2
```

SVL_DATASHARE_USAGE_CONSUMER

Registra a atividade e o uso de datashares. Esta visualização só é relevante no cluster de consumidores.

SVL_DATACHASHARE_USAGE_CONSUMER é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_DATASHARE_USAGE_CONSUMER. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que emite a solicitação.
pid	integer	O ID do processo líder que executa a consulta.
xid	bigint	O contexto da transação atual.
request_id	varchar(50)	O ID exclusivo da chamada à API solicitada.
request_type	varchar(25)	O tipo de solicitação feita ao cluster produtor.
transaction_uid	varchar(50)	O ID exclusivo da transação.
recordtime	timestamp	A hora em que a ação é registrada.
status	integer	O status da chamada de API solicitada.
erro	varchar(512)	A mensagem para um erro.

Consultas de exemplo

O exemplo a seguir mostra uma visualização SVL_DATASHARE_USAGE_CONSUMER.

```
SELECT request_type, status, trim(error) AS error
FROM svl_datashare_usage_consumer

  request_type  | status | error
----------------+--------+--------
 "GET RELATION" |   0    |
```

SVL_DATASHARE_USAGE_PRODUCER

Registra a atividade e o uso de datashares. Essa visualização é relevante apenas no cluster do produtor.

SVL_DATASHARE_USAGE_PRODUER é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_DATASHARE_USAGE_PRODUCER. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
share_id	integer	O ID do objeto (OID) do datashare.
share_name	varchar(128)	O nome do datashare.
request_id	varchar(50)	O ID exclusivo da chamada à API solicitada.
request_type	varchar(25)	O tipo de solicitação feita ao cluster produtor.

Nome da coluna	Tipo de dados	Descrição
object_type	varchar(64)	O tipo do objeto que está sendo compartilhado do datashare. Os valores possíveis são esquemas, tabelas, colunas, funções e exibições.
object_oid	integer	O ID do objeto que está sendo compartilhado do datashare.
nome_objeto	varchar(128)	O nome do objeto que está sendo compartilhado do datashare.
consumer_account	varchar(16)	A conta da conta de consumidor com a qual o datashare é compartilhado.
consumer_namespace	varchar(64)	O namespace da conta de consumidor com a qual o datashare é compartilhado.
consumer_transaction_uid	varchar(50)	O ID de transação exclusivo da instrução no cluster do consumidor.
recordtime	timestamp	A hora em que a ação é registrada.
status	integer	O status do datashare.
erro	varchar(512)	A mensagem para um erro.
consumer_region	char(64)	A região na qual o cluster do consumidor está.

Consultas de exemplo

O exemplo a seguir mostra uma visualização SVL_DATASHARE_USAGE_PRODUCER.

```
SELECT DISTINCT request_type
FROM svl_datashare_usage_producer
WHERE object_name LIKE 'tickit%';
```

```
  request_type
------------------
  "GET RELATION"
```

SVL_FEDERATED_QUERY

Use a exibição SVL_FEDERATED_QUERY para visualizar informações sobre uma chamada de consulta federada.

SVL_FEDERATED_QUERY é visível por todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_EXTERNAL_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que executa a consulta.
xid	bigint	O ID da transação.
pid	integer	O ID do processo líder que executa a consulta.
consultar	integer	O ID de consulta de uma chamada federada.
sourcetype	character (32)	O tipo de fonte de chamada federada, por exemplo "PG".
recordtime	timestamp	A hora em que uma consulta é enviada para federação. UTC é usado.

Nome da coluna	Tipo de dados	Descrição
querytext	character (4000)	A string de consulta enviada ao mecanismo PostgreSQL remoto para execução.
num_rows	bigint	O número de linhas retornado pela consulta federada.
num_bytes	bigint	O número de bytes retornado pela consulta federada.
duration	bigint	O tempo gasto (em microsseg undos) para obter linhas das chamadas do cursor. É o tempo gasto para executar a consulta federada (bem como) para obter os resultados de volta.

Consultas de exemplo

Para mostrar informações sobre chamadas de consulta federada, execute a seguinte consulta.

```
select query, trim(sourcetype) as type, recordtime, trim(querytext) as "PG Subquery"
 from svl_federated_query where query = 4292;

 query | type |         recordtime         |                         pg subquery
-------+------+----------------------------
+------------------------------------------------------------
  4292 | PG   | 2020-03-27 04:29:58.485126 | SELECT "level" FROM functional.employees
 WHERE ("level" >= 6)
(1 row)
```

SVL_MULTI_STATEMENT_VIOLATIONS

Use a visualização SVL_MULTI_STATEMENT_VIOLATIONS para obter um registro completo de todos os comandos SQL executados no sistema que violam as restrições do bloco de transação.

Violações ocorrem quando você executa qualquer um dos seguintes comandos SQL que o Amazon Redshift restringe dentro de um bloco de transação ou solicitações de várias instruções:

- CREATE DATABASE
- DROP DATABASE
- ALTER TABLE APPEND
- CREATE EXTERNAL TABLE
- DROP EXTERNAL TABLE
- RENAME EXTERNAL TABLE
- ALTER EXTERNAL TABLE
- CREATE TABLESPACE
- DROP TABLESPACE
- CREATE LIBRARY
- DROP LIBRARY
- REBUILDCAT
- INDEXCAT
- REINDEX DATABASE
- VACUUM
- GRANT
- COPY

> **Note**
>
> Se houver alguma entrada nesta exibição, altere suas aplicações e scripts SQL correspondentes. Recomendamos alterar o código da aplicação para mover o uso desses comandos SQL restritos para fora do bloco de transação. Se precisar de assistência adicional, entre em contato com o AWS Support.

SVL_MULTI_STATEMENT_VIOLATIONS é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_QUERY_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que causou a violação.
banco de dados	character(32)	O nome do banco de dados ao qual o usuário estava conectado.
cmdname	character(20)	O nome do comando que não pode ser executado dentro de um bloco de transação ou solicitação de instrução múltipla. Por exemplo, CREATE DATABASE, DROP DATABASE, ALTER TABLE APPEND, CREATE EXTERNAL TABLE, DROP EXTERNAL TABLE, RENAME EXTERNAL TABLE, ALTER EXTERNAL TABLE, CREATE LIBRARY, DROP LIBRARY, REBUILDCAT, INDEXCAT, REINDEX DATABASE, VACUUM e GRANT em recursos externos, CLUSTER, COPY, CREATE TABLESPACE e DROP TABLESPACE.
xid	bigint	O ID da transação associada à instrução.
pid	integer	O ID do processo para a instrução.
rótulo	character(320)	O nome do arquivo usado para executar a consulta ou um rótulo definido com o comando SET QUERY_GROUP. Se a consulta não for baseada em arquivos ou o parâmetro QUERY_GROUP não estiver definido, o valor deste campo será branco.
starttime	timestamp	A hora exata em que a instrução começou a ser executada , com 6 dígitos de precisão para segundos fracionários, por exemplo: **`2009-06-12 11:29:19.131358`**

Nome da coluna	Tipo de dados	Descrição
endtime	timestamp	A hora exata em que a instrução terminou de ser executada, com 6 dígitos de precisão para segundos fracionários, por exemplo: **2009-06-12 11:29:19.193640**
sequence	integer	Quando uma única instrução contém mais de 200 caracteres, são registradas linhas adicionais para essa instrução. O valor 0 da sequência é a primeira linha, 1 é a segunda, e assim por diante.
type	varchar(10)	O tipo da instrução SQL: **QUERY**, **DDL** ou **UTILITY**.
texto	character(200)	O texto SQL, em incrementos de 200 caracteres. Esse campo pode conter caracteres especiais como barra invertida (\\) e nova linha (\n).

Consulta de exemplo

A consulta a seguir retorna várias instruções que têm violações.

```
select * from svl_multi_statement_violations order by starttime asc;

userid | database | cmdname | xid | pid | label | starttime | endtime | sequence | type
 | text
================================================================================================
1 | dev | CREATE DATABASE | 1034 | 5729 |label1 | ******** | ******* | 0 | DDL |
 create table c(b int);
1 | dev | CREATE DATABASE | 1034 | 5729 |label1 | ******** | ******* | 0 | UTILITY |
 create database b;
1 | dev | CREATE DATABASE | 1034 | 5729 |label1 | ******** | ******* | 0 | UTILITY |
 COMMIT
...
```

SVL_MV_REFRESH_STATUS

A visualização SVL_MV_MV_REFRESH_STATUS contém uma linha para a atividade de atualização de visualizações materializadas, incluindo visualizações materializadas do Iceberg criadas com USING ICEBERG.

Para obter mais informações sobre visões materializadas, consulte Visões materializadas no Amazon Redshift.

SVL_MV_REFRESH_STATUS é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Para visualizações materializadas do Iceberg, a coluna `schema_name` contém o nome do banco de dados AWS Glue e `mv_name` contém o nome da tabela do Iceberg. Os valores de status de atualização da visualização materializada do Iceberg incluem “MV já foi atualizada” (quando a visualização materializada não está obsoleta), “Atualizar MV recalculada com sucesso do zero” (atualização completa) e “Atualizar MV atualizada com êxito de forma incremental” (atualização incremental).

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_MV_REFRESH_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
db_name	char(128)	O banco de dados que contém a visão materiali zada.
userid	bigint	O ID do usuário que executou a atualização.
schema_name	char(128)	O esquema da visão materializada.
mv_name	char(128)	O nome da visão materializada.
xid	bigint	O ID da transação da atualização.

Nome da coluna	Tipo de dados	Descrição
starttime	timestamp	A hora de início da atualização.
endtime	timestamp	A hora de término da atualização.

Nome da coluna	Tipo de dados	Descrição
status	texto	O status da atualização. Exemplos de valores possíveis incluem o seguinte: • Atualizar MV atualizado com êxito increment almente Se for uma visão materializada para streaming, a mensagem pode ter qualificadores adicionais em relação ao número de registros. Incluindo o seguinte: • O fluxo não retornou dados novos: não houve nenhum registro recuperado. • Todos os registros recebidos do fluxo foram ignorados: os registros foram recuperados, mas devido a erros, todos foram ignorados. • Alguns registros do fluxo foram ignorados: os registros foram recuperados, mas devido a erros, todos foram ignorados. Se não houver qualificadores, pelo menos um registro foi recuperado e todos os registros estarão disponíveis na visão materializada. Ainda há um possível qualificador restante: • O fluxo pode conter mais dados: a atualizaç ão terminou antes que o Amazon Redshift determinasse que não havia mais registros para consumir. O fluxo pode estar atualizad o, mas não foi confirmado pelo Amazon Redshift. • Atualizar MV recalculado novamente com êxito do zero • Atualizar MV atualizado parcialmente increment almente até uma transação ativa

Nome da coluna	Tipo de dados	Descrição
		• O MV já foi atualizado • A atualização automática em cascata foi ignorada porque a visão materializada <schema>.<name> não foi atualizada. • A atualização em cascata foi ignorada porque a visão materializada <schema>.<name> não foi atualizada. • A atualização automática em cascata falhou porque a visão materializada <schema>.<name> não foi atualizada. A atualização em cascata falhou porque a visão materializada <schema>.<name> não foi atualizada. • Falha na atualização. Uma coluna da tabela base foi renomeada • Falha na atualização. Um tipo de coluna da tabela base foi alterado • Falha na atualização. Uma tabela base foi renomeada • Atualização falhou devido a um erro interno • Falha na atualização. Uma coluna da tabela base foi descartada • Falha na atualização. Esquema do MV foi renomeado • Falha na atualização. O MV não foi encontrado • Atualização automática cancelada devido à workload excessiva do usuário • Falha na atualização. Violação de isolamento serializável
refresh_type	char(32)	A definição do tipo de atualização. Os valores de exemplo incluem Manual e Auto.

Nome da coluna	Tipo de dados	Descrição
consumer_account	char(12)	O ID da conta da AWS do cluster de consumidor que iniciou a atualização. Esse campo é preenchido quando a atualização é iniciada em um cluster de consumidor em uma configuração de compartilhamento de dados.
consumer_region	char(32)	A região da AWS do cluster de consumidor que iniciou a atualização. Esse campo é preenchido quando a atualização é iniciada em um cluster de consumidor em uma configuração de compartilhamento de dados.
consumer_namespace	char(36)	O identificador de namespace do cluster de consumidor que iniciou a atualização. Esse campo é preenchido quando a atualização é iniciada em um cluster de consumidor em uma configuração de compartilhamento de dados.

Consulta de exemplo

Para exibir o status de atualização das visões materializadas, execute a consulta a seguir.

```
select * from svl_mv_refresh_status;
```

Essa consulta retorna os seguintes dados de saída de exemplo:

```
 db_name | userid |  schema   |  name   |  xid  |         starttime          |
      endtime           |                        status
                          |  refresh_type | consumer_account | consumer_region |
 consumer_namespace
---------+--------+-----------+---------+-------+----------------------------
+----------------------------
+-------------------------------------------------------------------------------
+---------------+------------------+-----------------
+-----------------------------------
```

```
dev     |    169 | mv_schema | mv_test |  6640 | 2020-02-14 02:26:53.497935 |
2020-02-14 02:26:53.556156 | Refresh successfully recomputed MV from scratch
                          |  Manual       |                 |                |
dev     |    166 | mv_schema | mv_test |  6517 | 2020-02-14 02:26:39.287438 |
2020-02-14 02:26:39.349539 | Refresh successfully updated MV incrementally
                          |  Auto         |                 |                |
ext_db  |    162 | producer_schema | producer_mv |  6388 | 2020-02-14 02:26:27.863426
| 2020-02-14 02:26:27.918307 | Refresh successfully updated MV incrementally
                            |  Manual       | 0123456789       | us-east-1       |
623d8ff2-4391-4381-83d7-177caa6767af
dev     |    161 | mv_schema | mv_test |  6323 | 2020-02-14 02:26:20.020717 |
2020-02-14 02:26:20.080002 | Refresh successfully updated MV incrementally
                          |  Auto         |                 |                |
dev     |    161 | mv_schema | mv_test |  6301 | 2020-02-14 02:26:05.796146 |
2020-02-14 02:26:07.853986 | Refresh successfully recomputed MV from scratch
                          |  Manual       |                 |                |
dev     |    153 | mv_schema | mv_test |  6024 | 2020-02-14 02:25:18.762335 |
2020-02-14 02:25:20.043462 | MV was already updated
                          |  Manual       |                 |                |
dev     |    143 | mv_schema | mv_test |  5557 | 2020-02-14 02:24:23.100601 |
2020-02-14 02:24:23.100633 | MV was already updated
                          |  Manual
dev     |    141 | mv_schema | mv_test |  5447 | 2020-02-14 02:23:54.102837 |
2020-02-14 02:24:00.310166 | Refresh successfully updated MV incrementally
                          |  Auto
dev     |      1 | mv_schema | mv_test |  5329 | 2020-02-14 02:22:26.328481 |
2020-02-14 02:22:28.369217 | Refresh successfully recomputed MV from scratch
                          |  Auto
dev     |    138 | mv_schema | mv_test |  5290 | 2020-02-14 02:21:56.885093 |
2020-02-14 02:21:56.885098 | Refresh failed. MV was not found
                          |  Manual
dev     |    138 | mv_schema | mv_test |  5100 | 2020-02-14 02:20:33.123445 |
2020-02-14 02:20:33.123445 | Cascade refresh failed because materialized view
mv_schema.child was not refreshed. |  Manual
dev     |    138 | mv_schema | child   |  5100 | 2020-02-14 02:20:33.123445 |
2020-02-14 02:20:33.123445 | Refresh failed due to an internal error.
                          |  Manual
dev     |    138 | mv_schema | mv_test |  5099 | 2020-02-14 02:10:23.492344 |
2020-02-14 02:10:23.492344 | Cascade refresh skipped because materialized view
mv_schema.child was not refreshed.|  Manual
dev     |    138 | mv_schema | child   |  5099 | 2020-02-14 02:10:23.492344 |
2020-02-14 02:10:23.492344 | Refresh failed due to an internal error.
                          |  Manual
```

SVL_QERROR

A exibição SVL_QERROR está obsoleta.

SVL_QLOG

A exibição SVL_QLOG contém um log de todas as consultas executadas para o banco de dados.

O Amazon Redshift cria a exibição SVL_QLOG como um subconjunto legível de informações da tabela STL_QUERY. Use esta tabela para encontrar o ID de uma consulta executada recentemente ou para ver em quanto tempo uma consulta foi concluída.

SVL_QLOG é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_QUERY_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. Este ID pode ser usado para unir várias outras tabelas e exibições do sistema.
xid	bigint	ID da transação.
pid	integer	O ID do processo associado à consulta.
starttime	timestamp	A hora exata em que a instrução começou a ser executada , com seis dígitos de precisão para segundos fracionários, por exemplo: **`2009-06-12 11:29:19.131358`**
endtime	timestamp	A hora exata em que a instrução terminou de ser executada, com seis dígitos de precisão para segundos

Nome da coluna	Tipo de dados	Descrição
		fracionários, por exemplo: **2009-06-12 11:29:19.193640**
elapsed	bigint	O tempo que levou para a consulta ser executada (em microssegundos).
aborted	integer	Se uma consulta for interrompida pelo sistema ou cancelada pelo usuário, essa coluna terá o valor **1**. Se a consulta foi executada até o final, essa coluna conterá **0**. As consultas que são canceladas para fins de gerenciam ento de workload e reiniciadas posteriormente também têm um valor **1** nesta coluna.
rótulo	character(320)	O nome do arquivo usado para executar a consulta ou um rótulo definido com o comando SET QUERY_GROUP. Se a consulta não for baseada em arquivos ou o parâmetro QUERY_GROUP não estiver definido, o valor deste campo será `default`.
substring	character(60)	O texto truncado da consulta.
source_qu ery	integer	Se a consulta tiver usado o armazenamento em cache do resultado, este é o ID da consulta que originou os resultados armazenados em cache. Se o armazenam ento em cache de resultados não foi usado, o valor desse campo é `NULL`.
concurren cy_scalin g_status_ txt	texto	Uma descrição de se a consulta foi executada no cluster principal ou no cluster de escalonamento de simultane idade.

Nome da coluna	Tipo de dados	Descrição
from_sp_call	integer	Se a consulta foi chamada a partir de um procedimento armazenado, o ID da consulta da chamada de procedimento. Se a consulta não tiver sido executada como parte de um procedimento armazenado, esse campo será NULL.

Consultas de exemplo

O exemplo a seguir retorna o ID da consulta, o tempo de execução e o texto truncado da consulta para as cinco consultas de banco de dados mais recentes executadas pelo usuário com `userid = 100`.

```
select query, pid, elapsed, substring from svl_qlog
where userid = 100
order by starttime desc
limit 5;

 query  |  pid  | elapsed  |           substring
--------+-------+----------+-----------------------------------------------
 187752 | 18921 | 18465685 | select query, elapsed, substring from svl_...
 204168 |  5117 |    59603 | insert into testtable values (100);
 187561 | 17046 |  1003052 | select * from pg_table_def where tablename...
 187549 | 17046 |  1108584 | select * from STV_WLM_SERVICE_CLASS_CONFIG
 187468 | 17046 |  5670661 | select * from pg_table_def where schemaname...
(5 rows)
```

O exemplo a seguir retorna o nome do script SQL (coluna LABEL) e o tempo decorrido para uma consulta que foi cancelada (**`aborted=1`**):

```
select query, elapsed, trim(label) querylabel
from svl_qlog where aborted=1;

 query | elapsed  |       querylabel
-------+----------+-------------------------
    16 |  6935292 | alltickittablesjoin.sql
(1 row)
```

SVL_QUERY_METRICS

A exibição SVL_QUERY_METRICS mostra as métricas das consultas concluídas. Essa exibição é derivada da tabela do sistema STL_QUERY_METRICS. Use os valores dessa exibição como um auxílio para determinar os valores limite de definição das regras de monitoramento de consultas. Para obter mais informações, consulte Regras de monitoramento de consulta do WLM.

SVL_QUERY_METRICS permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que executou a consulta que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
service_class	integer	O ID da fila de consultas do WLM (classe de serviço). As filas de consultas são definidas na configuração do WLM. As métricas são relatadas somente para as filas definidas pelo usuário. Para obter uma lista dos IDs de classe de serviço, consulte IDs da classe de serviço do WLM.
dimension	varchar(24)	A dimensão em que a métrica é relatada. Os valores possíveis são consulta, segmento e etapa.
segment	integer	O número do segmento. Uma consulta consiste em vários segmentos e cada segmento consiste em uma ou mais etapas. Os segmentos de uma consulta podem ser executado

Nome da coluna	Tipo de dados	Descrição
		s em paralelo. Cada segmento executa em um único processo. Se o valor do segmento é 0, os valores das métricas do segmento são acumulados no nível de consulta.
etapa	integer	O ID do tipo da etapa realizada. A descrição do tipo da etapa é mostrada na coluna `step_label` .
step_label	varchar(30)	Tipo de etapa realizada.
query_cpu_time	bigint	O tempo de CPU usado pela consulta (em segundos). O tempo de CPU é diferente do tempo de execução da consulta.
query_blocks_read	bigint	O número de blocos de 1 MB lidos pela consulta.
query_execution_time	bigint	O tempo de execução decorrido para uma consulta (em segundos). O tempo de execução não inclui o tempo de espera gasto em uma fila. Consulte `query_queue_time` para obter o tempo na fila.
query_cpu_usage_percent	bigint	A porcentagem da capacidade de CPU usada pela consulta.
query_temp_blocks_to_disk	bigint	A quantidade de espaço em disco usada por uma consulta para gravar resultados intermediários, em MB.
segment_execution_time	bigint	O tempo de execução decorrido para um único segmento (em segundos).
cpu_skew	numeric(38,2)	A taxa de utilização máxima da CPU para uma fatia em relação à média de utilização da CPU para todas as fatias. Essa métrica é definida no nível do segmento.
io_skew	numeric(38,2)	A taxa máxima de leitura de blocos (E/S) para uma fatia em relação à média de leitura de blocos para todas as fatias.

Nome da coluna	Tipo de dados	Descrição
scan_row_count	bigint	O número de linhas em uma etapa de varredura. A contagem de linhas é o número total de linhas emitidas antes da filtragem das linhas marcadas para exclusão (linhas fantasmas) e antes da aplicação dos filtros de consulta definidos pelo usuário.
join_row_count	bigint	O número de linhas processadas em uma etapa de junção.
nested_loop_join_row_count	bigint	O número de linhas em uma etapa de junção de loops aninhados.
return_row_count	bigint	O número de linhas retornado pela consulta.
spectrum_scan_row_count	bigint	O número de linhas processadas pela varredura do Amazon Redshift Spectrum no Amazon S3.
spectrum_scan_size_mb	bigint	A quantidade de dados, em MB, verificados pelo Amazon Redshift Spectrum no Amazon S3.
query_queue_time	bigint	O tempo em segundos que a consulta estava em fila.

SVL_QUERY_METRICS_SUMMARY

A exibição SVL_QUERY_METRICS_SUMMARY mostra os valores máximos das métricas das consultas concluídas. Essa exibição é derivada da tabela do sistema STL_QUERY_METRICS. Use os valores dessa exibição como um auxílio para determinar os valores limite de definição das regras de monitoramento de consultas. Para obter mais informações sobre regras e métricas do monitoramento de consultas do Amazon Redshift, consulte Regras de monitoramento de consulta do WLM.

SVL_QUERY_METRICS_SUMMARY permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que executou a consulta que gerou a entrada.
consultar	integer	ID da consulta. A coluna de consulta pode ser usada para unir outras tabelas e exibições do sistema.
service_class	integer	O ID da fila de consultas do WLM (classe de serviço). As filas de consultas são definidas na configuração do WLM. As métricas são relatadas somente para as filas definidas pelo usuário. Para obter uma lista dos IDs de classe de serviço, consulte IDs da classe de serviço do WLM.
query_cpu_time	bigint	O tempo de CPU usado pela consulta (em segundos). O tempo de CPU é diferente do tempo de execução da consulta.
query_blocks_read	bigint	O número de blocos de 1 MB lidos pela consulta.
query_execution_time	bigint	O tempo de execução decorrido para uma consulta (em segundos). O tempo de execução não inclui o tempo de espera gasto em uma fila.
query_cpu_usage_percent	numeric(38,2)	A porcentagem da capacidade de CPU usada pela consulta.

Nome da coluna	Tipo de dados	Descrição
query_temp_blocks_to_disk	bigint	A quantidade de espaço em disco usada por uma consulta para gravar resultados intermediários, em MB.
segment_execution_time	bigint	O tempo de execução decorrido para um único segmento (em segundos).
cpu_skew	numeric(38,2)	A taxa de utilização máxima da CPU para uma fatia em relação à média de utilização da CPU para todas as fatias. Essa métrica é definida no nível do segmento.
io_skew	numeric(38,2)	A taxa máxima de leitura de blocos (E/S) para uma fatia em relação à média de leitura de blocos para todas as fatias.
scan_row_count	bigint	O número de linhas em uma etapa de varredura. A contagem de linhas é o número total de linhas emitidas antes da filtragem das linhas marcadas para exclusão (linhas fantasmas) e antes da aplicação dos filtros de consulta definidos pelo usuário.
join_row_count	bigint	O número de linhas processadas em uma etapa de junção.
nested_loop_join_row_count	bigint	O número de linhas em uma etapa de junção de loops aninhados.
return_row_count	bigint	O número de linhas retornado pela consulta.
spectrum_scan_row_count	bigint	O número de linhas processadas pela varredura do Amazon Redshift Spectrum no Amazon S3.
spectrum_scan_size_mb	bigint	A quantidade de dados, em MB, verificados pelo Amazon Redshift Spectrum no Amazon S3.
query_queue_time	bigint	O tempo em segundos que a consulta estava em fila.

SVL_QUERY_QUEUE_INFO

Apresenta um resumo com os detalhes das consultas que permanecem em uma fila de consultas de gerenciamento de workload (WLM) ou em uma fila de confirmação.

A exibição SVL_QUERY_QUEUE_INFO filtra as consultas realizadas pelo sistema e mostra somente as consultas realizadas por um usuário.

A exibição SVL_QUERY_QUEUE_INFO resume as informações das tabelas do sistema STL_QUERY, STL_WLM_QUERY e STL_COMMIT_STATS.

SVL_QUERY_QUEUE_INFO só permanece visível para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
banco de dados	texto	O nome do banco de dados ao qual o usuário estava conectado quando a consulta foi enviada.
consultar	integer	ID da consulta.
xid	bigint	ID da transação.
userid	integer	O ID do usuário que gerou a consulta.
querytxt	texto	Os primeiros 100 caracteres do texto da consulta.
queue_start_time	timestamp	O horário (em UTC) em que a consulta entrou na fila WLM.
exec_start_time	timestamp	O horário (em UTC) do início da execução da consulta.
service_class	integer	ID da classe de serviço. As classes de serviço são definidas no arquivo de configuração do WLM.
slots	integer	O número de slots da consulta de WLM.

Nome da coluna	Tipo de dados	Descrição
queue_elapsed	bigint	O tempo de espera da consulta em uma fila do WLM (em segundos).
exec_elapsed	bigint	O tempo de execução da consulta (em segundos).
wlm_total_elapsed	bigint	O tempo de espera da consulta em uma fila do WLM (queue_elapsed), mais o tempo de execução da consulta (exec_elapsed).
commit_queue_elapsed	bigint	O tempo de espera da consulta em uma fila de confirmação (em segundos).
commit_exec_time	bigint	O tempo gasto pela consulta na operação de confirmação (em segundos).
service_class_name	character(64)	O nome da classe de serviços.

Consultas de exemplo

O exemplo a seguir mostra o tempo que as consultas passaram nas filas do WLM.

```
select query, service_class, queue_elapsed, exec_elapsed, wlm_total_elapsed
from svl_query_queue_info
where wlm_total_elapsed > 0;

  query  | service_class | queue_elapsed | exec_elapsed | wlm_total_elapsed
---------+---------------+---------------+--------------+-------------------
 2742669 |             6 |             2 |          916 |               918
 2742668 |             6 |             4 |          197 |               201
(2 rows)
```

SVL_QUERY_REPORT

O Amazon Redshift cria a visualização SVL_QUERY_REPORT de uma UNION de várias tabelas de sistema STL do Amazon Redshift para fornecer informações sobre as etapas de consulta concluídas.

Essa visualização divide as informações sobre as consultas executadas por fatia e por etapa, o que pode ajudar a solucionar problemas de nó e fatia no cluster do Amazon Redshift.

SVL_QUERY_REPORT é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. Ele pode ser usado para unir várias outras tabelas e exibições do sistema.
slice	integer	A fatia dos dados onde a etapa foi executada.
segment	integer	O número do segmento. Uma consulta consiste em vários segmentos e cada segmento consiste em uma ou mais etapas. Os segmentos de uma consulta podem ser executados em paralelo. Cada segmento executa em um único processo.
etapa	integer	A etapa da consulta que foi concluída.

Nome da coluna	Tipo de dados	Descrição
start_time	timestamp	O horário exato (em UTC) do início da execução do segmento, com 6 dígitos de precisão para as frações de segundo. Por exemplo, .: **2012-12-12 11:29:19.131358**
end_time	timestamp	O horário exato (em UTC) do término da execução do segmento, com 6 dígitos de precisão para as frações de segundo. Por exemplo, .: **2012-12-12 11:29:19.131467**
elapsed_time	bigint	O tempo (em microssegundos) que levou para o segmento ser executado.
rows	bigint	O número de linhas produzidas por etapa (por fatia). Esse número representa o número de linhas da fatia que resulta da execução das etapas, e não do número de linhas recebidas ou processadas por etapa. Em outras palavras, esse é o número de linhas que sobreviveram essa etapa e foram passadas para a próxima etapa.
bytes	bigint	O número de bytes produzidos pela etapa (por fatia).
rótulo	char(256)	O rótulo da etapa, que consiste no nome da etapa da consulta e, se for aplicável, no ID e no nome da tabela (por exemplo, `scan tbl=100448 name =user`). Os IDs de tabela com três dígitos geralmente indicam varreduras de tabelas transitórias. Quando você vê `tbl=0`, isso normalmente indica uma varredura de um valor constante.
is_diskbased	character (1)	Indica se esta etapa da consulta foi realizada como uma operação em disco: true (**t**) ou false (**f**). Somente algumas etapas, como hash, classificação e etapas de agregação podem ir para o disco. Muitos tipos de etapas são sempre realizados na memória.
workmem	bigint	A quantidade de memória de trabalho (em bytes) atribuída à etapa da consulta. Este valor é o limite de query_working_mem alocado para ser usado durante a execução, não a quantidade de memória que foi realmente usada

Nome da coluna	Tipo de dados	Descrição
is_rrscan	character (1)	O valor true (**t**) indica que a varredura restrita ao intervalo foi utilizada na etapa.
is_delayed_scan	character (1)	O valor true (**t**) indica que a varredura com atraso foi utilizada na etapa.
rows_pre_filter	bigint	Para varreduras de tabelas permanentes, o número total de linhas enviadas antes da filtragem das linhas marcadas para exclusão (linhas fantasma) e antes de aplicar os filtros de consulta definidos pelo usuário.

Consultas de exemplo

A consulta a seguir demonstra a distorção dos dados das linhas retornadas para a consulta com o ID 279. Use esta consulta para determinar se os dados do banco de dados serão distribuídos uniformemente pelas fatias no cluster do data warehouse:

```
select query, segment, step, max(rows), min(rows),
case when sum(rows) > 0
then ((cast(max(rows) -min(rows) as float)*count(rows))/sum(rows))
else 0 end
from svl_query_report
where query = 279
group by query, segment, step
order by segment, step;
```

Esta consulta deve retornar dados semelhantes aos apresentados na seguinte saída de exemplo:

```
query | segment | step |   max    |   min    |         case
------+---------+------+----------+----------+----------------------
279 |       0 |    0 | 19721687 | 19721687 |                    0
279 |       0 |    1 | 19721687 | 19721687 |                    0
279 |       1 |    0 |   986085 |   986084 | 1.01411202804304e-06
279 |       1 |    1 |   986085 |   986084 | 1.01411202804304e-06
279 |       1 |    4 |   986085 |   986084 | 1.01411202804304e-06
279 |       2 |    0 |  1775517 |   788460 |     1.00098637606408
279 |       2 |    2 |  1775517 |   788460 |     1.00098637606408
```

```
279 |       3 |    0 |  1775517 |   788460 |     1.00098637606408
279 |       3 |    2 |  1775517 |   788460 |     1.00098637606408
279 |       3 |    3 |  1775517 |   788460 |     1.00098637606408
279 |       4 |    0 |  1775517 |   788460 |     1.00098637606408
279 |       4 |    1 |  1775517 |   788460 |     1.00098637606408
279 |       4 |    2 |        1 |        1 |                    0
279 |       5 |    0 |        1 |        1 |                    0
279 |       5 |    1 |        1 |        1 |                    0
279 |       6 |    0 |       20 |       20 |                    0
279 |       6 |    1 |        1 |        1 |                    0
279 |       7 |    0 |        1 |        1 |                    0
279 |       7 |    1 |        0 |        0 |                    0
(19 rows)
```

SVL_QUERY_SUMMARY

Use a exibição SVL_QUERY_SUMMARY para encontrar informações gerais sobre a execução de uma consulta.

A exibição SVL_QUERY_SUMMARY contém um subconjunto de dados da exibição SVL_QUERY_REPORT. Observe que as informações na SVL_QUERY_SUMMARY são agregadas de todos os nós.

> **Note**
>
> A visualização SVL_QUERY_SUMMARY contém apenas informações sobre as consultas realizadas pelo Amazon Redshift, e não por outros comandos de utilitário e DDL. Para obter uma lista completa e informações sobre todas as instruções executadas pelo Amazon Redshift, inclusive DDL e comandos de utilitário, é possível consultar a visualização SVL_STATEMENTTEXT.

SVL_QUERY_SUMMARY é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Para obter informações sobre SVCS_QUERY_SUMMARY, consulte SVCS_QUERY_SUMMARY.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
consultar	integer	ID da consulta. Ele pode ser usado para unir várias outras tabelas e exibições do sistema.
stm	integer	Stream: um conjunto de segmentos simultâneos em uma consulta. Uma consulta tem um ou mais streams.
seg	integer	O número do segmento. Uma consulta consiste em vários segmentos e cada segmento consiste em uma ou mais etapas. Os segmentos de uma consulta podem ser executados em paralelo. Cada segmento executa em um único processo.
etapa	integer	Etapa da consulta que foi executada.
maxtime	bigint	O tempo máximo de execução para esta etapa (em microsseg undos).
avgtime	bigint	O tempo médio de execução para esta etapa (em microsseg undos).
rows	bigint	O número de linhas de dados envolvidas na etapa da consulta.
bytes	bigint	O número de bytes de dados envolvidos na etapa da consulta.
rate_row	double precision	A taxa de execução da consulta por linha.
rate_byte	double precision	A taxa de execução da consulta por byte.
rótulo	texto	O rótulo da etapa, que consiste no nome da etapa da consulta e, se for aplicável, no ID e no nome da tabela (por exemplo,

Nome da coluna	Tipo de dados	Descrição
		scan tbl=100448 name =user). Os IDs de tabela com três dígitos geralmente indicam varreduras de tabelas transitórias. Quando você vê `tbl=0`, isso normalmente indica uma varredura de um valor constante.
is_diskbased	character(1)	Indica se esta etapa da consulta foi realizada como uma operação em disco em qualquer nó do cluster: true (**t**) ou false (**f**). Somente algumas etapas, como hash, classificação e etapas de agregação podem ir para o disco. Muitos tipos de etapas são sempre realizados na memória.
workmem	bigint	A quantidade de memória de trabalho (em bytes) atribuída à etapa da consulta.
is_rrscan	character(1)	O valor true (**t**) indica que a varredura restrita ao intervalo foi utilizada na etapa. O padrão é falso (**f**).
is_delayed_scan	character(1)	O valor true (**t**) indica que a varredura com atraso foi utilizada na etapa. O padrão é falso (**f**).
rows_pre_filter	bigint	Para varreduras de tabelas permanentes, o número total de linhas enviadas antes da filtragem das linhas marcadas para exclusão (linhas fantasma).

Consultas de exemplo

Visualização de informações de processamento para uma etapa da consulta

A consulta a seguir mostra as informações básicas de processamento para cada etapa da consulta 87:

```
select query, stm, seg, step, rows, bytes
from svl_query_summary
where query = 87
order by query, seg, step;
```

Esta consulta recupera as informações do processamento da consulta 87, como mostram os seguintes dados de saída de exemplo:

```
 query | stm | seg | step |  rows  |  bytes
-------+-----+-----+------+--------+---------
87     |   0 |   0 |    0 |     90 |    1890
87     |   0 |   0 |    2 |     90 |     360
87     |   0 |   1 |    0 |     90 |     360
87     |   0 |   1 |    2 |     90 |    1440
87     |   1 |   2 |    0 | 210494 | 4209880
87     |   1 |   2 |    3 |  89500 |       0
87     |   1 |   2 |    6 |      4 |      96
87     |   2 |   3 |    0 |      4 |      96
87     |   2 |   3 |    1 |      4 |      96
87     |   2 |   4 |    0 |      4 |      96
87     |   2 |   4 |    1 |      1 |      24
87     |   3 |   5 |    0 |      1 |      24
87     |   3 |   5 |    4 |      0 |       0
(13 rows)
```

Determinar se houve transbordamento das etapas de consulta para o disco

A consulta a seguir mostra se houve transbordamento para o disco em alguma das etapas da consulta de ID 1025 (veja a exibição SVL_QLOG para saber como obter o ID de uma consulta) ou se a consulta foi totalmente executada na memória:

```
select query, step, rows, workmem, label, is_diskbased
from svl_query_summary
where query = 1025
order by workmem desc;
```

Essa consulta retorna os seguintes dados de saída de exemplo:

```
query| step|  rows  |  workmem   |  label        | is_diskbased
-----+-----+--------+------------+---------------+--------------
1025 |  0  |16000000|  141557760 |scan tbl=9     | f
1025 |  2  |16000000|  135266304 |hash tbl=142   | t
1025 |  0  |16000000|  128974848 |scan tbl=116536| f
1025 |  2  |16000000|  122683392 |dist           | f
(4 rows)
```

Ao fazer a varredura dos valores pelo IS_DISKBASED, você pode ver quais etapas de consulta foram executadas no disco. Para a consulta 1025, a etapa de hash foi executada em disco. As etapas que podem ser executadas em disco incluem a hash, a aggr e as etapas de classificação. Para visualizar apenas a etapas da consulta que são baseadas em disco, adicione a cláusula **`and is_diskbased = 't'`** na instrução de SQL no exemplo acima.

SVL_RESTORE_ALTER_TABLE_PROGRESS

Use SVL_RESTORE_ALTER_TABLE_PROGRESS para monitorar o andamento da migração de cada tabela no cluster durante um redimensionamento clássico para nós RG ou RA3. Ele captura o throughput histórico da migração de dados durante a operação de redimensionamento. Para obter mais informações sobre o redimensionamento clássico para nós RG ou RA3, consulte Redimensionamento clássico.

SVL_RESTORE_ALTER_TABLE_PROGRESS é visível somente aos superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_RESTORE_LOG. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

> **Note**
>
> As linhas com andamento de `100.00%` ou `ABORTED` são excluídas após sete dias. As linhas de tabelas descartadas durante ou após um redimensionamento clássico ainda podem aparecer em SVL_RESTORE_ALTER_TABLE_PROGRESS.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
tbl	integer	O ID da tabela.
progresso	char(32)	O status do progresso da redistribuição da tabela. Os valores possíveis são porcentagens de `0.00%` até `100.00%` e a mensagem`ABORTED`. `ABORTED` significa que a redistribuição foi

Nome da coluna	Tipo de dados	Descrição
		interrompida sem ser concluída, e o motivo é explicado na coluna `message`.
message	char(256)	A mensagem associada ao andamento da redistribuição da tabela.

Consulta de exemplo

A consulta a seguir retorna consultas em execução e enfileiradas.

```
select * from svl_restore_alter_table_progress;

tbl     | progress |                           message
--------+----------+-------------------------------------------------------------
105614  | ABORTED  | Abort:Table no longer contains the prior dist key column.
105610  | ABORTED  | Abort:Table no longer contains the prior dist key column.
105594  | 0.00%    | Table waiting for alter diststyle conversion.
105602  | ABORTED  | Abort:Table no longer contains the prior dist key column.
105606  | ABORTED  | Abort:Table no longer contains the prior dist key column.
105598  | 100.00%  | Restored to distkey successfully.
```

SVL_S3LIST

Use a visualização SVL_S3LIST para obter detalhes sobre as consultas de data lake no nível do segmento.

SVL_S3LIST é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> SVL_S3LIST contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS

SYS_EXTERNAL_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
consultar	integer	O ID da consulta.
segment	integer	O número do segmento. Uma consulta consiste em vários segmentos.
nó	integer	O número de nós.
slice	integer	A fatia dos dados com os quais determinado segmento foi executado.
eventtime	timestamp	O horário em UTC em que o evento é gravado.
bucket	texto	O nome do bucket do Amazon S3.
prefix	texto	O prefixo do local do bucket do Amazon S3.
recursive	char(1)	Se há varredura recursiva para subpastas.
retrieved_files	integer	O número de arquivos listados.
max_file_size	bigint	O tamanho máximo de arquivo entre os arquivos listados.
avg_file_size	double precision	O tamanho médio de arquivo entre os arquivos listados.
generated_splits	integer	O número de divisões de arquivos.

Nome da coluna	Tipo de dados	Descrição
avg_split_length	double precision	O tamanho médio das divisões de arquivos em bytes.
duration	bigint	A duração da listagem, em microssegundos.

Consulta de exemplo

O exemplo a seguir consulta SVL_S3LIST para verificar a última consulta que foi executada.

```
select *
from svl_s3list
where query = pg_last_query_id()
order by query,segment;
```

SVL_S3LOG

Use a visualização SVL_S3LOG para obter detalhes sobre as consultas de data lake nos níveis de segmento e de fatia do nó.

SVL_S3LOG é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> SVL_S3LOG contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_EXTERNAL_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
pid	integer	O ID do processo.
consultar	integer	O ID da consulta.
segment	integer	O número do segmento. Uma consulta consiste em vários segmentos e cada segmento consiste em uma ou mais etapas.
etapa	integer	A etapa da consulta que foi executada.
nó	integer	O número de nós.
slice	integer	A fatia dos dados com os quais determinado segmento foi executado.
eventtime	timestamp	O horário (em UTC) de início da execução da etapa.
message	texto	A mensagem de entrada do log.

Consulta de exemplo

O exemplo a seguir consulta SVL_S3LOG para verificar a última consulta executada.

```
select *
from svl_s3log
where query = pg_last_query_id()
order by query,segment,slice;
```

SVL_S3PARTITION

Use a visualização SVL_S3PARTITION para obter detalhes sobre as partições do data lake nos níveis de segmento e de fatia do nó.

SVL_S3PARTITION é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> SVL_S3PARTITION contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_EXTERNAL_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
consultar	integer	O ID da consulta.
segment	integer	O número do segmento. Uma consulta consiste em vários segmentos e cada segmento consiste em uma ou mais etapas.
nó	integer	O número de nós.
slice	integer	A fatia dos dados com os quais determinado segmento foi executado.
starttime	time stamp sem fuso horário	O horário em UTC que o corte de partição começou a ser executado.
endtime	time stamp sem fuso horário	O horário em UTC que o corte de partição foi concluído.
duration	bigint	Tempo decorrido (em microssegundos).

Nome da coluna	Tipo de dados	Descrição
total_partitions	integer	Número de partições totais.
qualified_partitions	integer	O número de partições qualificadas.
assigned_partitions	integer	O número de partições atribuídas na fatia.
assignment	caractere	Tipo de atribuição.

Consulta de exemplo

O exemplo a seguir obtém os detalhes de partição da última consulta concluída.

```
SELECT query, segment,
       MIN(starttime) AS starttime,
       MAX(endtime) AS endtime,
       datediff(ms,MIN(starttime),MAX(endtime)) AS dur_ms,
       MAX(total_partitions) AS total_partitions,
       MAX(qualified_partitions) AS qualified_partitions,
       MAX(assignment) as assignment_type
FROM svl_s3partition
WHERE query=pg_last_query_id()
GROUP BY query, segment
```

```
query | segment |           starttime           |           endtime           | dur_ms|
 total_partitions | qualified_partitions | assignment_type
------+---------+-------------------------------+-----------------------------+-------
+------------------+----------------------+----------------
99232 |       0 | 2018-04-17 22:43:50.201515    | 2018-04-17 22:43:54.674595  |  4473 |
       2526       |        334           | p
```

SVL_S3PARTITION_SUMMARY

Use a visualização SVL_S3PARTITION_SUMMARY para obter um resumo do processamento da partição de consultas do Redshift Spectrum no nível do segmento.

SVL_S3PARTITION_SUMMARY é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Para obter informações sobre SVCS_S3PARTITION, consulte SVCS_S3PARTITION_SUMMARY.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
consultar	integer	O ID da consulta. Este valor pode ser usado para unir várias outras tabelas e exibições do sistema.
segment	integer	O número do segmento. Uma consulta consiste em vários segmentos.
assignment	char(1)	O tipo de atribuição de partição entre os nós.
min_starttime	timestamp	O horário em UTC em que o processamento de partição foi iniciado.
max_endtime	timestamp	O horário em UTC em que o processamento de partição foi concluído.
min_duration	bigint	O tempo mínimo de processamento de partição usado por um nó para esta consulta (em microssegundos).
max_duration	bigint	O tempo máximo de processamento de partição usado por um nó para esta consulta (em microssegundos).
avg_duration	bigint	O tempo médio de processamento de partição usado por um nó para esta consulta (em microssegundos).

Nome da coluna	Tipo de dados	Descrição
total_partitions	integer	O número total de partições em uma tabela externa.
qualified_partitions	integer	O número total de partições qualificadas.
min_assigned_partitions	integer	O número mínimo de partições atribuídas em um nó.
max_assigned_partitions	integer	O número máximo de partições atribuídas em um nó.
avg_assigned_partitions	bigint	O número médio de partições atribuídas em um nó.

Consulta de exemplo

O exemplo a seguir obtém os detalhes da varredura de partição da última consulta concluída.

```
select query, segment, assignment, min_starttime, max_endtime, min_duration,
 avg_duration
from svl_s3partition_summary
where query = pg_last_query_id()
order by query,segment;
```

SVL_S3QUERY

Use a exibição SVL_S3QUERY para obter detalhes sobre as consultas de data lake nos níveis de segmento e de fatia do nó.

SVL_S3QUERY é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Note**
>
> SVL_S3QUERY contém apenas as consultas executadas nos principais clusters provisionados. Ele não contém consultas executadas em clusters de escalabilidade simultânea ou em namespaces sem servidor. Para acessar os planos de explicação das consultas executadas em clusters principais, clusters de escalabilidade simultânea e namespaces sem servidor, recomendamos usar a visualização de monitoramento SYS SYS_EXTERNAL_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou uma determinada entrada.
consultar	integer	O ID da consulta.
segment	integer	O número do segmento. Uma consulta consiste em vários segmentos e cada segmento consiste em uma ou mais etapas.
etapa	integer	A etapa da consulta que foi executada.
nó	integer	O número de nós.
slice	integer	A fatia dos dados com os quais determinado segmento foi executado.
starttime	timestamp	O horário (em UTC) de início da execução da consulta.
endtime	timestamp	O horário (em UTC) de término da execução da consulta.
elapsed	integer	Tempo decorrido (em microssegundos).

Nome da coluna	Tipo de dados	Descrição
external_table_name	char(136)	O formato interno do nome da tabela externa para a etapa de varredura do S3.
is_partitioned	char(1)	O valor true (**t**) indica que a tabela externa está particionada.
is_rrscan	char(1)	O valor true (**t**) indica que uma varredura restrita ao intervalo foi aplicada.
s3_scanned_rows	bigint	O número de linhas digitalizadas do Amazon S3 e enviadas para a camada do Redshift Spectrum. Em clusters RG provisionados, esse é o número de linhas lidas diretamente do Amazon S3 pelo leitor nativo do cluster, antes do envio do filtro.
s3_scanned_bytes	bigint	O número de bytes verificados do Amazon S3 e enviados para a camada Redshift Spectrum. Em clusters RG provisionados, esse é o tamanho total do intervalo de varredura em bytes processado pelo leitor nativo do cluster.
s3query_returned_rows	bigint	O número de linhas retornadas da camada do Redshift Spectrum para o cluster. Em clusters RG provisionados, esse é o número de linhas produzidas pela varredura da tabela externa após o envio do filtro.
s3query_returned_bytes	bigint	O número de bytes retornados da camada do Redshift Spectrum para o cluster. Em clusters RG provisionados, esse é o número de bytes produzidos pela varredura da tabela externa após o envio do filtro.
files	integer	O número de arquivos que foram processados para a etapa de varredura do S3 nesta fatia.

Nome da coluna	Tipo de dados	Descrição
splits	int	O número de divisões processadas nesta fatia. Com arquivos de dados grandes que podem ser divididos, por exemplo, arquivos de dados maiores do que 512 MB, o Redshift Spectrum tenta dividir os arquivos em várias solicitações do S3 para o processamento paralelo. Em clusters RG provisionados, esse é o número de intervalos de varredura consumidos pelo leitor nativo do cluster.
total_split_size	bigint	O tamanho total de todas as divisões processadas nesta fatia, em bytes. Em clusters RG provisionados, esse é o tamanho total de todos os intervalos de varredura consumidos, em bytes.
max_split_size	bigint	O tamanho máximo da divisão processada para esta fatia, em bytes. Em clusters RG provisionados, esse é o tamanho do maior intervalo de varredura consumido, em bytes.
total_retries	integer	O número total de novas tentativas para os arquivos processados. Em clusters do tipo de nó RG, essa coluna está obsoleta e contém -1. Para novas tentativas do cliente do Amazon S3 no RG, consulte `STL_S3CLIENT` .
max_retries	integer	O número máximo de novas tentativas para um arquivo individual processado. Em clusters do tipo de nó RG, essa coluna está obsoleta e contém -1. Para novas tentativas do cliente do Amazon S3 no RG, consulte `STL_S3CLIENT` .
max_request_duration	integer	A duração máxima de uma solicitação individual do Redshift Spectrum (em microssegundos). Em clusters do tipo de nó RG, essa coluna está obsoleta e contém -1.
avg_request_duration	double precision	A duração média das solicitações do Redshift Spectrum (em microssegundos). Em clusters do tipo de nó RG, essa coluna está obsoleta e contém -1.

Nome da coluna	Tipo de dados	Descrição
max_request_parallelism	integer	O número máximo de solicitações pendentes do Redshift Spectrum nesta fatia para a etapa de varredura do S3. Em clusters do tipo de nó RG, essa coluna está obsoleta e contém -1.
avg_request_parallelism	double precision	O número médio de solicitações paralelas do Redshift Spectrum nesta fatia para a etapa de varredura do S3. Em clusters do tipo de nó RG, essa coluna está obsoleta e contém -1.

Consulta de exemplo

O exemplo a seguir obtém os detalhes da etapa de varredura da última consulta concluída.

```
select query, segment, slice, elapsed, s3_scanned_rows, s3_scanned_bytes,
 s3query_returned_rows, s3query_returned_bytes, files
from svl_s3query
where query = pg_last_query_id()
order by query,segment,slice;
```

```
query | segment | slice | elapsed | s3_scanned_rows | s3_scanned_bytes |
 s3query_returned_rows | s3query_returned_bytes | files
------+---------+-------+---------+-----------------+------------------
+----------------------+------------------------+------
 4587 |       2 |     0 |   67811 |               0 |                0 |
      0 |                      0 |     0
 4587 |       2 |     1 |  591568 |          172462 |         11260097 |
   8513 |                 170260 |     1
 4587 |       2 |     2 |  216849 |               0 |                0 |
      0 |                      0 |     0
 4587 |       2 |     3 |  216671 |               0 |                0 |
      0 |                      0 |     0
```

SVL_S3QUERY_SUMMARY

Use a visualização SVL_S3QUERY_SUMMARY para obter um resumo de todas as consultas de data lake que foram executadas no sistema. A exibição SVL_S3QUERY_SUMMARY agrega os detalhes da SVL_S3QUERY no nível de segmento.

SVL_S3QUERY_SUMMARY é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_EXTERNAL_QUERY_DETAIL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Para SVCS_S3QUERY_SUMMARY, consulte SVCS_S3QUERY_SUMMARY.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou determinada entrada.
consultar	integer	O ID da consulta. Este valor pode ser usado para unir várias outras tabelas e exibições do sistema.
xid	bigint	O ID da transação.
pid	integer	O ID do processo.
segment	integer	O número do segmento. Uma consulta consiste em vários segmentos e cada segmento consiste em uma ou mais etapas.
etapa	integer	A etapa da consulta que foi executada.
starttime	timestamp	O horário (em UTC) de início da execução da consulta.
endtime	timestamp	O horário (em UTC) de término da consulta.

Nome da coluna	Tipo de dados	Descrição
elapsed	integer	O tempo (em microssegundos) que levou para a consulta ser executada.
aborted	integer	Se uma consulta for interrompida pelo sistema ou cancelada pelo usuário, essa coluna terá o valor **1**. Se a consulta foi executada até o final, essa coluna conterá **0**.
external_table_name	char(136)	O formato interno do nome externo da tabela para a varredura da tabela externa.
file_format	character(16)	O formato de arquivo dos dados da tabela externa.
is_partitioned	char(1)	O valor true (**t**) indica que a tabela externa está particionada.
is_rrscan	char(1)	O valor true (**t**) indica que uma varredura restrita ao intervalo foi aplicada.
is_nested	char(1)	O valor true (**t**) indica que o tipo de dados da coluna aninhada é acessado.
s3_scanned_rows	bigint	O número de linhas digitalizadas do Amazon S3 e enviadas para a camada do Redshift Spectrum. Em clusters RG provisionados, esse é o número total de linhas digitalizadas diretamente do Amazon S3 pelo leitor nativo do cluster em todas as fatias, antes do envio do filtro.
s3_scanned_bytes	bigint	O número de bytes verificados do Amazon S3 e enviados para a camada do Redshift Spectrum, com base em dados compactados. Em clusters RG provisionados, esse é o número total de bytes digitalizados diretamente do Amazon S3 pelo leitor nativo do cluster em todas as fatias, antes do envio do filtro.

Nome da coluna	Tipo de dados	Descrição
s3query_returned_rows	bigint	O número de linhas retornadas da camada do Redshift Spectrum para o cluster. Em clusters RG provisionados, esse é o número total de linhas retornadas pelo leitor nativo do cluster em todas as fatias, antes do envio do filtro.
s3query_returned_bytes	bigint	O número de bytes retornados da camada do Redshift Spectrum para o cluster. Uma grande quantidade de dados retornados ao Amazon Redshift pode afetar a performance do sistema. Em clusters RG provisionados, esse é o número total de bytes retornados pelo leitor nativo do cluster em todas as fatias, antes do envio do filtro.
files	integer	O número de arquivos que foram processados para esta consulta do data lake. Um número pequeno de arquivos limita os benefícios do processamento paralelo.
files_max	integer	O número máximo de arquivos processados em uma fatia.
files_avg	integer	O número médio de arquivos processados em uma fatia.
splits	int	O número de divisões processadas para este segmento. O número de divisões processadas nesta fatia. Com arquivos de dados grandes que podem ser divididos, por exemplo, arquivos de dados maiores do que 512 MB, o Redshift Spectrum tenta dividir os arquivos em várias solicitações do S3 para o processamento paralelo.
splits_max	int	O número máximo de divisões processadas nesta fatia.
splits_avg	int	O número médio de divisões processadas nesta fatia.
total_split_size	bigint	O tamanho total de todas as divisões processadas.

Nome da coluna	Tipo de dados	Descrição
max_split_size	bigint	O tamanho máximo da divisão processada, em bytes.
avg_split_size	bigint	O tamanho médio da divisão processada, em bytes.
total_retries	integer	O número total de novas tentativas para um arquivo individual processado. Em clusters do tipo de nó RG, essa coluna está obsoleta e contém -1. Para novas tentativas do cliente do Amazon S3 no RG, consulte `STL_S3CLIENT` .
max_retries	integer	O número máximo de novas tentativas para os arquivos processados. Em clusters do tipo de nó RG, essa coluna está obsoleta e contém -1. Para novas tentativas do cliente do Amazon S3 no RG, consulte `STL_S3CLIENT` .
max_request_duration	integer	A duração máxima de uma solicitação de arquivo individual (em microssegundos). As consultas de longa duração podem indicar um gargalo. Em clusters do tipo de nó RG, essa coluna está obsoleta e contém -1.
avg_request_duration	double precision	A duração média das solicitações de arquivos (em microssegundos). Em clusters do tipo de nó RG, essa coluna está obsoleta e contém -1.
max_request_parallelism	integer	O número máximo de solicitações paralelas em uma fatia para esta consulta do Redshift Spectrum. Em clusters do tipo de nó RG, essa coluna está obsoleta e contém -1.
avg_request_parallelism	double precision	O número médio de solicitações paralelas em uma fatia para esta consulta do Redshift Spectrum. Em clusters do tipo de nó RG, essa coluna está obsoleta e contém -1.

Nome da coluna	Tipo de dados	Descrição
total_slowdown_count	bigint	O número total de solicitações do Amazon S3 com um erro de desaceleração ocorrido durante a varredura da tabela externa. Em clusters do tipo de nó RG, essa coluna está obsoleta e contém -1.
max_slowdown_count	integer	O número máximo de solicitações do Amazon S3 com um erro de desaceleração que ocorreu durante a varredura de tabela externa em um slice. Em clusters do tipo de nó RG, essa coluna está obsoleta e contém -1.

Consulta de exemplo

O exemplo a seguir obtém os detalhes da etapa de varredura da última consulta concluída.

```
select query, segment, elapsed, s3_scanned_rows, s3_scanned_bytes,
 s3query_returned_rows, s3query_returned_bytes, files
from svl_s3query_summary
where query = pg_last_query_id()
order by query,segment;
```

```
query | segment | elapsed | s3_scanned_rows | s3_scanned_bytes | s3query_returned_rows
 | s3query_returned_bytes | files
------+---------+---------+-----------------+------------------+----------------------
+------------------------+------
 4587 |       2 |   67811 |               0 |                0 |                    0
 |                      0 |     0
 4587 |       2 |  591568 |          172462 |         11260097 |                 8513
 |                 170260 |     1
 4587 |       2 |  216849 |               0 |                0 |                    0
 |                      0 |     0
 4587 |       2 |  216671 |               0 |                0 |                    0
 |                      0 |     0
```

SVL_S3RETRIES

Use a visualização SVL_S3RETRIES para obter informações sobre por que uma consulta de data lake no Amazon S3 que usa o Amazon Redshift Spectrum falhou.

SVL_S3RETRIES é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

> **Clusters RG provisionados**
>
> `SVL_S3RETRIES` não está preenchido em clusters RG provisionados. A visualização existe e permanece consultável, mas não retorna linhas para consultas executadas em clusters RG, porque o RG usa o leitor nativo do cluster para consultar o Amazon S3 e não usa o modelo de nova tentativa de solicitação do Spectrum.
> Para obter detalhes de repetição e erro do cliente doAmazon S3 em clusters RG provisionados, use `STL_S3CLIENT` e `STL_S3CLIENT_ERROR`.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
consultar	integer	O ID da consulta.
segment	integer	O número do segmento. Uma consulta consiste em vários segmentos e cada segmento consiste em uma ou mais etapas. Os segmentos de uma consulta podem ser executados em paralelo. Cada segmento executa em um único processo.

Nome da coluna	Tipo de dados	Descrição
nó	integer	O número de nós.
slice	integer	A fatia dos dados com os quais determinado segmento foi executado .
eventtime	time stamp sem fuso horário	O horário (em UTC) de início da execução da etapa.
retries	integer	O número de tentativas para a consulta.
successful_fetches	integer	O número de vezes que os dados foram retornados.
file_size	bigint	O tamanho do arquivo em bytes.
location	texto	A localização da tabela.
message	texto	A mensagem de erro.

Consulta de exemplo

O exemplo a seguir recupera dados sobre consultas falhas do S3.

```
SELECT svl_s3retries.query, svl_s3retries.segment, svl_s3retries.node,
 svl_s3retries.slice, svl_s3retries.eventtime, svl_s3retries.retries,
svl_s3retries.successful_fetches, svl_s3retries.file_size,
 btrim((svl_s3retries."location")::text) AS "location",
 btrim((svl_s3retries.message)::text)
AS message FROM svl_s3retries;
```

SVL_SPATIAL_SIMPLIFY

Você pode consultar a visualização do sistema SVL_SPATIAL_SIMPLIFY para obter informações sobre objetos de geometria espacial simplificada usando o comando COPY. Quando você usa COPY em um shapefile, você pode especificar as opções de ingestão SIMPLIFY `tolerance`, SIMPLIFY AUTO e SIMPLIFY AUTO `max_tolerance`. O resultado da simplificação é resumido na exibição de sistema SVL_SPATIAL_SIMPLIFY.

Quando SIMPLIFY AUTO `max_tolerance` estiver definido, essa visualização contém uma linha para cada geometria que excedeu o tamanho máximo. Quando SIMPLIFY `tolerance` estiver definido, então uma linha para toda a operação COPY é armazenada. Esta linha faz referência ao ID da consulta COPY e à tolerância de simplificação especificada.

SVL_SPATIAL_SIMPLIFY é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_SPATIAL_SIMPLIFY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
consultar	integer	O ID da consulta (comando COPY) que gerou essa linha.
line_number	integer	Quando COPY `SIMPLIFY AUTO`for especificado, esse valor será o número de registro do registro simplificado no shapefile.
maximum_tolerance	double	O valor da tolerância de distância especificado no comando COPY. Este é o valor máximo de tolerância usando a opção `SIMPLIFY AUTO` ou o valor de tolerância fixa usando a opção `SIMPLIFY`.

Nome da coluna	Tipo de dados	Descrição
initial_size	integer	O tamanho em bytes do valor de dados GEOMETRY antes da simplificação.
simplified	char(1)	Quando a opção COPY SIMPLIFY AUTO é especificada, t se a geometria foi simplificada com sucesso, ou f caso contrário. A geometria pode não ser simplificada com sucesso se após a simplificação com a tolerância máxima dada seu tamanho ainda for maior do que o tamanho máximo da geometria.
final_size	integer	Quando a opção COPYSIMPLIFY AUTO for especificada, este é o tamanho em bytes da geometria após a simplific ação.
final_tol erance	double	

Consulta de exemplo

A consulta a seguir retorna a lista de registros que COPY simplificou.

```
SELECT * FROM svl_spatial_simplify WHERE query = pg_last_copy_id();
 query | line_number | maximum_tolerance | initial_size | simplified | final_size |
 final_tolerance
-------+-------------+-------------------+--------------+------------+------------
+----------------------
    20 |     1184704 |                -1 |      1513736 | t          |    1008808 |
 1.276386653895e-05
    20 |     1664115 |                -1 |      1233456 | t          |    1023584 |
 6.11707814796635e-06
```

SVL_SPECTRUM_SCAN_ERROR

Você pode consultar a visualização do sistema SVL_SPECTRUM_SCAN_ERROR para obter informações sobre erros de escaneamento do Redshift Spectrum.

SVL_SPECTRUM_SCAN_ERROR é visível a todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_EXTERNAL_QUERY_ERROR. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Exibe uma amostra dos erros registrados. O padrão são 10 entradas por consulta.

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou essa linha.
consultar	integer	O ID da consulta que gerou essa linha.
location	character(128)	A localização dos dados que estão sendo consultados.
rowid	character(128)	A localização do erro no arquivo. As partes de `rowid` são separadas com `:` (dois-pontos) e as outras partes podem ser adicionadas no futuro. `row_offset :row_group :row_id` Um row_offset é o deslocamento (em bytes) da linha dentro do arquivo e é definido como `-1` para formatos de arquivo não compatíveis. Uma tabela é dividida em row_groups, e cada grupo tem linhas com row_ids distintos .
colname	character(128)	O nome da coluna retornada pela consulta.
original_ value	character(128)	Valor original consultado.

Nome da coluna	Tipo de dados	Descrição
modified_value	character(128)	Valor modificado retornado com base na opção de configuração de tratamento de dados especificada na consulta.
Acionador	character(128)	Opção de tratamento de dados especificada na consulta.
ação	character(128)	Ação associada à opção de tratamento de dados especific ada na consulta.
action_value	character(128)	Valor do parâmetro de ação associado à opção de tratamento de dados especificada na consulta.
error_code	integer	Código do resultado da opção de tratamento de dados especificada na consulta.

Consulta de exemplo

A consulta a seguir retorna a lista de linhas para as quais as operações de manipulação de dados foram executadas.

```
SELECT * FROM svl_spectrum_scan_error;
```

A consulta retorna resultados semelhantes aos resultados a seguir.

```
  userid  query     location                                        rowid   colname
         original_value             modified_value       trigger          action
                 action_valueerror_code
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:0     league_name
         Barclays Premier League    Barclays Premier Lea UNSPECIFIED      TRUNCATE
                                     156
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:0     league_nspi
          34595                      32767                UNSPECIFIED
OVERFLOW_VALUE                                  199
```

```
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:1     league_nspi
          34151                      32767               UNSPECIFIED
OVERFLOW_VALUE                                  199
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:2     league_name
         Barclays Premier League    Barclays Premier Lea UNSPECIFIED      TRUNCATE
                                    156
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:2     league_nspi
          33223                      32767               UNSPECIFIED
OVERFLOW_VALUE                                  199
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:3     league_name
         Barclays Premier League    Barclays Premier Lea UNSPECIFIED      TRUNCATE
                                    156
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:3     league_nspi
          32808                      32767               UNSPECIFIED
OVERFLOW_VALUE                                  199
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:4     league_nspi
          32790                      32767               UNSPECIFIED
OVERFLOW_VALUE                                  199
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:5     league_name
         Spanish Primera Division   Spanish Primera Divi UNSPECIFIED      TRUNCATE
                                    156
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:6     league_name
         Spanish Primera Division   Spanish Primera Divi UNSPECIFIED      TRUNCATE
                                    156
```

SVL_STATEMENTTEXT

Use a exibição SVL_STATEMENTTEXT para obter um registro completo de todos os comandos SQL que foram executados no sistema.

A exibição SVL_STATEMENTTEXT contém a união de todas as linhas nas tabelas STL_DDLTEXT, STL_QUERYTEXT e STL_UTILITYTEXT. Essa exibição também inclui uma junção com a tabela STL_QUERY.

SVL_STATEMENTTEXT é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_QUERY_HISTORY. Os dados na exibição de monitoramento SYS são

formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário que gerou a entrada.
xid	bigint	O ID da transação associada à instrução.
pid	integer	O ID de processo para a instrução.
rótulo	character(320)	O nome do arquivo usado para executar a consulta ou um rótulo definido com o comando SET QUERY_GROUP. Se a consulta não for baseada em arquivos ou o parâmetro QUERY_GROUP não estiver definido, o valor deste campo será branco.
starttime	timestamp	O horário exato do início da execução da instrução, com 6 dígitos de precisão para as frações de segundo. Por exemplo, .: **`2009-06-12 11:29:19.131358`**
endtime	timestamp	O horário exato do término da execução da instrução, com 6 dígitos de precisão para as frações de segundo. Por exemplo, .: **`2009-06-12 11:29:19.193640`**
sequence	integer	Quando uma única instrução contém mais de 200 caracteres, são registradas linhas adicionais para essa instrução. O valor 0 da sequência é a primeira linha, 1 é a segunda, e assim por diante.
type	varchar(10)	O tipo da instrução SQL: **`QUERY`**, **`DDL`** ou **`UTILITY`**.
texto	character(200)	O texto em SQL, em incrementos de 200 caracteres. Esse campo pode conter caracteres especiais como barra invertida (\\) e nova linha (\n).

Consulta de exemplo

A consulta a seguir retorna as instruções em DDL que foram executadas em 16 de junho de 2009:

```
select starttime, type, rtrim(text) from svl_statementtext
where starttime like '2009-06-16%' and type='DDL' order by starttime asc;

starttime                  | type |              rtrim
---------------------------|------|--------------------------------
2009-06-16 10:36:50.625097 | DDL  | create table ddltest(c1 int);
2009-06-16 15:02:16.006341 | DDL  | drop view alltickitjoin;
2009-06-16 15:02:23.65285  | DDL  | drop table sales;
2009-06-16 15:02:24.548928 | DDL  | drop table listing;
2009-06-16 15:02:25.536655 | DDL  | drop table event;
...
```

Reconstrução de SQL armazenado

Para reconstruir o SQL armazenado na coluna `text` de SVL_STATEMENTTEXT, execute uma instrução SELECT para criar SQL a partir de uma ou mais partes na coluna `text`. Antes de executar o SQL reconstruído, substitua os caracteres especiais por uma nova linha (\n). O resultado da instrução SELECT a seguir são linhas de SQL reconstruído no campo `query_statement`.

```
select LISTAGG(CASE WHEN LEN(RTRIM(text)) = 0 THEN text ELSE RTRIM(text) END, '')
 within group (order by sequence) AS query_statement
from SVL_STATEMENTTEXT where pid=pg_backend_pid();
```

Por exemplo, a consulta a seguir seleciona três colunas. A consulta em si tem mais de 200 caracteres e é armazenada em partes emSVL_STATEMENTTEXT.

```
select
1 AS a01234567890123456789012345678901234567890123456789012345678901234567890,
2 AS b01234567890123456789012345678901234567890123456789012345678901234567890,
3 AS b01234567890123456789012345678901234567890123456789012345678901234
FROM stl_querytext;
```

Nesse exemplo, a consulta é armazenada em duas partes (linhas) na coluna `text` de SVL_STATEMENTTEXT.

```
select sequence, text from SVL_STATEMENTTEXT where pid = pg_backend_pid() order by
 starttime, sequence;
```

```
 sequence |
                text

----------
+-----------------------------------------------------------------------------------------------
        0 | select\n1 AS
 a01234567890123456789012345678901234567890123456789012345678901234567890,\n2 AS
 b01234567890123456789012345678901234567890123456789012345678901234567890,\n3 AS
 b01234567890123456789012345678901234567890123456789012345678901234
        1 | \nFROM stl_querytext;
```

Para reconstruir o SQL armazenado em STL_STATEMENTTEXT, execute o seguinte SQL.

```
select LISTAGG(CASE WHEN LEN(RTRIM(text)) = 0 THEN text ELSE RTRIM(text) END, '')
 within group (order by sequence) AS text
from SVL_STATEMENTTEXT where pid=pg_backend_pid();
```

Para usar o SQL reconstruído resultante em seu cliente, substitua os caracteres especiais por uma nova linha (\n).

```
                     text

-----------------------------------------------------------------------------------------------
 select\n1 AS a01234567890123456789012345678901234567890123456789012345678901234567890,
\n2 AS b01234567890123456789012345678901234567890123456789012345678901234567890,\n3 AS
 b01234567890123456789012345678901234567890123456789012345678901234\nFROM stl_querytext;
```

SVL_STORED_PROC_CALL

É possível consultar a visualização de sistema SVL_STORED_PROC_CALL para obter informações sobre as chamadas de procedimento armazenado, inclusive o horário de início, o horário de término e se uma chamada foi cancelada. Cada chamada de procedimento armazenado recebe um ID de consulta.

SVL_STORED_PROC_CALL está visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_PROCEDURE_CALL. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário cujos privilégios foram usados para executar a instrução. Se essa chamada tiver sido aninhada dentro de um procedimento armazenado DEFINIDOR DE SEGURANÇA, esse será o userid do proprietário desse procedimento armazenado.
session_userid	integer	O ID do usuário que criou a sessão e é o invocador da chamada de procedimento armazenado de nível superior.
consultar	integer	O ID da consulta da chamada de procedimento.
rótulo	character(320)	O nome do arquivo usado para executar a consulta ou um rótulo definido com o comando SET QUERY_GROUP. Se a consulta não for baseada em arquivos ou o parâmetro QUERY_GROUP não estiver definido, o valor deste campo será o padrão.
xid	bigint	O ID da transação.
pid	integer	O ID do processo. Normalmente, todas as consultas em uma sessão são executadas no mesmo processo, portanto, esse valor geralmente permanece constante se você executar uma série de consultas na mesma sessão. Após certos eventos internos, o Amazon Redshift pode reiniciar uma sessão ativa e atribuir um novo valor pid. Para obter mais informações, consulte STL_RESTARTED_SESSIONS.

Nome da coluna	Tipo de dados	Descrição
banco de dados	character(32)	O nome do banco de dados ao qual o usuário estava conectado quando a consulta foi emitida.
querytxt	character(4000)	O texto real da consulta de chamada de procedimento.
starttime	timestamp	O horário, em UTC, do início da execução da consulta, com 6 dígitos de precisão fracionária para segundos, por exemplo: **2009-06-12 11:29:19.131358.**
endtime	timestamp	O horário, em UTC, do término da execução da consulta, com 6 dígitos de precisão fracionária para segundos, por exemplo: **2009-06-12 11:29:19.131358.**
aborted	integer	Se um procedimento armazenado tiver sido interromp ido pelo sistema ou cancelado pelo usuário, essa coluna conterá o valor 1. Se a chamada for concluída, essa coluna terá o valor 0.
from_sp_c all	integer	Se a chamada de procedimento tiver sido invocada por outra chamada de procedimento, essa coluna terá o ID de consulta da outra chamada. Caso contrário, o campo será NULL.

Consulta de exemplo

A consulta a seguir retorna o tempo gasto em ordem decrescente e o status de conclusão para chamadas de procedimento armazenado no dia anterior.

```
select query, datediff(seconds, starttime, endtime) as elapsed_time, aborted,
 trim(querytxt) as call from svl_stored_proc_call where starttime >= getdate() -
 interval '1 day' order by 2 desc;

  query | elapsed_time | aborted |                                          call
--------+--------------+---------
+-----------------------------------------------------------------------------
   4166 |            7 |       0 | call search_batch_status(35,'succeeded');
```

```
2433 |            3 |       0 | call test_batch (123456)
1810 |            1 |       0 | call prod_benchmark (123456)
1836 |            1 |       0 | call prod_testing (123456)
1808 |            1 |       0 | call prod_portfolio ('N', 123456)
1816 |            1 |       1 | call prod_portfolio ('Y', 123456)
```

SVL_STORED_PROC_MESSAGES

É possível consultar a visualização do sistema SVL_STORED_PROC_MESSAGES para obter informações sobre mensagens de procedimento armazenado. As mensagens geradas são registradas mesmo se a chamada de procedimento armazenado for cancelada. Cada chamada de procedimento armazenado recebe um ID de consulta. Para obter mais informações sobre como definir o nível mínimo para mensagens registradas em log, consulte stored_proc_log_min_messages.

SVL_STORED_PROC_MESSAGES está visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_PROCEDURE_MESSAGES. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
userid	integer	O ID do usuário cujos privilégios foram usados para executar a instrução. Se essa chamada tiver sido aninhada dentro de um procedimento armazenado DEFINIDOR DE SEGURANÇA, esse será o userid do proprietário desse procedimento armazenado.
session_userid	integer	O ID do usuário que criou a sessão e é o invocador da chamada de procedimento armazenado de nível superior.
pid	integer	O ID do processo.

Nome da coluna	Tipo de dados	Descrição
xid	bigint	O ID da transação da consulta de chamada de procedimento.
consultar	integer	O ID da consulta da chamada de procedimento.
recordtime	timestamp	A hora em UTC em que a mensagem foi gerada.
loglevel	integer	O valor numérico do nível de log da mensagem gerada. Valores possíveis: 20 – para LOG 30 – para INFO 40 – para NOTICE 50 – para WARNING 60 – para EXCEPTION
loglevel_text	character(10)	O nível de log que corresponde ao valor numérico em loglevel. Valores possíveis: LOG, INFO, NOTICE, WARNING e EXCEPTION.
message	character(1024)	O texto da mensagem gerada.
linenum	integer	O número da linha da instrução gerada.
querytext	character(500)	O texto real da consulta de chamada de procedimento.
rótulo	character(320)	O nome do arquivo usado para executar a consulta ou um rótulo definido com o comando SET QUERY_GROUP. Se a consulta não for baseada em arquivos ou o parâmetro QUERY_GROUP não estiver definido, o valor deste campo será o padrão.
aborted	integer	Se um procedimento armazenado tiver sido interrompido pelo sistema ou cancelado pelo usuário, essa coluna conterá o valor 1. Se a chamada for concluída, essa coluna terá o valor 0.
message_xid	bigint	O ID da transação da mensagem gerada.

Consulta de exemplo

As instruções SQL a seguir mostram como usar SVL_STORED_PROC_MESSAGES para revisar mensagens geradas.

```
-- Create and run a stored procedure
CREATE OR REPLACE PROCEDURE test_proc1(f1 int) AS
$$
BEGIN
    RAISE INFO 'Log Level: Input f1 is %',f1;
    RAISE NOTICE 'Notice Level: Input f1 is %',f1;
    EXECUTE 'select invalid';
    RAISE NOTICE 'Should not print this';

EXCEPTION WHEN OTHERS THEN
     raise exception 'EXCEPTION level: Exception Handling';
END;
$$ LANGUAGE plpgsql;

-- Call this stored procedure
CALL test_proc1(2);

-- Show raised messages with level higher than INFO
SELECT query, recordtime, loglevel, loglevel_text, trim(message) as message, aborted
 FROM svl_stored_proc_messages
  WHERE loglevel > 30 AND query = 193 ORDER BY recordtime;

 query |         recordtime         | loglevel | loglevel_text |               message
              | aborted
-------+----------------------------+----------+---------------
+-------------------------------------+---------
   193 | 2020-03-17 23:57:18.277196 |       40 | NOTICE        | Notice Level: Input f1
 is 2          |       1
   193 | 2020-03-17 23:57:18.277987 |       60 | EXCEPTION     | EXCEPTION level:
 Exception Handling |       1
(2 rows)

-- Show raised messages at EXCEPTION level
SELECT query, recordtime, loglevel, loglevel_text, trim(message) as message, aborted
 FROM svl_stored_proc_messages
  WHERE loglevel_text = 'EXCEPTION' AND query = 193 ORDER BY recordtime;

 query |         recordtime         | loglevel | loglevel_text |               message
              | aborted
```

```
-------+----------------------------+----------+---------------
+-------------------------------------+---------
   193 | 2020-03-17 23:57:18.277987 |       60 | EXCEPTION     | EXCEPTION level:
 Exception Handling |       1
```

As instruções SQL a seguir mostram como usar SVL_STORED_PROC_MESSAGES para revisar mensagens geradas com a opção SET ao criar um procedimento armazenado. Como test_proc() tem um nível de log mínimo de NOTICE, apenas as mensagens de nível NOTICE, WARNING e EXCEPTION são registradas em log em SVL_STORED_PROC_MESSAGES.

```
-- Create a stored procedure with minimum log level of NOTICE
CREATE OR REPLACE PROCEDURE test_proc() AS
$$
BEGIN
    RAISE LOG 'Raise LOG messages';
    RAISE INFO 'Raise INFO messages';
    RAISE NOTICE 'Raise NOTICE messages';
    RAISE WARNING 'Raise WARNING messages';
    RAISE EXCEPTION 'Raise EXCEPTION messages';
    RAISE WARNING 'Raise WARNING messages again'; -- not reachable
END;
$$ LANGUAGE plpgsql SET stored_proc_log_min_messages = NOTICE;

-- Call this stored procedure
CALL test_proc();

-- Show the raised messages
SELECT query, recordtime, loglevel_text, trim(message) as message, aborted FROM
 svl_stored_proc_messages
  WHERE query = 149 ORDER BY recordtime;

 query |         recordtime         | loglevel_text |          message         |
 aborted
-------+----------------------------+---------------+--------------------------
+---------
   149 | 2020-03-16 21:51:54.847627 | NOTICE        | Raise NOTICE messages    |
 1
   149 | 2020-03-16 21:51:54.84766  | WARNING       | Raise WARNING messages   |
 1
   149 | 2020-03-16 21:51:54.847668 | EXCEPTION     | Raise EXCEPTION messages |
 1
(3 rows)
```

SVL_TERMINATE

Registra a hora em que um usuário cancela ou encerra um processo.

SELECT PG_TERMINATE_BACKEND(pid), SELECT PG_CANCEL_BACKEND(pid) e CANCEL pid cria uma entrada de log em SVL_TERMINATE.

SVL_ TERMINATE só permanece visível para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_QUERY_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
pid	integer	O ID do processo cancelado ou terminado.
eventtime	timestamp	A hora em que o processo é cancelado ou terminado.
userid	integer	O ID do usuário que executa o comando.
type	string	O tipo de encerramento. Pode ser CANCEL ou TERMINATE.

O comando a seguir mostra a última consulta cancelada.

```
select * from svl_terminate order by eventtime desc limit 1;
 pid  |         eventtime          | userid |  type
------+----------------------------+--------+--------
 8324 | 2020-03-24 09:42:07.298937 |      1 | CANCEL
(1 row)
```

SVL_UDF_LOG

Registra mensagens de erro e aviso definidas pelo sistema geradas durante a execução da função definida pelo usuário (UDF).

SVL_UDF_LOG permanece visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_UDF_LOG. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
consultar	bigint	O ID da consulta. Este ID pode ser usado para unir várias outras tabelas e exibições do sistema.
message	char(4096)	A mensagem gerada pela função.
created	timestamp	O horário em que o log foi criado.
traceback	char(4096)	Se estiver disponível, esse valor fornece um rastreamento de pilha para a UDF. Para obter mais informações, consulte traceback na Python Standard Library.
funcname	character(256)	O nome da UDF que está em execução.
nó	integer	O nó onde a mensagem foi gerada.
slice	integer	A fatia onde a mensagem foi gerada.

Nome da coluna	Tipo de dados	Descrição
seq	integer	A sequência das mensagens na fatia.

Consultas de exemplo

O exemplo a seguir mostra como as UDFs tratam os erros definidos pelo sistema. O primeiro bloco mostra a definição de uma função de UDF que retorna o inverso de um argumento. Quando você executa a função e fornece um argumento de valor 0, como mostra o segundo bloco, a função retorna um erro. A terceira instrução lê a mensagem de erro que é registrada em SVL_UDF_LOG

```
-- Create a function to find the inverse of a number

CREATE OR REPLACE FUNCTION  f_udf_inv(a int)
  RETURNS float IMMUTABLE
AS $$
   return 1/a
$$ LANGUAGE plpythonu;

-- Run the function with a 0 argument to create an error
Select f_udf_inv(0) from sales;

-- Query SVL_UDF_LOG to view the message

Select query, created, message::varchar
from svl_udf_log;

 query |          created           | message
-------+----------------------------
+-----------------------------------------------------------
  2211 | 2015-08-22 00:11:12.04819  | ZeroDivisionError: long division or modulo by
 zero\nNone
```

O exemplo a seguir adiciona o registro em log e uma mensagem de aviso à UDF para que operações de divisão por zero gerem uma mensagem de aviso em vez de interromper o processamento com uma mensagem de erro.

```
-- Create a function to find the inverse of a number and log a warning
```

```
CREATE OR REPLACE FUNCTION f_udf_inv_log(a int)
  RETURNS float IMMUTABLE
 AS $$
  import logging
  logger = logging.getLogger() #get root logger
  if a==0:
    logger.warning('You attempted to divide by zero.\nReturning zero instead of error.
\n')
    return 0
  else:
     return 1/a
$$ LANGUAGE plpythonu;
```

O exemplo a seguir executa a função e, em seguida, consulta a exibição SVL_UDF_LOG para ver a mensagem.

```
-- Run the function with a 0 argument to trigger the warning
Select f_udf_inv_log(0) from sales;

-- Query SVL_UDF_LOG to view the message

Select query, created, message::varchar
from svl_udf_log;

query |          created           | message
------+----------------------------+---------------------------------
    0 | 2015-08-22 00:11:12.04819  | You attempted to divide by zero.
                                     Returning zero instead of error.
```

SVL_USER_INFO

Você pode recuperar dados sobre os usuários do banco de dados Amazon Redshift com a exibição SVL_USER_INFO.

SVL_USER_INFO só permanece visível para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
usename	texto	O nome do usuário da função.
usesysid	integer	O ID do usuário.
usecreatedb	booleano	Um valor que indica se o usuário tem permissões para criar bancos de dados.
usesuper	booleano	Um valor que indica se o usuário é um superusuário.
usecatupd	booleano	Um valor que indica se o usuário pode atualizar catálogos do sistema.
useconnlimit	texto	O número de conexões que o usuário pode abrir.
syslogaccess	texto	Um valor que indica se o usuário tem acesso aos logs do sistema. Os dois valores possíveis são `RESTRICTED` e `UNRESTRICTED`. `RESTRICTED` significa que os usuários que não são superusuários podem ver seus próprios registros. `UNRESTRICTED` significa que o usuário que não é superusuário pode ver todos os registros nas exibições e tabelas do sistema para as quais ele tem privilégios `SELECT`.
last_ddl_ts	timestamp	O registro de data e hora da última linguagem de definição de dados (DDL) cria uma instrução executada pelo usuário.
sessiontimeout	integer	O tempo máximo em segundos em que uma sessão permanece inativa ou ociosa antes do tempo limite. 0 indica que nenhum tempo limite está definido. Para obter informações sobre a configuração de tempo limite ocioso ou inativo do cluster, consulte “Cotas e limites no Amazon Redshift” no Guia de gerenciamento de clusters do Amazon Redshift.
external_id	texto	O identificador exclusivo do usuário no provedor de identidades de terceiros.

Consultas de exemplo

O comando a seguir recupera informações de usuário de SVL_USER_INFO.

```
SELECT * FROM SVL_USER_INFO;
```

SVL_VACUUM_PERCENTAGE

A exibição SVL_VACUUM_PERCENTAGE relata a porcentagem de blocos de dados alocados para uma tabela após a realização de um vacuum. Essa porcentagem mostra o espaço em disco que foi recuperado. Consulte VACUUM para obter mais informações sobre o utilitário de limpeza.

SVL_VACUUM_PERCENTAGE é visível somente para superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Alguns ou todos os dados nessa tabela também podem ser encontrados na exibição de monitoramento SYS SYS_VACUUM_HISTORY. Os dados na exibição de monitoramento SYS são formatados para serem mais fáceis de usar e compreender. É recomendável usar a exibição de monitoramento SYS nas consultas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
xid	bigint	O ID da transação para a instrução de limpeza.
table_id	integer	O ID da tabela que foi limpada.
percentage	bigint	A porcentagem de blocos de dados após uma limpeza (em relação ao número de blocos existentes na tabela antes da execução da limpeza).

Consulta de exemplo

A consulta a seguir mostra a porcentagem para uma operação específica na tabela 100238:

```
select * from svl_vacuum_percentage
```

```
where table_id=100238 and xid=2200;

xid  | table_id | percentage
-----+----------+------------
1337 |   100238 |         60
(1 row)
```

Após esta operação de limpeza, a tabela contém 60 por cento dos blocos originais.

Tabelas de catálogo do sistema

Tópicos

- PG_ATTRIBUTE_INFO
- PG_CLASS_INFO
- PG_DATABASE_INFO
- PG_DEFAULT_ACL
- PG_EXTERNAL_SCHEMA
- PG_LIBRARY
- PG_PROC_INFO
- PG_STATISTIC_INDICATOR
- PG_TABLE_DEF
- PG_USER_INFO
- Consultar as tabelas de catálogo

Os catálogos de sistema armazenam metadados de esquema, como informações sobre tabelas e colunas. As tabelas de catálogo do sistema têm o prefixo PG.

As tabelas do catálogo PostgreSQL padrão são acessíveis aos usuários do Amazon Redshift. Para mais informações sobre catálogos de sistema PostgreSQL, consulte Tabelas de sistema PostgreSQL

PG_ATTRIBUTE_INFO

PG_ATTRIBUTE_INFO é uma visualização do sistema Amazon Redshift construída na tabela do catálogo PostgreSQL PG_ATTRIBUTE e na tabela do catálogo interno PG_ATTRIBUTE_ACL. A

PG_ATTRIBUTE_INFO inclui detalhes sobre colunas de uma tabela ou exibição, incluindo listas de controle de acesso a colunas, se houver.

Colunas da tabela

Além das colunas em PG_ATTRIBUTE, o PG_ATTRIBUTE_INFO mostra a coluna a seguir.

Nome da coluna	Tipo de dados	Descrição
attacl	aclitem[]	Os privilégios de acesso em nível de coluna, se houver, que foram concedidos especificamente nesta coluna.

PG_CLASS_INFO

PG_CLASS_INFO é uma visualização do sistema Amazon Redshift construída nas tabelas do catálogo PostgreSQL PG_CLASS e PG_CLASS_EXTENDED. PG_CLASS_INFO inclui detalhes sobre o horário de criação da tabela e o estilo de distribuição atual. Para obter mais informações, consulte Distribuição de dados para otimização de consultas.

PG_CLASS_INFO é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

PG_CLASS_INFO mostra as seguintes colunas, além das colunas em PG_CLASS. A coluna `oid` em PG_CLASS é chamada de `reloid` na tabela PG_CLASS_INFO.

Nome da coluna	Tipo de dados	Descrição
relcreationtime	timestamp	Horário (UTC) no qual a tabela foi criada.
releffectivediststyle	integer	O estilo de distribuição de uma tabela ou, se a tabela usa distribuição automática, o estilo de distribuição atual atribuído pelo Amazon Redshift.

A coluna RELEFFECTIVEDISTSTYLE em PG_CLASS_INFO indica o estilo de distribuição atual da tabela. Se a tabela usar distribuição automática, RELEFFECTIVEDISTSTYLE será 10, 11 ou 12, o que indica se o estilo de distribuição efetivo é AUTO (ALL) ou AUTO (EVEN) ou AUTO (KEY). Se a tabela usar distribuição automática, o estilo de distribuição poderá mostrar inicialmente AUTO (ALL) e mudar para AUTO (EVEN) quando a tabela crescer, ou AUTO (KEY) se uma coluna for útil como uma chave de distribuição.

A tabela a seguir fornece o estilo de distribuição para cada valor na coluna RELEFFECTIVEDISTSTYLE:

RELEFFECTIVEDISTSTYLE	Estilo de distribuição atual
0	EVEN
1	KEY
8	ALL
10	AUTO (ALL)
11	AUTO (EVEN)
12	AUTO (KEY)

Exemplo

A consulta a seguir retorna o estilo de distribuição atual das tabelas no catálogo.

```
select reloid as tableid,trim(nspname) as schemaname,trim(relname) as
 tablename,reldiststyle,releffectivediststyle,
CASE WHEN "reldiststyle" = 0 THEN 'EVEN'::text
     WHEN "reldiststyle" = 1 THEN 'KEY'::text
     WHEN "reldiststyle" = 8 THEN 'ALL'::text
     WHEN "releffectivediststyle" = 10 THEN 'AUTO(ALL)'::text
     WHEN "releffectivediststyle" = 11 THEN 'AUTO(EVEN)'::text
     WHEN "releffectivediststyle" = 12 THEN 'AUTO(KEY)'::text ELSE '<<UNKNOWN>>'::text
 END as diststyle,relcreationtime
from pg_class_info a left join pg_namespace b on a.relnamespace=b.oid;
```

```
 tableid | schemaname | tablename | reldiststyle | releffectivediststyle | diststyle  |
      relcreationtime
---------+------------+-----------+--------------+-----------------------+------------
+---------------------------
 3638033 | public     | customer  |            0 |                     0 | EVEN       |
 2019-06-13 15:02:50.666718
 3638037 | public     | sales     |            1 |                     1 | KEY        |
 2019-06-13 15:03:29.595007
 3638035 | public     | lineitem  |            8 |                     8 | ALL        |
 2019-06-13 15:03:01.378538
 3638039 | public     | product   |            9 |                    10 | AUTO(ALL)  |
 2019-06-13 15:03:42.691611
 3638041 | public     | shipping  |            9 |                    11 | AUTO(EVEN) |
 2019-06-13 15:03:53.69192
 3638043 | public     | support   |            9 |                    12 | AUTO(KEY)  |
 2019-06-13 15:03:59.120695
(6 rows)
```

PG_DATABASE_INFO

PG_DATABASE_INFO é uma visualização do sistema Amazon Redshift que estende a tabela do catálogo PostgreSQL PG_DATABASE.

PG_DATABASE_INFO fica visível para todos os usuários.

Colunas da tabela

PG_DATABASE_INFO contém as seguintes colunas além das colunas em PG_DATABASE. A coluna `oid` em PG_DATABASE é chamada de `datid` na tabela PG_DATABASE_INFO. Para obter mais informações, consulte a Documentação do PostgreSQL.

Nome da coluna	Tipo de dados	Descrição
datid	oid	O object identifier (OID – Identificador de objeto) usado internamente por tabelas de sistema.
datconnlimit	texto	O número máximo de conexões simultâneas que podem ser feitas nesse banco de dados. O valor -1 indica que não há limite.

PG_DEFAULT_ACL

Armazena informações sobre privilégios de acesso padrão. Para mais informações sobre privilégios de acesso padrão, consulte ALTER DEFAULT PRIVILEGES.

PG_DEFAUL_ACL é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
defacluser	integer	ID do usuário a que os privilégios listados são aplicados.
defaclnamespace	oid	O ID de objeto do esquema onde os privilégios padrão são aplicados. Se nenhum esquema for especificado, o valor padrão será 0.
defaclobjtype	caractere	O tipo de objeto ao qual os privilégios padrão são aplicados. Os valores válidos são os seguintes: • r: relation (tabela ou visualização) • f: function • p: stored procedure
defaclacl	aclitem[]	Uma string que define os privilégios padrão para o usuário ou o grupo de usuários especificado e o tipo de objeto. Se os privilégios forem concedidos a um usuário, a string estará na seguinte forma: *{ username=privilegestring/grantor }* Nome de usuário do O nome do usuário ao qual privilégios são concedidos. Se o nome do usuário for omitido, os privilégios serão concedidos a PUBLIC.

Nome da coluna	Tipo de dados	Descrição
		Se os privilégios forem concedidos a um grupo de usuários, a string estará na seguinte forma: *{ "group groupname=privilegestring/grantor" }* privilegestring Uma string que especifica quais privilégios são concedidos. Os valores válidos são: • a: INSERT (anexo). • r: SELECT (leitura). • w: UPDATE (gravação) • d; DELETE. • R: RULE. • x: concede o privilégio de criar uma restrição de chave externa (REFERENCES). • t: TRIGGER. • X: EXECUTE. • U: USAGE. • C: CREATE. • T: CREATE TEMP. • D: DROP. • P: TRUNCATE. • A: ALTER. • *: indica que o usuário que recebe o privilégio anterior pode, por sua vez, conceder o mesmo privilégio a outros (WITH GRANT OPTION).

Nome da coluna	Tipo de dados	Descrição
		Uma string contendo todos os caracteres do código de privilégio, em ordem de acordo com a posição da máscara de bits, é semelhante a “arwdRxtXUCTDPA”. grantor O nome do usuário que concedeu os privilégios. O exemplo a seguir indica que o usuário `admin` concedeu todos os privilégios, inclusive WITH GRANT OPTION, ao usuário `dbuser`. `dbuser=r*a*w*d*x*X*/admin`

Exemplo

A consulta a seguir retorna todos os privilégios padrão definidos para o banco de dados.

```
select pg_get_userbyid(d.defacluser) as user,
n.nspname as schema,
case d.defaclobjtype when 'r' then 'tables' when 'f' then 'functions' end
as object_type,
array_to_string(d.defaclacl, ' + ')  as default_privileges
from pg_catalog.pg_default_acl d
left join pg_catalog.pg_namespace n on n.oid = d.defaclnamespace;

 user  | schema | object_type |              default_privileges
-------+--------+-------------+-------------------------------------------------------
 admin | tickit | tables      | user1=r/admin + "group group1=a/admin" + user2=w/admin
```

O resultado no exemplo anterior mostra que, para todas as tabelas novas criadas pelo usuário `admin` no esquema `tickit`, `admin` concede privilégios SELECT a `user1`, privilégios INSERT a `group1` e privilégios UPDATE a `user2`.

PG_EXTERNAL_SCHEMA

Armazena informações sobre os esquemas externos.

PG_EXTERNAL_SCHEMA é visível para todos os usuários. Os superusuários podem ver todas as linhas e os usuários comuns podem ver somente os metadados aos quais eles têm acesso. Para obter mais informações, consulte CREATE EXTERNAL SCHEMA.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
esoid	oid	ID do esquema externo.
eskind	integer	Tipo de esquema externo.
esdbname	texto	O nome do banco de dados externo.
esoptions	texto	As opções do esquema externo.

Exemplo

O exemplo a seguir mostra os detalhes dos esquemas externos.

```
select esoid, nspname as schemaname, nspowner, esdbname as external_db, esoptions
from pg_namespace a,pg_external_schema b where a.oid=b.esoid;

esoid  | schemaname      | nspowner | external_db | esoptions

-------+-----------------+----------+-------------
+-------------------------------------------------------------
100134 | spectrum_schema |      100 | spectrum_db |
 {"IAM_ROLE":"arn:aws:iam::123456789012:role/mySpectrumRole"}
100135 | spectrum        |      100 | spectrumdb  |
 {"IAM_ROLE":"arn:aws:iam::123456789012:role/mySpectrumRole"}
100149 | external        |      100 | external_db |
 {"IAM_ROLE":"arn:aws:iam::123456789012:role/mySpectrumRole"}
```

PG_LIBRARY

Armazena informações sobre bibliotecas definidas pelo usuário.

PG_LIBRARY é visível para todos os usuários. Os superusuários podem ver todas as linhas; usuários regulares podem ver somente seus próprios dados. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
name	name	Nome da biblioteca.
language_oid	oid	Reservada para uso do sistema.
file_store_id	integer	Reservada para uso do sistema.
proprietário	integer	ID de usuário do proprietário da biblioteca.

Exemplo

O exemplo a seguir retorna informações de bibliotecas instaladas pelo usuário.

```
select * from pg_library;

name       | language_oid | file_store_id | owner
-----------+--------------+---------------+------
f_urlparse |       108254 |          2000 |   100
```

PG_PROC_INFO

PG_PROC_INFO é uma visualização do sistema Amazon Redshift construída na tabela do catálogo PostgreSQL PG_PROC e na tabela do catálogo interno PG_PROC_EXTENDED. PG_PROC_INFO inlcui detalhes sobre os procedimentos armazenados e as funções, incluindo informações relacionadas aos argumentos de saída, se houver.

Colunas da tabela

PG_PROC_INFO mostra as seguintes colunas, além das colunas em PG_PROC. A coluna `oid` em PG_PROC é chamada de `prooid` na tabela PG_PROC_INFO.

Nome da coluna	Tipo de dados	Descrição
prooid	oid	O ID de objeto da função ou do procedimento armazenado.
prokind	“char”	Um valor que indica o tipo de funções ou procedimentos armazenados. Esse valor é “f” para funções regulares, “p” para procedimentos armazenados e “a” para funções agregadas.
proargmodes	“char”[ ]	Uma matriz com os modos dos argumentos do procedimento, codificada como “i” para argumentos IN, “o” para argumentos OUT e “b” para argumentos INOUT. Se todos os argumentos forem IN, esse campo será NULL. Subscritos correspondem a posições na matriz proallargtypes.
proallargtypes	oid[ ]	Uma matriz com os tipos de dados dos argumentos do procedimento. Essa matriz inclui todos os tipos de argumentos (incluindo argumentos OUT e INOUT). No entanto, se todos os argumentos forem IN, esse campo será NULL. A assinatura é baseada em um. Por outro lado, proargtypes é assinada de 0.

O campo proargnames em PG_PROC_INFO contém os nomes de todos os tipos de argumentos (incluindo OUT e INOUT), se houver.

PG_STATISTIC_INDICATOR

Armazena informações sobre o número de linhas inseridas ou excluídas desde o último comando ANALYZE. Como a tabela PG_STATISTIC_INDICATOR é atualizada frequentemente depois de operações DML, as estatísticas são aproximadas.

PG_STATISTIC_INDICATOR é visível apenas por superusuários. Para obter mais informações, consulte Visibilidade de dados em tabelas e visualizações de sistema.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
stairelid	oid	ID da tabela
stairows	flutuação	O número total de linhas na tabela.
staiins	flutuação	Número de linhas inseridas desde o último comando ANALYZE.
staidels	flutuação	Número de linhas excluídas ou atualizadas desde o último comando ANALYZE.

Exemplo

O exemplo a seguir retorna informações para alterações feitas na tabela desde o último comando ANALYZE.

```
select * from pg_statistic_indicator;

stairelid | stairows | staiins | staidels
----------+----------+---------+---------
   108271 |       11 |       0 |        0
   108275 |      365 |       0 |        0
   108278 |     8798 |       0 |        0
   108280 |    91865 |       0 |   100632
   108267 |    89981 |   49990 |     9999
   108269 |      808 |     606 |      374
   108282 |   152220 |   76110 |   248566
```

PG_TABLE_DEF

Armazena informações sobre colunas de tabela.

PG_TABLE_DEF retorna somente informações sobre tabelas visíveis para o usuário. Se PG_TABLE_DEF não retornar os resultados esperados, verifique se o parâmetro search_path está definido corretamente para incluir os esquemas relevantes.

Você pode usar SVV_TABLE_INFO para visualizar mais informações abrangentes sobre uma tabela, inclusive distorção dos dados, distorção da distribuição da chave, tamanho da tabela e estatísticas.

Colunas da tabela

Nome da coluna	Tipo de dados	Descrição
schemaname	name	Nome do esquema.
tablename	name	O nome da tabela.
column	name	Nome da coluna.
type	texto	Datatype da coluna.
encoding	character(32)	Codificação da coluna.
distkey	booliano	Verdadeiro se essa coluna for a chave de distribuição da tabela.
sortkey	integer	Ordem da coluna na chave de classificação. Se a tabela usar uma chave de classificação composta, todas as colunas que fizerem parte da chave de classificação terão um valor positivo que indicará a posição da coluna na chave de classificação. Se a tabela usar uma chave de classificação intercalada, cada coluna que fizer parte da chave de classificação terá um valor alternadamente positivo ou negativo, onde o valor absoluto indica a posição da coluna na chave de classificação. Se 0, a coluna não fará parte de uma chave de classificação.
notnull	booleano	Verdadeiro se a coluna tiver uma restrição NOT NULL.

Exemplo

O exemplo a seguir mostra as colunas da chave de classificação composta da tabela LINEORDER_COMPOUND.

```
select "column", type, encoding, distkey, sortkey, "notnull"
from pg_table_def
where tablename = 'lineorder_compound'
and sortkey <> 0;

column       | type    | encoding | distkey | sortkey | notnull
-------------+---------+----------+---------+---------+--------
lo_orderkey  | integer | delta32k | false   |       1 | true
lo_custkey   | integer | none     | false   |       2 | true
lo_partkey   | integer | none     | true    |       3 | true
lo_suppkey   | integer | delta32k | false   |       4 | true
lo_orderdate | integer | delta    | false   |       5 | true
(5 rows)
```

O exemplo a seguir mostra as colunas da chave de classificação intercalada da tabela LINEORDER_INTERLEAVED.

```
select "column", type, encoding, distkey, sortkey, "notnull"
from pg_table_def
where tablename = 'lineorder_interleaved'
and sortkey <> 0;

column       | type    | encoding | distkey | sortkey | notnull
-------------+---------+----------+---------+---------+--------
lo_orderkey  | integer | delta32k | false   |      -1 | true
lo_custkey   | integer | none     | false   |       2 | true
lo_partkey   | integer | none     | true    |      -3 | true
lo_suppkey   | integer | delta32k | false   |       4 | true
lo_orderdate | integer | delta    | false   |      -5 | true
(5 rows)
```

PG_TABLE_DEF retornará somente informações de tabelas nos esquemas incluídos no caminho de pesquisa. Para obter mais informações, consulte search_path.

Por exemplo, suponhamos que você crie um novo esquema e uma nova tabela e, em seguida, consulte PG_TABLE_DEF.

```
create schema demo;
create table demo.demotable (one int);
select * from pg_table_def where tablename = 'demotable';

schemaname|tablename|column| type | encoding | distkey | sortkey | notnull
----------+---------+------+------+----------+---------+---------+--------
```

A consulta não retorna linhas para a nova tabela. Examine a configuração de `search_path`.

```
show search_path;

  search_path
---------------
 $user, public
(1 row)
```

Adicione o esquema demo ao caminho de pesquisa e execute a consulta novamente.

```
set search_path to '$user', 'public', 'demo';

select * from pg_table_def where tablename = 'demotable';

schemaname| tablename |column|  type   | encoding |distkey|sortkey| notnull
----------+-----------+------+---------+----------+-------+-------+--------
demo      | demotable | one  | integer | none     | f     |     0 | f
(1 row)
```

PG_USER_INFO

PG_USER_INFO é uma visualização do sistema do Amazon Redshift que mostra informações do usuário, como ID do usuário e tempo de expiração da senha.

Somente superusuários podem ver PG_USER_INFO.

Colunas da tabela

PG_USER_INFO contém as colunas a seguir. Para obter mais informações, consulte a Documentação do PostgreSQL.

Nome da coluna	Tipo de dados	Descrição
usename	name	O nome de usuário.
usesysid	integer	O ID de usuário.
usecreatedb	booleano	Verdadeiro se o usuário puder criar bancos de dados.
usesuper	booleano	Verdadeiro se o usuário for um superusuário.
usecatupd	booleano	Verdadeiro se o usuário puder atualizar catálogos do sistema.
passwd	texto	A senha.
valuntil	abstime	A data e a hora de expiração da senha.
useconfig	text[]	A sessão usa como padrão as variáveis do tempo de execução.
useconnlimit	texto	O número de conexões que o usuário pode abrir.

Consultar as tabelas de catálogo

Tópicos

- Exemplos de consultas de catálogo

Em geral, você pode associar tabelas e visualizações de catálogo (relações cujos nomes começam com **PG_**) a tabelas e visualizações do Amazon Redshift.

As tabelas do catálogo usam vários tipos de dados não compatíveis com o Amazon Redshift. Os seguintes tipos de dados são compatíveis quando as consultas unem tabelas de catálogo a tabelas do Amazon Redshift:

- bool

- "char"
- float4
- int2
- int4
- int8
- name
- oid
- texto
- varchar

Se você gravar uma consulta de união que referencie explícita ou implicitamente uma coluna com um tipo de dados incompatível, a consulta retornará um erro. As funções SQL usadas em algumas das tabelas de catálogo também são incompatíveis, exceto as usadas pelas tabelas PG_SETTINGS e PG_LOCKS.

Por exemplo, a tabela PG_STATS não pode ser consultada em uma junção com uma tabela Amazon Redshift devido a funções não suportadas.

As tabelas e visualizações do catálogo a seguir fornecem informações úteis que podem ser associadas às informações nas tabelas do Amazon Redshift. Algumas dessas tabelas permitem somente acesso parcial por causa do tipo de dados e das restrições da função. Ao consultar as tabelas parcialmente acessíveis, selecione ou referencie as colunas com cuidado.

As seguintes tabelas são completamente acessíveis e não contêm tipos ou funções incompatíveis:

- pg_attribute
- pg_cast
- pg_depend
- pg_description
- pg_locks
- pg_opclass

As tabelas a seguir são parcialmente acessíveis e contêm alguns tipos incompatíveis, funções e colunas de texto truncadas. Os valores em colunas de texto são truncados em valores varchar(256).

- pg_class
- pg_constraint
- pg_database
- pg_group
- pg_language
- pg_namespace
- pg_operator
- pg_proc
- pg_settings
- pg_statistic
- pg_tables
- pg_type
- pg_user
- pg_views

As tabelas do catálogo que não estão listadas aqui são inacessíveis ou improváveis de serem úteis para administradores do Amazon Redshift. No entanto, você pode consultar qualquer tabela de catálogo ou visualizar abertamente se sua consulta não envolver uma junção a uma tabela do Amazon Redshift.

Você pode usar as colunas OID nas tabelas de catálogos Postgres como colunas de união. Por exemplo, a condição de união `pg_database.oid = stv_tbl_perm.db_id` compara o ID de objeto de banco de dados interno de cada linha PG_DATABASE com a coluna DB_ID visível na tabela STV_TBL_PERM. As colunas OID são chaves primárias internas que não são visíveis quando você seleciona uma na tabela. As exibições de catálogo não têm colunas OID.

Algumas funções do Amazon Redshift devem ser executadas apenas nos nós de computação. Se uma consulta fizer referência a uma tabela criada pelo usuário, o SQL executará em nós de computação.

Uma consulta que se refere apenas a tabelas do catálogo (tabelas com um prefixo PG, tal como PG_TABLE_DEF), ou que não se refere a qualquer tabela, é executada exclusivamente no nó de liderança.

Se uma consulta que usa uma função de nó de computação não fizer referência a uma tabela definida pelo usuário ou a uma tabela do sistema do Amazon Redshift, o erro a seguir será retornado.

```
[Amazon](500310) Invalid operation: One or more of the used functions must be applied
 on at least one user created table.
```

Exemplos de consultas de catálogo

As consultas a seguir mostram algumas das maneiras pelas quais você pode consultar as tabelas do catálogo para obter informações úteis sobre um banco de dados do Amazon Redshift.

ID da tabela de visualização, banco de dados, esquema e nome da tabela

A definição a seguir une a tabela de sistema STV_TBL_PERM às tabelas de catálogos de sistema PG_CLASS, PG_NAMESPACE e PG_DATABASE para retornar o ID da tabela, o nome do banco de dados, o nome do esquema e o nome da tabela.

```
create view tables_vw as
select distinct(stv_tbl_perm.id) table_id
,trim(pg_database.datname)   db_name
,trim(pg_namespace.nspname)   schema_name
,trim(pg_class.relname)   table_name
from stv_tbl_perm
join pg_class on pg_class.oid = stv_tbl_perm.id
join pg_namespace on pg_namespace.oid = pg_class.relnamespace
join pg_database on pg_database.oid = stv_tbl_perm.db_id;
```

O exemplo a seguir retorna as informações do ID da tabela 117855.

```
select * from tables_vw where table_id = 117855;
```

```
table_id | db_name   | schema_name | table_name
---------+-----------+-------------+-----------
  117855 |       dev | public      | customer
```

Listar o número de colunas por tabela do Amazon Redshift

A consulta a seguir une algumas tabelas de catálogo para descobrir quantas colunas cada tabela do Amazon Redshift contém. Os nomes de tabela do Amazon Redshift são armazenados em

PG_TABLES e STV_TBL_PERM; sempre que possível, use PG_TABLES para retornar nomes de tabela do Amazon Redshift.

Essa consulta não envolve tabelas do Amazon Redshift.

```
select nspname, relname, max(attnum) as num_cols
from pg_attribute a, pg_namespace n, pg_class c
where n.oid = c.relnamespace and  a.attrelid = c.oid
and c.relname not like '%pkey'
and n.nspname not like 'pg%'
and n.nspname not like 'information%'
group by 1, 2
order by 1, 2;

nspname | relname  | num_cols
--------+----------+----------
public  | category |        4
public  | date     |        8
public  | event    |        6
public  | listing  |        8
public  | sales    |       10
public  | users    |       18
public  | venue    |        5
(7 rows)
```

Listar os esquemas e as tabelas em um banco de dados

A consulta a seguir une STV_TBL_PERM a algumas tabelas PG para retornar uma lista de tabelas no banco de dados TICKIT e os nomes de esquema (coluna NSPNAME). A consulta também retorna o número total de linhas em cada tabela. (Essa consulta é útil quando vários esquemas no sistema têm os mesmos nomes de tabela.)

```
select datname, nspname, relname, sum(rows) as rows
from pg_class, pg_namespace, pg_database, stv_tbl_perm
where pg_namespace.oid = relnamespace
and pg_class.oid = stv_tbl_perm.id
and pg_database.oid = stv_tbl_perm.db_id
and datname ='tickit'
group by datname, nspname, relname
order by datname, nspname, relname;

datname | nspname | relname  |  rows
```

```
--------+---------+----------+--------
tickit  | public  | category |     11
tickit  | public  | date     |    365
tickit  | public  | event    |   8798
tickit  | public  | listing  | 192497
tickit  | public  | sales    | 172456
tickit  | public  | users    |  49990
tickit  | public  | venue    |    202
(7 rows)
```

IDs de tabela de listas, tipos de dados, nomes de colunas e nomes de tabelas

A consulta a seguir lista algumas informações sobre cada tabela de usuários e as colunas: ID da tabela, o nome da tabela, os nomes de coluna e o tipo de dados de cada coluna:

```
select distinct attrelid, rtrim(name), attname, typname
from pg_attribute a, pg_type t, stv_tbl_perm p
where t.oid=a.atttypid and a.attrelid=p.id
and a.attrelid between 100100 and 110000
and typname not in('oid','xid','tid','cid')
order by a.attrelid asc, typname, attname;

attrelid |  rtrim   |    attname     |  typname
---------+----------+----------------+-----------
  100133 | users    | likebroadway   | bool
  100133 | users    | likeclassical  | bool
  100133 | users    | likeconcerts   | bool
...
  100137 | venue    | venuestate     | bpchar
  100137 | venue    | venueid        | int2
  100137 | venue    | venueseats     | int4
  100137 | venue    | venuecity      | varchar
...
```

Contar o número de blocos de dados de cada coluna em uma tabela

A consulta a seguir une a tabela STV_BLOCKLIST a PG_CLASS para retornar informações de armazenamento das colunas na tabela SALES.

```
select col, count(*)
from stv_blocklist s, pg_class p
where s.tbl=p.oid and relname='sales'
```

```
group by col
order by col;

col | count
----+-------
  0 |     4
  1 |     4
  2 |     4
  3 |     4
  4 |     4
  5 |     4
  6 |     4
  7 |     4
  8 |     4
  9 |     8
 10 |     4
 12 |     4
 13 |     8
(13 rows)
```

Referência da configuração

Você pode usar arquivos de configuração para ajudar a personalizar seu ambiente. Com o Amazon Redshift, você pode personalizar e otimizar seu ambiente de data warehouse definindo vários parâmetros e configurações. A referência de configuração descreve as propriedades do cluster, os parâmetros do banco de dados e as opções de configuração do gerenciamento de workloads (WLM) disponíveis. Você pode consultar essa referência para ajustar o desempenho, a segurança e a alocação de recursos com base em seus requisitos específicos. A referência a seguir apresenta orientações detalhadas sobre como modificar essas configurações para conseguir a configuração de data warehouse desejada.

Tópicos

- Modificar a configuração do servidor
- analyze_threshold_percent
- cast_super_null_on_error
- datashare_break_glass_session_var
- datestyle
- default_array_search_null_handling
- default_geometry_encoding
- describe_field_name_in_uppercase
- downcase_delimited_identifier
- enable_case_sensitive_identifier
- enable_case_sensitive_super_attribute
- enable_numeric_rounding
- enable_result_cache_for_session
- enable_vacuum_boost
- error_on_nondeterministic_update
- extra_float_digits
- interval_forbid_composite_literals
- json_serialization_enable
- json_serialization_parse_nested_strings
- max_concurrency_scaling_clusters

- max_cursor_result_set_size
- max_failed_login_attempts
- mv_enable_aqmv_for_session
- navigate_super_null_on_error
- parse_super_null_on_error
- pg_federation_repeatable_read
- query_group
- search_path
- spectrum_enable_pseudo_columns
- enable_spectrum_oid
- spectrum_query_maxerror
- statement_timeout
- stored_proc_log_min_messages
- timezone
- use_fips_ssl
- wlm_query_slot_count

Modificar a configuração do servidor

Você pode alterar a configuração do servidor das seguintes formas:

- Usando um comando SET para substituir uma configuração somente pela duração da sessão atual.

  Por exemplo:

  ```
  set extra_float_digits to 2;
  ```

- Modificando as configurações do parameter group do cluster. Entre as configurações do parameter group estão os parâmetros adicionais que você pode configurar. Para obter mais informações, consulte “Grupos de parâmetros do Amazon Redshift” no Guia de gerenciamento de clusters do Amazon Redshift.
- Usando o comando ALTER USER a fim de definir um parâmetro de configuração para um novo valor padrão em todas as sessões executadas pelo usuário especificado.

```
ALTER USER username SET parameter { TO | = } { value | DEFAULT }
```

Use o comando SHOW para exibir as configurações de parâmetro atuais. Use SHOW ALL para exibir todas as configurações que você pode definir usando o comando SET.

```
SHOW ALL;

name                        | setting
----------------------------+--------------
analyze_threshold_percent | 10
datestyle                   | ISO, MDY
extra_float_digits          | 2
query_group                 | default
search_path                 | $user, public
statement_timeout           | 0
timezone                    | UTC
wlm_query_slot_count        | 1
```

> **Note**
>
> Os parâmetros de configuração são aplicados ao banco de dados ao qual você se conecta no data warehouse.

analyze_threshold_percent

Valores (padrão em negrito)

10, 0 a 100,0

Descrição

Define o limite para a porcentagem de linhas alteradas tendo em vista a análise de uma tabela. Para reduzir o tempo de processamento e melhorar a performance geral do sistema, o Amazon Redshift ignora a ANALYZE de qualquer tabela que tenha uma porcentagem menor de linhas alteradas do que o especificado por `analyze_threshold_percent`. Por exemplo, se uma tabela contiver 100.000.000 linhas e 9.000.000 delas tiverem sido alteradas desde o comando ANALYZE mais recente, por padrão, a tabela será ignorada porque menos de 10% delas mudaram. Para

analisar tabelas quando apenas uma quantidade pequena de linhas tiver sido alterada, defina `analyze_threshold_percent` como um número arbitrariamente pequeno. Por exemplo, se você definir `analyze_threshold_percent` como 0,01, a tabela com 100.000.000 linhas não será ignorada se pelo menos 10.000 linhas tiverem sido alteradas. Para analisar todas as tabelas, mesmo se nenhuma linha tiver sido alterada, defina `analyze_threshold_percent` como 0.

Você pode modificar o parâmetro `analyze_threshold_percent` da sessão atual usando apenas um comando SET. O parâmetro não pode ser modificado em um grupo de parâmetros.

Exemplo

```
set analyze_threshold_percent to 15;
set analyze_threshold_percent to 0.01;
set analyze_threshold_percent to 0;
```

cast_super_null_on_error

Valores (padrão em negrito)

on, off

Descrição

Especifica que quando você tenta acessar um membro inexistente de um objeto ou elemento de um array, o Amazon Redshift retornará um valor NULL se sua consulta for executada no modo lax padrão.

datashare_break_glass_session_var

Valores (padrão em negrito)

Não há padrão. O valor pode ser qualquer cadeia de caracteres gerada pelo Amazon Redshift quando ocorre uma operação que não é recomendada, conforme descrito a seguir.

Descrição

Aplica uma permissão que permite determinadas operações que geralmente não são recomendadas para uma unidade de compartilhamento de dados do AWS Data Exchange.

Em geral, recomendamos não descartar nem alterar uma unidade de compartilhamento de dados do AWS Data Exchange usando a instrução DROP DATASHARE ou ALTER DATASHARE SET PUBLICACCESSIBLE. Para permitir o descarte ou alteração de uma unidade de compartilhamento de dados do AWS Data Exchange para desativar a configuração publicamente acessível, defina a variável `datashare_break_glass_session_var` com um valor único. Esse valor único é gerado pelo Amazon Redshift e informado em uma mensagem de erro após a tentativa inicial na operação em questão.

Após definir a variável para o valor gerado uma vez, execute a instrução DROP DATASHARE ou ALTER DATASHARE novamente.

Para ter mais informações, consulte Observações de uso do ALTER DATASHARE ou Observações sobre o uso de DROP DATASHARE.

Exemplo

```
set datashare_break_glass_session_var to '620c871f890c49';
```

datestyle

Valores (padrão em negrito)

Especificação do formato (ISO, Postgres, SQL ou German) e ordem de ano/mês/dia (DMY, MDY, YMD).

- ISO: usa o estilo de data AAAA-MM-DD HH:MM:SS.
- Postgres: usa o estilo de data MM-DD HH:MM:SS AAAA.
- SQL: usa o estilo de data MM-DD-AAAA HH:MM:SS.
- Alemão: usa o estilo de data DD-MM-AAAA HH:MM:SS.

Descrição

Define o formato de exibição para valores de data e hora e também as regras para interpretar valores de entrada de data ambíguos. A string contém dois parâmetros que você pode alterar separadamente ou em conjunto.

Exemplo

```
show datestyle;
DateStyle
-----------
ISO, MDY
(1 row)

set datestyle to 'SQL,DMY';
```

default_array_search_null_handling

Valores (padrão em negrito)

TRUE, FALSE

Descrição

Especifica o comportamento de tratamento nulo para operações de pesquisa em matriz. Quando `default_array_search_null_handling` for `TRUE`, os valores NULL serão tratados como elementos válidos que podem ser pesquisados em matrizes. Quando `default_array_search_null_handling` for `FALSE`, as pesquisas de chave NULL exibirão NULL e, se a matriz contiver valores NULL sem nenhuma correspondência encontrada, a pesquisa exibirá NULL.

Exemplos

```
SET default_array_search_null_handling to TRUE;

-- ARRAY_CONTAINS: NULL search is allowed
SELECT ARRAY_CONTAINS(ARRAY('red', NULL, 'green'), NULL);
array_contains
----------------
 t
(1 row)

-- ARRAY_POSITION: Array can contain NULLs
SELECT ARRAY_POSITION(ARRAY('red', NULL, 'green'), 'blue');
 array_position
```

```
----------------
              -1
(1 row)
```

```
SET default_array_search_null_handling to FALSE;

-- ARRAY_CONTAINS: NULL search is disabled
SELECT ARRAY_CONTAINS(ARRAY('red', 'green'), NULL);
array_contains
----------------

(1 row)

-- ARRAY_POSITION: Array contains NULL but no match is found
SELECT ARRAY_POSITION(ARRAY('red', NULL, 'green'), 'blue');
 array_position
----------------

(1 row)
```

default_geometry_encoding

Valores (padrão em negrito)

1, 22

Descrição

Uma configuração de sessão que especifica se as geometrias espaciais criadas durante esta sessão são codificadas com uma caixa delimitadora. Se `default_geometry_encoding` é 1, as geometrias não são codificadas com uma caixa delimitadora. Se `default_geometry_encoding` é 2, então as geometrias são codificadas com uma caixa delimitadora. Para obter mais informações sobre o suporte para caixas delimitadoras, consulte Caixa delimitadora.

describe_field_name_in_uppercase

Valores (padrão em negrito)

off (false), on (true)

Descrição

Especifica se os nomes de coluna retornados pelas instruções SELECT estão em maiúsculas ou minúsculas. Se este parâmetro estiver ativado, os nomes das colunas serão retornados em maiúsculas. Se esse parâmetro for desativado, os nomes de coluna serão retornados em minúsculas. O Amazon Redshift armazena nomes de coluna em minúsculas, independentemente da configuração de `describe_field_name_in_uppercase`.

Exemplo

```
set describe_field_name_in_uppercase to on;

show describe_field_name_in_uppercase;

DESCRIBE_FIELD_NAME_IN_UPPERCASE
--------------------------------
on
```

downcase_delimited_identifier

Valores (padrão em negrito)

on, off

Descrição

Esta configuração está sendo retirada. Em vez disso, use `enable_case_sensitive_identifier`.

Permite que o superanalisador leia campos JSON que estão em letras maiúsculas ou mistas. Também habilita o suporte a consultas federadas para bancos de dados PostgreSQL compatíveis com nomes de maiúsculas e minúsculas de banco de dados, esquema, tabela e coluna. Para usar identificadores que diferenciam maiúsculas de minúsculas, defina esse parâmetro como desativado.

Observações sobre o uso

- Se você estiver usando recursos de mascaramento dinâmico de dados ou segurança por linha, recomendamos definir o valor de `downcase_delimited_identifier` no

grupo de parâmetros do cluster ou grupo de trabalho. Isso garante que `downcase_delimited_identifier` permaneça constante durante a criação e a anexação de uma política e a posterior consulta de uma relação que tenha uma política aplicada. Para obter informações sobre a segurança por linha, consulte Segurança por linha. Para obter informações sobre o mascaramento dinâmico de dados, consulte Mascaramento dinâmico de dados.

- Ao definir `downcase_delimited_identifier` como “off” e criar uma tabela, você pode definir nomes de coluna com distinção entre maiúsculas e minúsculas. Ao definir `downcase_delimited_identifier` como “on” e consultar a tabela, os nomes das colunas ficam em letras minúsculas. Isso pode produzir resultados de consulta diferentes de quando `downcase_delimited_identifier` está definido como “off”. Considere o seguinte exemplo:

```
SET downcase_delimited_identifier TO off;
--Amazon Redshift preserves case for column names and other identifiers.

--Create a table with two columns that are identical except for the case.
CREATE TABLE t ("c" int, "C" int);

INSERT INTO t VALUES (1, 2);

SELECT * FROM t;

 c | C
---+---
 1 | 2
(1 row)

SET enable_downcase_delimited_identifier TO on;
--Amazon Redshift no longer preserves case for column names and other identifiers.

SELECT * FROM t;

 c | c
---+---
 1 | 1
(1 row)
```

- Recomendamos que usuários regulares que consultam tabelas com políticas de segurança por linha ou de mascaramento dinâmico de dados anexadas tenham a configuração downcase_delimited_identifier padrão. Para obter mais informações sobre a segurança por linha, consulte Segurança por linha. Para obter informações sobre o mascaramento dinâmico de dados, consulte Mascaramento dinâmico de dados.

enable_case_sensitive_identifier

Valores (padrão em negrito)

true, false

Descrição

Um valor de configuração que determina se os identificadores de nome de bancos de dados, esquemas, tabelas e colunas diferenciam maiúsculas e minúsculas. As letras maiúsculas e minúsculas dos identificadores de nome são preservadas quando colocadas entre aspas duplas e o `enable_case_sensitive_identifier` é definido como `true`. As maiúsculas e minúsculas dos identificadores de nome não são preservadas e, em vez disso, são convertidas em minúsculas quando você não coloca os identificadores entre aspas duplas ou quando define `enable_case_sensitive_identifier` como `false`.

As maiúsculas e minnúsculas de um nome de usuário entre aspas duplas são sempre mantidas, independentemente da opção de configuração `enable_case_sensitive_identifier`.

Exemplos

O exemplo a seguir mostra como criar e usar identificadores que diferenciam maiúsculas e minúsculas para o nome da tabela e da coluna.

```
-- To create and use case sensitive identifiers
SET enable_case_sensitive_identifier TO true;

-- Create tables and columns with case sensitive identifiers
CREATE TABLE public."MixedCasedTable" ("MixedCasedColumn" int);

INSERT INTO public."MixedCasedTable" VALUES (1);
INSERT INTO public."MixedCasedTable" VALUES (2);
INSERT INTO public."MixedCasedTable" VALUES (3);
INSERT INTO public."MixedCasedTable" VALUES (4);
INSERT INTO public."MixedCasedTable" VALUES (5);

-- Now query with case sensitive identifiers
SELECT "MixedCasedColumn" FROM public."MixedCasedTable";

MixedCasedColumn
------------------
```

```
1
2
3
4
5

(5 rows)


SELECT * FROM public."MixedCasedTable" WHERE "MixedCasedColumn" = 1;

mixedcasedcolumn
------------------
1

(1 row)
```

O exemplo a seguir mostra quando a maiúscula e minúscula dos identificadores não são preservadas.

```
-- To not use case sensitive identifiers
RESET enable_case_sensitive_identifier;

-- Mixed case identifiers are lowercased despite double quotation marks

CREATE TABLE "MixedCasedTable2" ("MixedCasedColumn" int);

CREATE TABLE MixedCasedTable2 (MixedCasedColumn int);

ERROR:  Relation "mixedcasedtable2" already exists


SELECT "MixedCasedColumn" FROM "MixedCasedTable2";

 mixedcasedcolumn
------------------
(0 rows)


SELECT MixedCasedColumn FROM MixedCasedTable2;

 mixedcasedcolumn
------------------
```

```
(0 rows)
```

Observações sobre o uso

- Se você estiver usando a atualização automática para visões materializadas, recomendamos configurar o valor de `enable_case_sensitive_identifier` no grupo de parâmetros do cluster ou grupo de trabalho. Isso garante que `enable_case_sensitive_identifier` permaneça constante quando suas visões materializadas são atualizadas. Para obter informações sobre a atualização automática para visões materializadas, consulte Atualizar uma visualização materializada. Para obter informações sobre como definir valores de configuração em grupos de parâmetros, consulte Grupos de parâmetros do Amazon Redshift no Guia de gerenciamento do Amazon Redshift.
- Se você estiver usando recursos de mascaramento dinâmico de dados ou segurança por linha, recomendamos definir o valor de `enable_case_sensitive_identifier` no grupo de parâmetros do cluster ou grupo de trabalho. Isso garante que `enable_case_sensitive_identifier` permaneça constante durante a criação e a anexação de uma política e a posterior consulta de uma relação que tenha uma política aplicada. Para obter informações sobre a segurança por linha, consulte Segurança por linha. Para obter informações sobre o mascaramento dinâmico de dados, consulte Mascaramento dinâmico de dados.
- Ao definir `enable_case_sensitive_identifier` como “on” e criar uma tabela, você pode definir nomes de coluna com distinção entre maiúsculas e minúsculas. Ao definir `enable_case_sensitive_identifier` como “off” e consultar a tabela, os nomes das colunas ficam em letras minúsculas. Isso pode produzir resultados de consulta diferentes de quando `enable_case_sensitive_identifier` está definido como “on”. Considere o seguinte exemplo:

```
SET enable_case_sensitive_identifier TO on;
--Amazon Redshift preserves case for column names and other identifiers.

--Create a table with two columns that are identical except for the case.
CREATE TABLE t ("c" int, "C" int);

INSERT INTO t VALUES (1, 2);

SELECT * FROM t;

 c | C
```

```
---+---
 1 | 2
(1 row)

SET enable_case_sensitive_identifier TO off;
--Amazon Redshift no longer preserves case for column names and other identifiers.

SELECT * FROM t;

 c | c
---+---
 1 | 1
(1 row)
```

- Recomendamos que usuários regulares que consultam tabelas com políticas de segurança por linha ou de mascaramento dinâmico de dados anexadas tenham a configuração enable_case_sensitive_identifier padrão. Para obter informações sobre a segurança por linha, consulte Segurança por linha. Para obter informações sobre o mascaramento dinâmico de dados, consulte Mascaramento dinâmico de dados.
- Para fazer referência a identificadores com letras maiúsculas e minúsculas usando notação de ponto, coloque cada identificador que faz distinção de maiúsculas e minúsculas entre aspas duplas. Por exemplo, `public."MixedCasedTable"."MixedCasedColumn"`.

enable_case_sensitive_super_attribute

Valores (padrão em negrito)

true, false

Descrição

Um valor de configuração que determina se a navegação em estruturas do tipo de dado SUPER com nomes de atributo não delimitados diferencia letras maiúsculas de minúsculas. Quando você define `enable_case_sensitive_super_attribute` como `true`, a navegação em estruturas do tipo SUPER com nomes de atributos não delimitados não faz distinção entre letras maiúsculas e minúsculas. Quando você define o valor como `false`, a navegação em estruturas do tipo SUPER com nomes de atributos não delimitados não faz distinção entre letras maiúsculas e minúsculas.

Quando você coloca um nome de atributo entre aspas duplas e define `enable_case_sensitive_identifier` como `true`, a capitalização sempre é preservada, independentemente da opção definida para `enable_case_sensitive_super_attribute`.

`enable_case_sensitive_super_attribute` só se aplica a colunas com o tipo de dado SUPER. Para todas as outras colunas, use `enable_case_sensitive_identifier`.

Para obter mais informações sobre como navegar em campos JSON com maiúsculas e minúsculas, consulte Acessar campos JSON com nomes ou atributos de campo em maiúsculas ou mistas.

Exemplos

O exemplo a seguir mostra os resultados da seleção de valores SUPER com a opção `enable_case_sensitive_super_attribute` ativada e desativada.

```
--Create a table with a SUPER column.
CREATE TABLE tbl (col SUPER);

--Insert values.
INSERT INTO tbl VALUES (json_parse('{
 "A": "HELLO", "a": "123"
}'));

SET enable_case_sensitive_super_attribute TO ON;

SELECT col.A FROM tbl;
  a
-----
 "HELLO"
(1 row)

SELECT col.a FROM tbl;
  a
-----
 "123"
(1 row)

SET enable_case_sensitive_super_attribute TO OFF;

SELECT col.A FROM tbl;
  a
-----
```

```
 "123"
(1 row)

SELECT col.a FROM tbl;
  a
-----
 "123"
(1 row)
```

Observações sobre o uso

- As visões e visões materializadas seguem o valor de `enable_case_sensitive_super_attribute` no momento de sua criação. As visões de vinculação tardia, os procedimentos armazenados e as funções definidas pelo usuário seguem o valor de `enable_case_sensitive_super_attribute` no momento da consulta.
- Se você estiver usando a atualização automática para visões materializadas, recomendamos configurar `enable_case_sensitive_identifier value` no grupo de parâmetros do seu cluster ou grupo de trabalho. Isso garante que `enable_case_sensitive_identifier` permaneça constante quando suas visões materializadas são atualizadas. Para obter informações sobre a atualização automática para visões materializadas, consulte Atualizar uma visualização materializada. Para obter informações sobre como definir valores de configuração em grupos de parâmetros, consulte Grupos de parâmetros do Amazon Redshift no Guia de gerenciamento do Amazon Redshift.
- O nome da coluna nos resultados da instrução está sempre em letras minúsculas, independentemente do valor de `enable_case_sensitive_super_attribute`. Para que o nome da coluna também faça distinção entre maiúsculas e minúsculas, ative `enable_case_sensitive_identifier`.
- Recomendamos que os usuários regulares que consultam tabelas com políticas de segurança por linha anexadas tenham a configuração padrão `enable_case_sensitive_identifier`. Para obter mais informações sobre a segurança por linha, consulte Segurança por linha.

enable_numeric_rounding

Valores (padrão em negrito)

on (true), off (false)

Descrição

Especifica se o arredondamento numérico deve ser usado. Se `enable_numeric_rounding` for on, o Amazon Redshift arredondará valores NUMERIC ao convertê-los em outros tipos numéricos, como INTEGER ou DECIMAL. Se `enable_numeric_rounding` for `off`, o Amazon Redshift truncará valores NUMERIC ao convertê-los em outros tipos numéricos. Para obter mais informações sobre os tipos numéricos, consulte Tipos numéricos.

Exemplo

```
--Create a table and insert the numeric value 1.5 into it.
CREATE TABLE t (a numeric(10, 2));

INSERT INTO t VALUES (1.5);

SET enable_numeric_rounding to ON;
--Amazon Redshift now rounds NUMERIC values when casting to other numeric types.

SELECT a::int FROM t;

 a
---
 2
(1 row)


SELECT a::decimal(10, 0) FROM t;

 a
---
 2
(1 row)


 SELECT a::decimal(10, 5) FROM t;

    a
---------
 1.50000
(1 row)
```

```
SET enable_numeric_rounding to OFF;
--Amazon Redshift now truncates NUMERIC values when casting to other numeric types.

SELECT a::int FROM t;

 a
---
 1
(1 row)


SELECT a::decimal(10, 0) FROM t;

 a
---
 1
(1 row)


SELECT a::decimal(10, 5) FROM t;

    a
---------
 1.50000
(1 row)
```

enable_result_cache_for_session

Valores (padrão em negrito)

on (true), off (false)

Descrição

Especifica se o armazenamento em cache dos resultados da consulta será usado ou não. Se `enable_result_cache_for_session` for on, o Amazon Redshift verifica se há uma cópia em cache válida dos resultados da consulta quando uma consulta é enviada. Se for encontrada uma correspondência no cache de resultados, o Amazon Redshift usará os resultados armazenados no cache e não executará a consulta. Se `enable_result_cache_for_session` for `off`, o Amazon Redshift ignorará o cache de resultados e executará todas as consultas quando forem enviadas.

Exemplo

```
SET enable_result_cache_for_session TO off;
--Amazon Redshift now ignores the results cache
```

enable_vacuum_boost

Valores (padrão em negrito)

false, true

Descrição

Especifica se a opção vacuum boost deve ser habilitada para todos os comandos VACUUM executados em uma sessão. Se `enable_vacuum_boost` for `true`, o Amazon Redshift executa todos os comandos VACUUM na sessão com a opção BOOST. Se `enable_vacuum_boost` for`false`, o Amazon Redshift não é executado com a opção BOOST por padrão. Para obter mais informações sobre a opção BOOST, consulte VACUUM.

error_on_nondeterministic_update

Valores (padrão em negrito)

false, true

Descrição

Especifica se as consultas UPDATE com várias correspondências por linha geram um erro.

Exemplo

```
SET error_on_nondeterministic_update TO true;

CREATE TABLE t1(x1 int, y1 int);

CREATE TABLE t2(x2 int, y2 int);

INSERT INTO t1 VALUES (1,10), (2,20), (3,30);
```

```
INSERT INTO t2 VALUES (2,40), (2,50);

UPDATE t1 SET y1=y2 FROM t2 WHERE x1=x2;

ERROR: Found multiple matches to update the same tuple.
```

extra_float_digits

Valores (padrão em negrito)

0, -15 a 2

Descrição

Define o número de dígitos exibidos para valores de ponto flutuante, inclusive float4 e float8. O valor é adicionado ao número padrão de dígitos (FLT_DIG ou DBL_DIG, conforme apropriado). O valor pode ser definido em até 2, para incluir dígitos parcialmente significativos. Isso é especialmente útil para gerar dados flutuantes que devem ser restaurados com exatidão. Ou ele pode ser negativo a fim de suprimir dígitos indesejados.

Exemplo

O exemplo a seguir define `extra_float_digits` como `-2`. Primeiro, mostre a configuração atual do parâmetro.

```
show all;
      name                      | setting
--------------------------+----------------
analyze_threshold_percent | 10
datestyle                 | ISO, MDY
extra_float_digits        | 2
query_group               | default
search_path               | $user, public
statement_timeout         | 0
timezone                  | UTC
wlm_query_slot_count      | 1
```

Depois, defina o novo valor como `-2`.

```
set extra_float_digits to -2;
```

Por fim, mostre a configuração atualizada do parâmetro.

```
show all;
      name                       | setting
---------------------------+----------------
analyze_threshold_percent | 10
datestyle                 | ISO, MDY
extra_float_digits        | -2
query_group               | default
search_path               | $user, public
statement_timeout         | 0
timezone                  | UTC
wlm_query_slot_count      | 1
```

interval_forbid_composite_literals

Valores (padrão em negrito)

false, true

Descrição

Uma configuração de sessão que modifica o valor de um intervalo que contém as partes YEAR TO MONTH e DAY TO SECOND.

Caso `interval_forbid_composite_literals` seja `true`, será retornado um erro se for encontrado um intervalo com as partes YEAR TO MONTH e DAY TO SECOND. Por exemplo, o SQL a seguir contém INTERVAL DAY TO SECOND com as partes YEAR TO MONTH e DAY TO SECOND.

```
SELECT INTERVAL '1 year 1 day' DAY TO SECOND;
ERROR:  Interval Day To Second literal cannot contain year-month parts. Disable the GUC
 interval_forbid_composite_literals to suppress this error and silently discard the
 year-month part.
```

Caso `interval_forbid_composite_literals` seja `false`, o Amazon Redshift suprimirá um erro e truncará a parte YEAR TO MONTH de um valor de INTERVAL DAY TO SECOND. Por exemplo, o SQL a seguir contém INTERVAL DAY TO SECOND com as partes YEAR TO MONTH e DAY TO SECOND.

```
SET interval_forbid_composite_literals to "false";
SELECT INTERVAL '1 year 1 day' DAY TO SECOND;


 intervald2s
-----------------------------
1 days 0 hours 0 mins 0.0 secs
```

json_serialization_enable

Valores (padrão em negrito)

false, true

Descrição

Uma configuração de sessão que modifica o comportamento de serialização JSON dos dados formatados ORC, JSON, Ion e Parquet. Se `json_serialization_enable` for `true`, todas as coleções de nível superior são serializadas automaticamente para JSON e retornadas como VARCHAR(65535). Colunas não complexas não são afetadas ou serializadas. Como as colunas de coleção são serializadas como VARCHAR (65535), seus subcampos aninhados não podem mais ser acessados diretamente como parte da sintaxe de consulta (isto é, na cláusula de filtro). Se `json_serialization_enable` for `false`, coleções de nível superior não são serializadas para JSON. Para obter mais informações sobre serialização JSON aninhada, consulte Serializar JSON aninhado complexo.

json_serialization_parse_nested_strings

Valores (padrão em negrito)

false, true

Descrição

Uma configuração de sessão que modifica o comportamento de serialização JSON dos dados formatados ORC, JSON, Ion e Parquet. Quando ambos `json_serialization_parse_nested_strings` e `json_serialization_enable` tiverem o valor true, os valores da cadeia de caracteres que são armazenados em tipos complexos (como

mapas, estruturas ou vetores) serão analisados e gravados em linha diretamente no resultado se forem JSON válido. Se `json_serialization_parse_nested_strings` for false, strings dentro de tipos complexos aninhados são serializadas como strings JSON escapadas. Para obter mais informações, consulte Serializar tipos complexos contendo strings JSON.

max_concurrency_scaling_clusters

Valores (padrão em negrito)

1, 0 a 10

Descrição

Define o número máximo de clusters de escalabilidade da simultaneidade permitidos quando a escalabilidade da simultaneidade estiver habilitada. Aumente esse valor se mais escalabilidade da simultaneidade for necessária. Diminua esse valor para reduzir o uso de clusters de escalabilidade da simultaneidade e as cobranças resultantes.

O número máximo de clusters de escalabilidade da simultaneidade é uma cota ajustável. Para obter mais informações, consulte "Cotas do Amazon Redshift" no Guia de gerenciamento de clusters do Amazon Redshift.

max_cursor_result_set_size

Valores (padrão em negrito)

0 (assume o máximo como padrão) – 14.400.000 MB

Descrição

O parâmetro `max_cursor_result_set_size` não é mais usado. Para obter mais informações sobre o tamanho do conjunto de resultados do cursor, consulte Restrições de cursor.

max_failed_login_attempts

Valores (padrão em negrito)

5, 2 a 50

Descrição

Define o número de tentativas consecutivas de login com senha com falha que o Amazon Redshift permite antes de bloquear um usuário do banco de dados. Quando o número de tentativas consecutivas com falha atinge esse valor, o Amazon Redshift bloqueia o usuário. Um usuário bloqueado permanece bloqueado até que um superusuário ou um usuário com o privilégio ALTER USER o desbloqueie. Um login bem-sucedido zera o contador. Esse parâmetro se aplica somente à autenticação baseada em senha. Os usuários federados não são afetados.

Esse parâmetro é de âmbito do cluster. Para configurá-lo, você deve ser um superusuário e usar o comando ALTER SYSTEM SET. Você não pode alterá-lo com o comando SET. A alteração se aplica sem a reinicialização do cluster. O intervalo válido é de 2 a 50; um valor fora desse intervalo retorna um erro.

Exemplo

```
ALTER SYSTEM SET max_failed_login_attempts = 10;
```

mv_enable_aqmv_for_session

Valores (padrão em negrito)

true, false

Descrição

Especifica se o Amazon Redshift pode executar a reescrita automática de consultas de visões materializadas no nível da sessão.

navigate_super_null_on_error

Valores (padrão em negrito)

on, off

Descrição

Especifica que quando você tenta navegar por um membro inexistente de um objeto ou elemento de um array, o Amazon Redshift retornará um valor NULL se sua consulta for executada no modo lax padrão.

parse_super_null_on_error

Valores (padrão em negrito)

off, on

Descrição

Especifica que quando o Amazon Redshift tenta analisar um membro inexistente de um objeto ou elemento de um array, o Amazon Redshift retornará um valor NULL se sua consulta for executada no modo estrito.

pg_federation_repeatable_read

Valores (padrão em negrito)

true, false

Descrição

Especifica o nível de isolamento de transação de consulta federada para os resultados do banco de dados PostgreSQL.

- Quando `pg_federation_repeatable_read` é true, as transações federadas são processadas com a semântica de nível de isolamento REPEATABLE READ. Esse é o padrão.
- Quando `pg_federation_repeatable_read` é false, as transações federadas são processadas com a semântica de nível de isolamento READ COMMITTED.

Para saber mais, consulte:

- Considerações ao acessar dados federados com o Amazon Redshift.
- Gerenciamento de operações de gravação simultâneas.

Exemplos

O comando a seguir define `pg_federation_repeatable_read` como on para uma sessão. O comando show mostra o valor de set.

```
set pg_federation_repeatable_read to on;
```

```
show pg_federation_repeatable_read;

pg_federation_repeatable_read
-----------------------------
on
```

query_group

Valores (padrão em negrito)

Sem padrão; o valor pode ser qualquer string de caracteres.

Descrição

Aplica um rótulo definido pelo usuário a um grupo de consultas que são executadas durante a mesma sessão. Esse rótulo é capturado nos logs de consulta. Você pode usá-lo para restringir os resultados das tabelas STL_QUERY e STV_INFLIGHT e da visualização SVL_QLOG. Por exemplo, você pode aplicar um rótulo à parte a cada consulta executada para identificar com exclusividade consultas sem precisar pesquisar os IDs.

Este parâmetro não existe no arquivo de configuração do servidor e deve ser definido em tempo de execução com um comando SET. Embora você possa usar uma string de caracteres longa como um rótulo, o rótulo é truncado em 30 caracteres na coluna LABEL da tabela STL_QUERY e na exibição SVL_QLOG (e até 15 caracteres em STV_INFLIGHT).

No exemplo a seguir, query_group está definido como **`Monday`**, logo, várias consultas são executadas com esse rótulo.

```
set query_group to 'Monday';
SET
select * from category limit 1;
```

```
...
...
select query, pid, substring, elapsed, label
from svl_qlog where label ='Monday'
order by query;

query | pid  |        substring                   |  elapsed  | label
------+------+------------------------------------+-----------+--------
789   | 6084 | select * from category limit 1;    | 65468     | Monday
790   | 6084 | select query, trim(label) from ... | 1260327   | Monday
791   | 6084 | select * from svl_qlog where ..    | 2293547   | Monday
792   | 6084 | select count(*) from bigsales;     | 108235617 | Monday
...
```

search_path

Valores (padrão em negrito)

'$user', public, schema_names

Uma lista separada por vírgulas de nomes de esquemas existentes. Se '$user' estiver presente, o esquema com o mesmo nome de `SESSION_USER` será substituído, ou será ignorado.

Descrição

Especifica a ordem em que os esquemas são pesquisados quando um objeto (como uma tabela ou função) é referenciado por um nome simples sem nenhum componente de esquema:

- Os caminhos de pesquisa não são compatíveis com esquemas e tabelas externas. As tabelas externas devem ser qualificadas explicitamente por um esquema externo. As unidades de compartilhamento de dados se comportam como dados externos e, portanto, não são compatíveis como caminhos de pesquisa.
- Quando são criados sem um esquema de destino específico, os objetos são colocados no primeiro esquema listado no caminho de pesquisa. Se o caminho de pesquisa estiver vazio, o sistema retornará um erro.
- Quando houver objetos com nomes idênticos em esquemas diferentes, aquele encontrado no caminho de pesquisa será usado.
- Um objeto que não está em nenhum dos esquemas no caminho de pesquisa só pode ser referenciado especificando-se o esquema que o contém com um nome qualificado (com pontos).

- O esquema de catálogo do sistema, pg_catalog, sempre é pesquisado. Se estiver mencionado no caminho, ele será pesquisado na ordem especificada. Do contrário, ele será pesquisado antes de qualquer item de caminho.
- Se existir, o esquema de tabela temporária da sessão atual, pg_temp_nnn, sempre será pesquisado. Ele pode ser listado explicitamente no caminho usando-se o alias pg_temp. Se não estiver listado no caminho, ele será pesquisado primeiro (até mesmo antes de pg_catalog). Porém, o esquema temporário somente é pesquisado em busca de nomes de relação (tabelas, exibições). Ele não é pesquisado em busca de nomes de função.

Exemplo

O exemplo a seguir cria o esquema ENTERPRISE e define search_path como o novo esquema.

```
create schema enterprise;
set search_path to enterprise;
show search_path;

 search_path
-------------
 enterprise
(1 row)
```

O exemplo a seguir adiciona o esquema ENTERPRISE ao search_path padrão.

```
set search_path to '$user', public, enterprise;
show search_path;

         search_path
-----------------------------
 "$user", public, enterprise
(1 row)
```

O exemplo a seguir adiciona a tabela FRONTIER ao esquema ENTERPRISE.

```
create table enterprise.frontier (c1 int);
```

Quando a tabela PUBLIC.FRONTIER é criada no mesmo banco de dados e o usuário não especifica o nome do esquema em uma consulta, PUBLIC.FRONTIER tem precedência sobre ENTERPRISE.FRONTIER.

```
create table public.frontier(c1 int);
insert into enterprise.frontier values(1);
select * from frontier;

frontier
----
(0 rows)

select * from enterprise.frontier;

c1
----
1
(1 row)
```

spectrum_enable_pseudo_columns

Valores (padrão em negrito)

true, false

Descrição

Você pode desabilitar a criação de pseudocolunas em uma sessão. Basta definir o parâmetro de configuração `spectrum_enable_pseudo_columns` como `false`.

Exemplo

O comando a seguir desabilita a criação de pseudocolunas em uma sessão.

```
set spectrum_enable_pseudo_columns to false;
```

enable_spectrum_oid

Valores (padrão em negrito)

true, false

Descrição

Você também pode desabilitar somente a pseudocoluna `$spectrum_oid` definindo o parâmetro de configuração `enable_spectrum_oid` como `false`.

Exemplo

O comando a seguir desabilita a pseudocoluna `$spectrum_oid` definindo o parâmetro de configuração `enable_spectrum_oid` como `false`.

```
set enable_spectrum_oid to false;
```

spectrum_query_maxerror

Valores (padrão em negrito)

-1, inteiro

Descrição

Você pode especificar um inteiro para indicar o número máximo de erros que devem ser aceitos antes que a consulta seja cancelada. Um valor negativo desativa o tratamento de máximo de dados de erro. Os resultados estão registrados em SVL_SPECTRUM_SCAN_ERROR.

Exemplo

O exemplo a seguir pressupõe dados ORC que contêm caracteres excedentes e caracteres que não são válidos. A definição de coluna para `my_string` especifica um comprimento de 3 caracteres. A seguir, um exemplo de dados para esse exemplo:

```
my_string
---------------
abcdef
gh
ab
```

Os comandos a seguir definem o número máximo de erros como 1 e executam a consulta.

```
set spectrum_query_maxerror to 1;
```

```
SELECT my_string FROM orc_data;
```

A consulta é interrompida e os resultados são registrados em SVL_SPECTRUM_SCAN_ERROR.

statement_timeout

Valores (padrão em negrito)

0 (desativa a limitação), x milissegundos

Descrição

Interrompe qualquer instrução que ultrapasse o número especificado de milissegundos.

O valor `statement_timeout` é a quantidade máxima de tempo que uma consulta pode ser executada antes que o Amazon Redshift a encerre. Esse tempo inclui planejamento, enfileiramento no gerenciamento de workload (WLM) e tempo de execução. Compare esse tempo como o tempo limite do WLM (max_execution_time) e de uma QMR (query_execution_time), que inclui somente o tempo de execução.

Se o tempo limite do WLM (max_execution_time) também for especificado como parte de uma configuração do WLM, o mais baixo de statement_timeout e de max_execution_time será usado. Para obter mais informações, consulte Tempo limite do WLM.

Exemplo

Como a consulta a seguir leva mais de 1 milissegundo, ela atinge o tempo limite e é cancelada.

```
set statement_timeout = 1;

select * from listing where listid>5000;
ERROR:  Query (150) canceled on user's request
```

stored_proc_log_min_messages

Valores (padrão em negrito)

LOG, INFO, NOTICE, WARNING, EXCEPTION

Descrição

Especifica o nível mínimo de registro em log das mensagens de procedimento armazenado geradas. As mensagens no nível especificado ou acima dele são registradas. O padrão é LOG (todas as mensagens são registradas em log). A ordem dos níveis de log do mais alto para o mais baixo é a seguinte:

1. EXCEPTION
2. WARNING
3. NOTICE
4. INFORMAÇÕES
5. LOG

Por exemplo, se você especificar um valor de NOTICE, as mensagens serão registradas em log somente para NOTICE, WARNING e EXCEPTION.

timezone

Valores (padrão em negrito)

UTC, fuso horário

Sintaxe

```
SET timezone { TO | = } [ time_zone | DEFAULT ]

SET time zone [ time_zone | DEFAULT ]
```

Descrição

Define o fuso horário da sessão atual. O fuso horário pode ser a diferença do Tempo Universal Coordenado (UTC) ou um nome de fuso horário.

> **Note**
>
> Você não pode definir o parâmetro de configuração `timezone` usando um parameter group de cluster. O fuso horário pode ser definido somente para a sessão atual usando um

comando SET. Para definir o fuso horário de para todas as sessões executadas por um usuário de banco de dados específico, use o comando ALTER USER. ALTER USER … SET TIMEZONE altera o fuso horário de sessões subsequentes, e não da sessão atual.

Ao definir o fuso horário usando o comando `SET timezone` (uma palavra) com qualquer `TO` ou `=`, você pode especificar time_zone, como um nome de fuso horário, uma diferença em formato POSIX ou ISO-8601, conforme mostrado a seguir.

```
SET timezone { TO | = } time_zone
```

Ao definir o fuso horário usando o comando de fuso horário SET sem `TO` ou `=`, você pode especificar time_zone usando um INTERVAL, bem como um nome de fuso horário, um deslocamento de formato no estilo POSIX ou um deslocamento de formato ISO-8601, conforme mostrado a seguir.

```
SET time zone time_zone
```

Formatos de fuso horário

O Amazon Redshift oferece suporte aos seguintes formatos de fuso horário:

- Nome do fuso horário
- INTERVAL
- especificação de fuso horário em estilo POSIX
- Diferença ISO-8601

Como as abreviações de fuso horário, como PST ou PDT, são definidas como uma diferença fixa em relação ao UTC e não incluem regras de horário de verão, o comando SET não dá suporte a abreviações de fuso horário.

Para obter mais detalhes sobre formatos de fuso horário, consulte as informações a seguir.

Nome do fuso horário – o nome completo do fuso horário, como America/New_York. Entre os nomes de fuso horário completos estão regras de horário de verão. Observe que o Amazon Redshift usa o banco de dados de fuso horário IANA como fonte autorizada de especificação de fuso horário.

Estes são exemplos de nomes de fuso horário:

- Etc/Greenwich
- America/New_York
- CST6CDT
- GB

> **Note**
>
> Muitos nomes de fuso horário, como EST, MST, NZ e UCT, também são abreviações.

Para exibir uma lista de nomes de fuso horário válidos, execute o comando a seguir.

```
select pg_timezone_names();
```

INTERVAL – Um deslocamento do UTC. Por exemplo, PST é -8:00 ou -8 horas.

Estes são exemplos de diferenças de fuso horário INTERVAL:

- -8:00
- -8 horas
- 30 minutos

Formato de estilo POSIX – Uma especificação de fuso horário no formato STDoffset ou STDoffsetDST, em que STD é uma abreviatura de fuso horário, deslocamento é o deslocamento numérico em horas a oeste do UTC e DST é uma abreviação opcional do fuso horário de verão. Supõe-se que o horário de verão seja uma hora a mais que o deslocamento fornecido.

Os formatos de fuso horário estilo POSIX usam deslocamentos positivos à Oeste de Greenwich, em contraste com a convenção ISO-8601, usando deslocamentos positivos à Leste de Greenwich.

A seguir, exemplos de fusos horários no estilo POSIX:

- PST8
- PST8PDT
- EST5
- EST5EDT

> **Note**
>
> O Amazon Redshift não valida especificações de fuso horário no estilo POSIX, portanto é possível definir o fuso horário com um valor inválido. Por exemplo, o seguinte comando não retorna um erro, mesmo que ele defina o fuso horário com um valor inválido.
>
> ```
> set timezone to ‘xxx36’;
> ```

Deslocamento ISO-8601 – O deslocamento do UTC no formulário `±[hh]:[mm]`.

Estes são exemplos de diferenças ISO-8601:

- -8:00
- +7:30

Exemplos

O exemplo a seguir define o fuso horário da sessão atual como Nova York.

```
set timezone = 'America/New_York';
```

O exemplo a seguir define o fuso horário da sessão atual como UTC–8 (PST).

```
set timezone to '-8:00';
```

O exemplo a seguir usa INTERVAL para definir o fuso horário como PST.

```
set timezone interval '-8 hours'
```

O exemplo a seguir redefine o fuso horário da sessão atual como o fuso horário padrão do sistema (UTC).

```
set timezone to default;
```

Para definir o fuso horário do usuário do banco de dados, use uma instrução ALTER USER … SET. O exemplo a seguir define o fuso horário de dbuser como Nova York. O novo valor persiste para o usuário em todas as sessões subsequentes.

```
ALTER USER dbuser SET timezone to 'America/New_York';
```

use_fips_ssl

Valores (padrão em negrito)

true, false

Descrição

Um valor de grupo de parâmetros que especifica se o modo SSL compatível com FIPS é utilizado. Se `use_fips_ssl` for `true`, o modo SSL compatível com FIPS será utilizado. Se `use_fips_ssl` for `false`, o modo SSL compatível com FIPS não será utilizado. Para ter mais informações, consulte Configurar as opções de segurança para conexões no Guia de gerenciamento do Amazon Redshift.

Para configurar parâmetros para um cluster provisionado do Amazon Redshift, consulte Sobre grupos de parâmetros no Guia de gerenciamento do Amazon Redshift. Para configurar parâmetros para o Redshift sem servidor, consulte Configuring a FIPS-compliant SSL connection to Amazon Redshift Serverless no Guia de gerenciamento do Amazon Redshift e CreateWorkgroup ou UpdateWorkgroup na Referência de API do Redshift sem servidor.

wlm_query_slot_count

Valores (padrão em negrito)

1, 1 de 1 a 50 (não pode exceder o número de slots disponíveis [o nível de simultaneidade] da classe de serviço)

Descrição

Define o número de slots que uma consulta usa.

O gerenciamento de workload (WLM) reserva slots em uma classe de serviço de acordo com o nível de simultaneidade definido para a fila. Por exemplo, se o nível de simultaneidade for definido como 5, a classe de serviço terá 5 slots. O WLM aloca igualmente a memória disponível para uma classe de serviço para cada slot. Para obter mais informações, consulte Gerenciamento do workload.

> **Note**
>
> Se o valor de wlm_query_slot_count for maior que o número de slots disponíveis (nível de simultaneidade) da classe de serviço, haverá falha na consulta. Se você encontrar um erro, diminua wlm_query_slot_count para um valor permissível.

Para operações em que a performance seja muito afetada pelo valor de memória alocado, como vacuum, o aumento do valor de wlm_query_slot_count pode melhorar a performance. Em especial, para comandos vacuum lentos, inspecione o registro correspondente na exibição SVV_VACUUM_SUMMARY. Se você vir valores elevados (próximos de ou maiores que 100) para sort_partitions e merge_increments na exibição SVV_VACUUM_SUMMARY, leve em consideração aumentar o valor para wlm_query_slot_count na próxima vez em que executar o Vacuum em relação a essa tabela.

Aumentar o valor de wlm_query_slot_count limita o número de consultas simultâneas que podem ser executadas. Por exemplo, suponhamos que a classe de serviço tem um nível de simultaneidade 5 e wlm_query_slot_count esteja definido como 3. Enquanto uma consulta está em execução na sessão, com wlm_query_slot_count definido como 3, no máximo 2 consultas simultâneas podem ser executadas dentro da mesma classe de serviço. As consultas subsequentes aguardarão na fila até a execução de consultas em andamento ser concluída e os slots serem liberados.

Exemplos

Use o comando SET para definir o valor de wlm_query_slot_count para a duração da sessão atual.

```
set wlm_query_slot_count to 3;
```

Histórico do documento

> **Note**
>
> Para ver uma descrição dos novos recursos no Amazon Redshift, consulte What's new.

A tabela a seguir descreve as alterações importantes feitas no Guia do desenvolvedor de banco de dados do Amazon Redshift após maio de 2018. Para receber notificações sobre atualizações dessa documentação, é possível inscrever-se em um feed RSS.

Versão da API: 2012-12-01

Para obter uma lista das alterações feitas no Guia de gerenciamento de clusters do Amazon Redshift, consulte "Histórico do documento" do Guia de gerenciamento de clusters do Amazon Redshift.

Para obter mais informações sobre os novos recursos, incluindo uma lista de correções e os números de versão do cluster associados a cada versão, consulte Histórico das versões de cluster.

Alteração	Descrição	Data
Suporte à Integração ETL zero do Amazon DynamoDB com o Amazon Redshift	Agora é possível criar Integrações ETL zero entre o DynamoDB e o Amazon Redshift para habilitar analytics em tempo real em dados do DynamoDB sem processos tradicionais de ETL.	1.º de dezembro de 2024
Suporte a Integrações ETL zero do Amazon Aurora com a filtragem de dados do Amazon Redshift	Agora é possível aplicar filtros de dados a Integrações ETL zero do Amazon Aurora para replicar dados específicos seletivamente para o Amazon Redshift.	1.º de junho de 2024

Suporte à implantação multi-AZ para o Amazon Redshift	O Amazon Redshift agora permite implantações multi-AZ para clusters RA3, fornecend o alta disponibilidade por meio da implantação de recursos de computação em duas zonas de disponibilidade para continuar operando durante eventos inesperados em uma AZ.	1º de março de 2024
Suporte a Integrações ETL zero do Amazon Aurora com o Amazon Redshift	Agora é possível criar Integrações ETL zero entre o Aurora e o Amazon Redshift para permitir analytics e machine learning quase em tempo real usando o Amazon Redshift em petabytes de dados transacionais do Amazon Aurora.	1º de novembro de 2023
Suporte a Integrações ETL zero do Amazon RDS com o Amazon Redshift	Agora é possível criar Integrações ETL zero entre o Amazon RDS e o Amazon Redshift para disponibilizar dados do banco de dados do RDS no Amazon Redshift quase em tempo real para workloads de analytics, ML e IA.	1º de junho de 2023

Suporte à ingestão de streaming entre regiões	A ingestão de streaming do Amazon Redshift agora permite a ingestão de streaming entre regiões, caso em que o Amazon Kinesis Data Streams de origem ou um tópico do Amazon MSK pode estar localizado em uma Região da AWS diferente da região do cluster do Amazon Redshift.	1º de novembro de 2022
Suporte à ingestão de streaming de origens do Apache Kafka	Agora é possível ingerir dados de streaming do Amazon MSK provisionado, do Amazon MSK Sem Servidor e de clusters do Apache Kafka de código aberto diretamente no Amazon Redshift usando visões materializadas.	1° de julho de 2022
Suporte ao Amazon Redshift sem servidor	Com o Amazon Redshift sem servidor, você tem a comodidade de executar e escalar analytics sem precisar provisionar e gerenciar data warehouses. Foi adicionad a documentação abrangent e para grupos de trabalho sem servidor, namespace s, monitoramento de uso e visualizações sem servidor específicas do sistema.	29 de junho de 2022

Suporte à ingestão de streaming do Amazon Kinesis Data Streams	Agora é possível ingerir dados do Amazon Kinesis Data Streams diretamente no Amazon Redshift usando visões materializadas, o que elimina a necessidade de preparar os dados no Amazon S3 antes de ingeri-los no Amazon Redshift.	1.º de abril de 2022
Suporte a dados espaciais com tipos de dados GEOMETRY e GEOGRAPHY	O Amazon Redshift agora permite dados espaciais com tipos de dados GEOMETRY e GEOGRAPHY, funções SQL espaciais e o carregamento de shapefiles para análise geoespacial e consultas baseadas em localização.	1º de março de 2022
Suporte ao AQUA (Advanced Query Accelerator)	O AQUA é um cache distribuí do e acelerado por hardware que permite que o Amazon Redshift seja executado até dez vezes mais depressa do que outros data warehouse s na nuvem, impulsionando automaticamente determina dos tipos de consulta com hardware personalizado.	1.º de janeiro de 2022

Recursos aprimorados do Amazon Redshift ML	Foi adicionada documentação abrangente para o Amazon Redshift ML, que inclui a instrução CREATE MODEL, suporte à XGBoost, agrupamento K-means, modelos de classificação multiclasse, recursos de explicabilidade de modelo e integração com o Amazon SageMaker AI para oferecer ML automatizado.	1º de dezembro de 2021
Suporte aprimorado a dados semiestruturados com o tipo de dados SUPER	Recursos aprimorados para o tipo de dados SUPER para armazenar e consultar dados semiestruturados, como JSON, Avro e Ion, com desempenho aprimorado, digitação dinâmica e suporte para carregar até 16 MB de dados semiestruturados.	1º de novembro de 2023
Suporte a recursos aprimorados de compartilhamento de dados	Funcionalidade aprimorada de compartilhamento de dados com suporte para compartilhamento de dados entre clusters e entre contas, integração com o Amazon Redshift sem servidor e unidades de compartilhamento de dados padrão entre diferentes tipos de cluster.	1.º de outubro de 2021

Suporte para geometrias 3D e 4D espaciais e novas funções espaciais	Agora é possível usar funções espaciais adicionais, e o suporte a geometria 3D e 4D foi adicionado a algumas funções.	19 de agosto de 2021
Suporte para codificação de compressão de coluna para otimização automática de tabelas	Você pode especificar a opção ENCODE AUTO para a tabela gerenciar automaticamente a codificação de compactação para todas as colunas da tabela.	3 de agosto de 2021
Suporte para várias ou para uma instrução SQL com parâmetros usando a API Data do Amazon Redshift	Agora é possível executar várias ou uma instrução SQL com parâmetros usando a API Data do Amazon Redshift.	28 de julho de 2021
Suporte para agrupamento sem distinção entre maiúsculas e minúsculas com substituições de nível	Agora é possível usar a cláusula COLLATE em uma instrução CREATE DATABASE para especificar o agrupamento padrão.	24 de junho de 2021
Suporte para compartilhamento de dados entre contas	Agora é possível compartilhar dados entre Contas da AWS.	30 de abril de 2021
Suporte para consultas de dados hierárquicos com CTE recursiva	Agora é possível usar uma expressão de tabela comum (CTE) recursiva em seu SQL.	29 de abril de 2021
Suporte para consultas entre bancos de dados	Agora é possível consultar dados em bancos de dados de um cluster.	10 de março de 2021

Suporte para controle de acesso refinado em comandos COPY e UNLOAD	Agora é possível conceder o privilégio de executar comandos COPY e UNLOAD para usuários e grupos específicos no cluster do Amazon Redshift para criar uma política de controle de acesso mais refinada.	12 de janeiro de 2021
Suporte ao JSON nativo e dados semiestruturados	Agora, você pode definir o tipo de dados SUPER.	9 de dezembro de 2020
Suporte à consulta federada para o MySQL	Agora você pode escrever uma consulta federada em um mecanismo MySQL suportado.	9 de dezembro de 2020
Suporte ao compartilhamento de dados	Agora você pode compartilhar dados entre clusters do Amazon Redshift.	9 de dezembro de 2020
Suporte à otimização automática de tabela	Agora você pode definir distribuição automática e chaves de classificação.	9 de dezembro de 2020
Suporte ao Amazon Redshift ML	Agora, você pode criar, treinar e implantar modelos de Machine Learning (ML).	8 de dezembro de 2020
Suporte para atualização automática e regravação de consultas de visões materializadas	Agora você pode manter as visões materializadas atualizadas com a atualização automática e a performance da consulta pode ser aprimorado com a regravação automática.	11 de novembro de 2020

Suporte a tipos de dados TIME e TIMETZ	Agora você pode criar tabelas com os tipos de dados TIME e TIMETZ. O tipo de dados TIME armazena a hora do dia sem informações de fuso horário e o TIMETZ armazena a hora do dia, incluindo informações de fuso horário	11 de novembro de 2020
Compatibilidade com UDFs do Lambda e tokenização	Agora é possível gravar UDFs do Lambda para habilitar a tokenização externa de dados.	26 de outubro de 2020
Suporte para alterar uma codificação de coluna de tabela	Agora você pode alterar uma codificação de coluna da tabela.	20 de outubro de 2020
Suporte para consultas entre bancos de dados	O Amazon Redshift agora pode consultar bancos de dados em um cluster.	15 de outubro de 2020
Suporte a esboços do HyperLogLog	O Amazon Redshift agora pode armazenar e processar HyperLogLogSketches.	2 de outubro de 2020
Suporte para Apache Hudi e Delta Lake	Aprimoramentos para a criação de tabelas externas para Redshift Spectrum.	24 de setembro de 2020
Suporte a melhorias na consulta de dados espaciais	Os aprimoramentos incluem o carregamento de um shapefile e várias novas funções SQL espaciais.	15 de setembro de 2020
Tabelas externas de suporte a visão materializada	Você pode criar visões materializadas no Amazon Redshift que fazem referência a fontes de dados externas.	19 de junho de 2020

Suporte a gravação em tabela externa	É possível gravar em tabelas externas executando CREATE EXTERNAL TABLE AS SELECT para gravar em uma tabela externa nova ou INSERT INTO para inserir dados em uma tabela externa existente.	8 de junho de 2020
Suporte a controles de armazenamento para esquemas	Atualizações de comandos e visualizações que gerenciam controles de armazenamento para esquemas.	2 de junho de 2020
Suporte para disponibilidade geral de consulta federada	Informações atualizadas sobre a consulta de dados com consultas federadas.	16 de abril de 2020
Suporte para funções espaciais adicionais	Adicionadas descrições de funções espaciais adicionais.	2 de abril de 2020
Suporte para disponibilidade geral de visões materializadas	As visões materializadas geralmente estão disponíveis a partir da versão do cluster 1.0.13059.	19 de fevereiro de 2020
Suporte para privilégios em nível de coluna	Os privilégios de nível de coluna estão disponíveis a partir da versão do cluster 1.0.13059.	19 de fevereiro de 2020
ALTER TABLE	Você pode usar um comando ALTER TABLE com a cláusula ALTER DISTSTYLE ALL para alterar o estilo de distribuição de uma tabela.	11 de fevereiro de 2020

Suporte para consulta federada	O guia foi atualizado para descrever a consulta federada com um comando CREATE EXTERNAL SCHEMA atualizado.	3 de dezembro de 2019
Suporte para exportação de data lake	O guia foi atualizado para descrever novos parâmetros do comando UNLOAD.	3 de dezembro de 2019
Suporte a dados espaciais	O guia foi atualizado para descrever o suporte a dados espaciais.	21 de novembro de 2019
Compatibilidade com o novo console	Guia atualizado para descrever o novo console do Amazon Redshift.	11 de novembro de 2019
Suporte à classificação automática de tabela	O Amazon Redshift pode classificar uma tabela automaticamente.	7 de novembro de 2019
Suporte à opção VACUUM BOOST	É possível usar a opção BOOST ao limpar tabelas.	7 de novembro de 2019
Suporte para colunas IDENTITY comuns	É possível criar tabelas com colunas IDENTITY padrão.	19 de setembro de 2019
Suporte para a codificação de compactação AZ	Você pode codificar algumas colunas com a codificação de compactação AZ64.	19 de setembro de 2019
Suporte à prioridade de consulta	É possível definir a prioridad e de consulta de uma fila do WLM.	22 de agosto de 2019

Suporte para AWS Lake Formation	Você pode usar um catálogo de dados do Lake Formation com o Amazon Redshift Spectrum.	8 de agosto de 2019
COMPUPDATE PRESET	Você pode usar um comando COPY com COMPUPDATE PRESET para permitir que o Amazon Redshift escolha a codificação de compactação.	13 de junho de 2019
ALTER COLUMN	Você pode usar um comando ALTER TABLE com ALTER COLUMN para aumentar o tamanho de uma coluna VARCHAR.	22 de maio de 2019
Suporte para os procedimentos armazenados	Você pode definir procedimentos armazenados PL/pgSQL no Amazon Redshift.	24 de abril de 2019
Suporte a uma configuração de gerenciamento de workload (WLM) automático	Você pode habilitar o Amazon Redshift para ser executado com WLM automático.	24 de abril de 2019
UNLOAD para Zstandard	Você pode usar o comando UNLOAD para aplicar a compactação padrão Z a arquivos de texto e valores separados por vírgula (CSV) descarregados no Amazon S3.	3 de abril de 2019

Escalabilidade simultânea	Quando a escalabilidade de simultaneidade está habilitado, o Amazon Redshift adiciona automaticamente capacidade de cluster adicional quando você precisa para processar um aumento nas consultas de leitura simultâneas.	21 de março de 2019
UNLOAD para CSV	Use o comando UNLOAD para descarregar em um arquivo formatado como texto de CSV.	13 de março de 2019
Estilo de distribuição AUTO	Para habilitar a distribuição automática, você pode especificar o estilo de distribuição AUTO com uma instrução CREATE TABLE. Quando você habilita a distribuição automática, o Amazon Redshift atribui um estilo de distribuição ideal com base nos dados da tabela. A mudança na distribuição ocorre em segundo plano, em poucos segundos.	23 de janeiro de 2019
COPY do Parquet é compatível com SMALLINT	Agora o COPY oferece suporte ao carregamento de arquivos formatados do Parquet em colunas que usam o tipo de dados SMALLINT. Para obter mais informações, consulte COPY de formatos de dados colunar.	2 de janeiro de 2019

DROP EXTERNAL DATABASE	Você pode descartar um banco de dados externo incluindo o comando DROP EXTERNAL DATABASE com um comando DROP SCHEMA.	3 de dezembro de 2018
UNLOAD entre regiões	Você pode UNLOAD para um bucket do Amazon S3 em outra região da AWS, especific ando o parâmetro REGION.	31 de outubro de 2018
Exclusão de vacuum automática	O Amazon Redshift executa automaticamente uma operação VACUUM DELETE em segundo plano, então você raramente, ou nunca, precisa executar um DELETE ONLY. O Amazon Redshift programa o VACUUM DELETE para execução durante períodos de carga reduzida e pausa a operação durante períodos de alta carga.	31 de outubro de 2018
Distribuição automática	Quando você não especific a um estilo de distribuição com uma instrução CREATE TABLE, o Amazon Redshift atribui um estilo de distribuição ideal com base nos dados da tabela. A mudança na distribui ção ocorre em segundo plano, em poucos segundos.	31 de outubro de 2018

Controle de acesso refinado para o AWS Glue Data Catalog	Agora você pode especificar os níveis de acesso aos dados armazenados no AWS Glue Data Catalog.	15 de outubro de 2018
UNLOAD com tipos de dados	Você pode especificar a opção MANIFEST VERBOSE com um comando UNLOAD para adicionar metadados ao arquivo manifesto, incluindo os nomes e tipos de dados de colunas, tamanhos de arquivo e contagens de linha.	10 de outubro de 2018
Adicionar várias partições usando uma única instrução ALTER TABLE	Para tabelas externas do Redshift Spectrum, você pode combinar várias cláusulas PARTITION em um único comando ALTER TABLE ADD. Para obter mais informações, consulte Exemplos de alteração de tabelas externas.	10 de outubro de 2018
UNLOAD com cabeçalho	Você pode especificar a opção HEADER com um comando UNLOAD para adicionar uma linha de cabeçalho contendo nomes de coluna na parte superior de cada arquivo de saída.	19 de setembro de 2018
Novas tabelas de sistema e visualizações	Documentação SVL_S3Retries, SVL_USER_INFO e STL_DISK_FULL_DIAG adiciona.	31 de agosto de 2018

Suporte para dados aninhados no Amazon Redshift Spectrum	Agora você pode consultar dados armazenados em ninho nas tabelas do Amazon Redshift Spectrum. Para obter mais informações, consulte Tutorial: Consultar dados aninhados com Amazon Redshift Spectrum.	8 de agosto de 2018
SQA ativado por padrão	A aceleração de consulta curta (SQA) agora está ativada por padrão para todos clusters novos. SQA usa machine learning para fornecer maior performance, resultados mais rápidos e melhor previsibi lidade do tempo de execução das consultas. Para obter mais informações, consulte Aceleração de consulta curta.	8 de agosto de 2018
Advisor do Amazon Redshift	Agora é possível obter recomendações personali zadas sobre como melhorar a performance do cluster e reduzir custos operacionais do Advisor do Amazon Redshift. Para obter mais informações, consulte Advisor do Amazon Redshift.	26 de julho de 2018
Referência imediata de alias	Agora é possível fazer referência a uma expressão usada como alias imediatam ente depois de defini-la. Para obter mais informações, consulte Lista SELECT.	18 de julho de 2018

Especificar o tipo de compactação ao criar uma tabela externa	Agora, você pode especificar o tipo de compactação ao criar uma tabela externa com o Amazon Redshift Spectrum. Para obter mais informações, consulte Criação de tabelas externas.	27 de junho de 2018
PG_LAST_UNLOAD_ID	Documentação adicionada para uma nova função de informações do sistema: PG_LAST_UNLOAD_ID. Para obter mais informações, consulte PG_LAST_UNLOAD_ID.	27 de junho de 2018
ALTER TABLE RENAME COLUMN	ALTER TABLE agora oferece suporte à renomeação de colunas em tabelas externas. Para obter mais informações, consulte Exemplos de alteração de tabelas externas.	7 de junho de 2018

Atualizações anteriores

A tabela a seguir descreve as alterações importantes em cada versão do Guia do desenvolvedor de banco de dados do Amazon Redshift antes de junho de 2018.

Alteração	Descrição	Alterado em
COPY do Parquet inclui SMALLINT	Agora o COPY oferece suporte ao carregamento de arquivos formatados do Parquet em colunas que usam o tipo de dados SMALLINT. Para obter mais informações, consulte COPY de formatos de dados colunar.	2 de janeiro de 2019

Alteração	Descrição	Alterado em
COPY de formatos colunares	O COPY agora oferece suporte ao carregamento de arquivos no Amazon S3 que usam os formatos de dados colunares Parquet e ORC. Para obter mais informações, consulte COPY de formatos de dados colunar.	17 de maio de 2018
Tempo máximo dinâmico para SQA	Por padrão, WLM agora atribui dinamicamente um valor para o tempo máximo de execução de SQA baseado na análise de workload de seu cluster. Para obter mais informações, consulte Tempo máximo de execução para consultas breves.	17 de maio de 2018
Nova coluna em STL_LOAD_COMMITS	A tabela do sistema STL_LOAD_COMMITS tem uma nova coluna, `file_format` .	10 de maio de 2018
Colunas novas em STL_HASHJOIN e outras tabelas de log do sistema	A tabela do sistema STL_HASHJOIN tem três novas colunas, `hash_segment` , `hash_step` e `checksum`. Além disso, `checksum` foi adicionado a STL_MERGEJOIN, STL_NESTLOOP, STL_HASH, STL_SCAN, STL_SORT, STL_LIMIT e STL_PROJECT.	17 de maio de 2018
Novas colunas em STL_AGGR	A tabela de sistema STL_AGGR tem duas novas colunas `resizes` e `flushable` .	19 de abril de 2018
Opções novas para funções REGEX	Para as funções REGEXP_INSTR e REGEXP_SUBSTR, agora você pode especificar qual ocorrência de uma correspondência usar e se uma correspondência diferenciando maiúsculas de minúsculas deve ser realizada. REGEXP_INSTR também permite especificar se a posição do primeiro caractere da correspondência ou a posição do primeiro caractere depois do final da correspondência deve ser retornada.	22 de março de 2018

Alteração	Descrição	Alterado em
Colunas novas em tabelas do sistema	As colunas tombstonedblocks, tossedblocks e batched_by foram adicionadas à tabela do sistema STL_COMMIT_STATS. A coluna localslice foi adicionada à visualização do sistema STV_SLICES.	22 de março de 2018
Adicionar e descartar colunas em tabelas externas	ALTER TABLE agora oferece suporte a ADD COLUMN e a DROP COLUMN para tabelas externas do Amazon Redshift Spectrum.	22 de março de 2018
Novas regiões da AWS do Redshift Spectrum	O Redshift Spectrum agora está disponível nas regiões de Mumbai e de São Paulo. Para obter uma lista de regiões compatíveis, consulte Regiões do Amazon Redshift Spectrum.	22 de março de 2018
Limite de tabela aumentado para 20.000	O número máximo de tabelas agora é 20.000 para tipos de nós de cluster 8xlarge. O limite para tipos de nós grandes e extragrandes é 9.900. Para obter mais informações, consulte Limites e cotas.	13 de março de 2018
O Redshift Spectrum oferece suporte a JSON e Ion	Com o Redshift Spectrum, você pode fazer referênci a a arquivos que usam dados escalares nos formatos de dados JSON ou Ion. Para obter mais informações, consulte CREATE EXTERNAL TABLE.	26 de fevereiro de 2018
Encadeamento de funções do IAM para Redshift Spectrum	Você pode encadear funções do AWS Identity and Access Management (IAM) para que seu cluster possa assumir outros papéis não anexados ao cluster, incluindo funções pertencentes a outra conta da AWS. Para obter mais informações, consulte Encadeamento de funções do IAM no Amazon Redshift Spectrum.	1 de fevereiro de 2018
ADD PARTITION suporta IF NOT EXISTS	A cláusula ADD PARTITION para ALTER TABLE agora suporta uma opção IF NOT EXISTS. Para obter mais informações, consulte ALTER TABLE.	11 de janeiro de 2018

Alteração	Descrição	Alterado em
Dados DATE para tabelas externas	As tabelas externas do Redshift Spectrum agora suportam o tipo de dados DATE. Para obter mais informações, consulte CREATE EXTERNAL TABLE.	11 de janeiro de 2018
Novas regiões da AWS do Redshift Spectrum	O Redshift Spectrum agora está disponível nas regiões de Cingapura, Sydney, Seul e Frankfurt. Para obter uma lista de regiões da AWS compatíveis, consulte Regiões do Amazon Redshift Spectrum.	16 de novembro de 2017
Aceleração de consulta curta no gerenciamento de workload do Amazon Redshift (WLM)	A aceleração de consultas breves (SQA) prioriza as consultas de curta execução sobre as consultas de execução demorada. A SQA executa consultas breves em um espaço dedicado, de maneira que as consultas SQA não sejam forçadas a esperar em filas atrás de consultas mais demoradas. Com a SQA, consultas breves são iniciadas com mais rapidez e os usuários veem os resultados mais cedo. Para obter mais informações, consulte Aceleração de consulta breve.	16 de novembro de 2017
O WLM reatribui as consultas saltadas	Em vez de cancelar e reiniciar uma consulta saltada, o gerenciamento de workload do Amazon Redshift (WLM) agora reatribui as consultas elegíveis a uma nova fila. Quando o WLM reatribui uma consulta, ele move a consulta para a nova fila e continua a execução, o que economiza tempo e recursos do sistema. As consultas saltadas que não são qualifica das para reatribuição são reiniciadas ou canceladas. Para obter mais informações, consulte Salto na fila de consultas do WLM.	16 de novembro de 2017

Alteração	Descrição	Alterado em
Acesso ao log do sistema para usuários	Na maioria das tabelas de log do sistema que são visíveis para usuários, as linhas geradas por um outro usuário são invisíveis para um usuário regular. Para permitir que um usuário regular veja todas as linhas nas tabelas visíveis para usuários, incluindo as linhas geradas por um outro usuário, execute ALTER USER ou CREATE USER, e defina o parâmetro SYSLOG ACCESS como UNRESTRICTED.	16 de novembro de 2017
Armazenamento em cache dos resultados	Com Armazenamento em cache dos resultados, quando você executa uma consulta, o Amazon Redshift armazena em cache o resultado. Quando você executa a consulta novamente, o Amazon Redshift verifica se há uma cópia em cache válida do resultado da consulta. Se for encontrada uma correspondência no cache de resultados, o Amazon Redshift usará o resultado armazenado em cache e não executará a consulta. O armazenamento em cache de resultados é ativado por padrão. Para desativar o armazenamento em cache, defina o parâmetro de configuração enable_result_cache_for_session como `off`.	16 de novembro de 2017
Funções dos metadados de coluna	PG_GET_COLS e PG_GET_LATE_BINDING_VIEW_COLS retornam metadados de coluna para as tabelas, visualizações e visualizações de vinculação tardia do Amazon Redshift.	16 de novembro de 2017
Salto na fila de consultas do WLM para CTAS	O gerenciamento de workload do Amazon Redshift (WLM) agora oferece suporte para salto de fila de consulta para instruções CREATE TABLE AS (CTAS), bem como consultas somente leitura, como instruções SELECT. Para obter mais informações, consulte Salto na fila de consultas do WLM.	19 de outubro de 2017

Alteração	Descrição	Alterado em
Arquivos manifesto do Amazon Redshift Spectrum	Ao criar uma tabela externa do Redshift Spectrum, você pode especificar um arquivo manifesto que lista os locais dos arquivos de dados no Amazon S3. Para obter mais informações, consulte CREATE EXTERNAL TABLE.	19 de outubro de 2017
Regiões da AWS do Amazon Redshift Spectrum	O Redshift Spectrum agora está disponível nas regiões da UE (Irlanda) e da Ásia-Pacífico (Tóquio). Para obter uma lista de regiões da AWS compatíveis, consulte Limitações do Amazon Redshift Spectrum.	19 de outubro de 2017
Formatos de arquivos adicionados ao Amazon Redshift Spectrum	Agora você pode criar tabelas externas do Redshift Spectrum baseadas nos formatos de arquivos de dados Regex, OpenCSV e Avro. Para obter mais informações, consulte CREATE EXTERNAL TABLE.	5 de outubro de 2017
Pseudocolunas para tabelas externas do Amazon Redshift Spectrum	Você pode selecionar as pseudocolunas `$path` e `$size` em uma tabela externa do Redshift Spectrum para visualizar a localização e o tamanho dos arquivos de dados referenciados no Amazon S3. Para obter mais informações, consulte Pseudocolunas.	5 de outubro de 2017
Funções para validar o JSON	Use as funções IS_VALID_JSON e IS_VALID_JSON_ARRAY para verificar o formato válido do JSON. As outras funções JSON agora têm um argumento `null_if_invalid` opcional.	5 de outubro de 2017
LISTAGG DISTINCT	Use a cláusula DISTINCT com a função agregada LISTAGG e a função de janela LISTAGG para eliminar valores duplicados da expressão especificada antes de concatenar.	5 de outubro de 2017
Exibir nomes de coluna em caixa alta	Para exibir os nomes de coluna nos resultados de SELECT em caixa alta, defina o parâmetro de configuração describe_field_name_in_uppercase como `true`.	5 de outubro de 2017

Alteração	Descrição	Alterado em
Ignorar linhas de cabeçalho em tabelas externas	Você pode definir a propriedade `skip.header.line.count` no comando CREATE EXTERNAL TABLE para ignorar as linhas de cabeçalho no início dos arquivos de dados do Redshift Spectrum.	5 de outubro de 2017
Contagem de linhas da verificação	As regras de monitoramento de consulta do WLM usa a métrica scan_row_count para retornar o número de linhas em uma etapa de verificação. A contagem de linhas é o número total de linhas emitidas antes da filtragem das linhas marcadas para exclusão (linhas fantasmas) e antes da aplicação dos filtros de consulta definidos pelo usuário. Para obter mais informações, consulte Métricas de monitoramento de consultas para o Amazon Redshift provisionado.	21 de setembro de 2017
Funções definidas pelo usuário SQL	Uma UDF (função definida pelo usuário) SQL escalar incorpora uma cláusula SQL SELECT que é executada quando a função é chamada e retorna um valor único. Para obter mais informações, consulte UDFs escalares de SQL.	31 de agosto de 2017

Alteração	Descrição	Alterado em
Visualizações de vinculação tardia	Uma exibição de vinculação tardia não é vinculada a objetos de banco de dados subjacentes, como tabelas e funções definidas pelo usuário. Como resultado , não há nenhuma dependência entre a exibição e os objetos aos quais ela faz referência. Você pode criar uma exibição mesmo se não houver objetos referenciados. Como não há dependência, você pode descartar ou alterar um objeto referenciado sem afetar a visualização. O Amazon Redshift não verifica as dependências até que a exibição seja consultada. Para criar uma exibição de vinculação tardia, especifiq ue a cláusula WITH NO SCHEMA BINDING com a instrução CREATE VIEW. Para obter mais informaçõ es, consulte CREATE VIEW.	31 de agosto de 2017
Função OCTET_LENGTH	OCTET_LENGTH retorna o tamanho da string especificada como número de bytes.	18 de agosto de 2017
Tipos de arquivos ORC e Grok compatíveis	O Amazon Redshift Spectrum agora oferece suporte aos formatos de dados ORC e Grok para arquivos de dados do Redshift Spectrum. Para obter mais informações, consulte Arquivos de dados para consultas no Amazon Redshift Spectrum.	18 de agosto de 2017
RegexSerDe agora compatível	O Amazon Redshift Spectrum agora oferece suporte ao formato de dados RegexSerDe. Para obter mais informações, consulte Arquivos de dados para consultas no Amazon Redshift Spectrum.	19 de julho de 2017
Colunas novas adicionadas a SVV_TABLES e SVV_COLUMNS	As colunas `domain_name` e `remarks` foram adicionadas a SVV_COLUMNS. Uma coluna de observações foi adicionada a SVV_TABLES.	19 de julho de 2017

Alteração	Descrição	Alterado em
Exibições de sistema SVV_TABLES e SVV_COLUMNS	As exibições de sistema SVV_TABLES e SVV_COLUMNS fornecem informações sobre colunas e dão outros detalhes para tabelas e exibições locais e externas.	7 de julho de 2017
VPC não é mais necessária para Amazon Redshift Spectrum com o metastore Amazon EMR Hive	O Redshift Spectrum removeu o requisito de que o cluster do Amazon Redshift e o cluster do Amazon EMR devem estar na mesma VPC e na mesma sub-rede ao usar um metastore Amazon EMR Hive. Para obter mais informações, consulte Trabalhar com catálogos externos no Amazon Redshift Spectrum.	7 de julho de 2017
UNLOAD para tamanhos de arquivos menores	Por padrão, UNLOAD cria vários arquivos no Amazon S3 com tamanho máximo de 6,2 GB. Para criar arquivos menores, especifique MAXFILESIZE com o comando UNLOAD. Você pode especificar um tamanho de arquivo máximo de 5 MB e 6,2 GB. Para obter mais informações, consulte UNLOAD.	7 de julho de 2017
TABLE PROPERTIES	Agora você pode definir o parâmetro TABLE PROPERTIES numRows de CREATE EXTERNAL TABLE ou ALTER TABLE para atualizar estatísti cas da tabela a fim de refletir o número de linhas na tabela.	6 de junho de 2017

Alteração	Descrição	Alterado em
ANALYZE PREDICATE COLUMNS	Para economizar tempo e recursos do cluster, você pode optar por analisar somente as colunas com mais chances de serem usadas como predicados. Quando você executa ANALYZE com a cláusula PREDICATE COLUMNS, a operação de análise inclui somente colunas que foram usadas em um união, uma condição de filtro ou um grupo por cláusula, ou que são usadas como uma chave de classificação ou uma chave de distribuição. Para obter mais informações, consulte Análise de tabelas.	25 de maio de 2017
Políticas do IAM do Amazon Redshift Spectrum	Para conceder acesso a um bucket do Amazon S3 usando apenas o Redshift Spectrum, você pode incluir uma condição que permite o acesso para o agente do usuário `AWS Redshift/Spectrum`. Para obter mais informações, consulte Políticas do IAM do Amazon Redshift Spectrum.	25 de maio de 2017
Varredura recursiva do Amazon Redshift Spectrum	O Redshift Spectrum agora verifica arquivos em subpastas, bem como a pasta especificada no Amazon S3. Para obter mais informações, consulte Tabelas externas para o Redshift Spectrum.	25 de maio de 2017
Regras de monitoramento de consulta	Usando as regras de monitoramento de consulta do WLM, você pode definir limites de performance baseados em métricas para filas do WLM e especificar a ação a ser tomada quando uma consulta ultrapassar esses limites — registrar, saltar ou anular. Você define regras de monitoramento de consultas como parte da configuração do Workload Management (WLM – Gerenciamento do workload). Para obter mais informações, consulte Regras de monitoramento de consulta do WLM.	21 de abril de 2017

Alteração	Descrição	Alterado em
Amazon Redshift Spectrum	Usando o Redshift Spectrum, você pode consultar e recuperar dados de arquivos no Amazon S3 com eficiência, sem ter que carregar os dados nas tabelas. As consultas do Redshift Spectrum são executadas muito rapidamente em grandes conjuntos de dados porque o Redshift Spectrum verifica os arquivos de dados diretamente no Amazon S3. Grande parte do processamento ocorre na camada do Amazon Redshift Spectrum e a maioria dos dados permanece no Amazon S3. Vários clusters podem consultar simultaneamente o mesmo conjunto de dados no Amazon S3 sem a necessidade de fazer cópias dos dados para cada cluster. Para obter mais informações, consulte Amazon Redshift Spectrum.	19 de abril de 2017
Novas tabelas de sistema para dar suporte ao Redshift Spectrum	Estas novas visualizações de sistema foram adicionadas para dar suporte ao Redshift Spectrum: • SVL_S3QUERY • SVL_S3QUERY_SUMMARY • SVV_EXTERNAL_COLUMNS • SVV_EXTERNAL_DATABASES • SVV_EXTERNAL_PARTITIONS • SVV_EXTERNAL_TABLES • PG_EXTERNAL_SCHEMA	19 de abril de 2017

Alteração	Descrição	Alterado em
Função agregada APPROXIMATE PERCENTILE_DISC	Agora a função agregada APPROXIMATE PERCENTILE_DISC está disponível.	4 de abril de 2017
Criptografia no lado do servidor com KMS	Agora você pode descarregar dados no Amazon S3 usando criptografia do lado do servidor com uma chave AWS Key Management Service (SSE-KMS). Além disso, agora COPY carrega de maneira transparente arquivos de dados criptografados em KMS do Amazon S3. Para obter mais informações, consulte UNLOAD.	9 de fevereiro de 2017
Nova sintaxe de autorização	Agora você pode usar os parâmetros IAM_ROLE, MASTER_SYMMETRIC_KEY, ACCESS_KEY_ID, SECRET_ACCESS_KEY e SESSION_TOKEN para dar autorização e acessar informações de comandos COPY, UNLOAD e CREATE LIBRARY. A nova sintaxe de autorização oferece uma alternativa mais flexível ao fornecimento de um único argumento de string para o parâmetro CREDENTIALS. Para obter mais informações, consulte Parâmetros de autorização.	9 de fevereiro de 2017
Aumento do limite do esquema	Agora você pode criar até 9.900 esquemas por cluster. Para obter mais informações, consulte CREATE SCHEMA.	9 de fevereiro de 2017
Codificação da tabela padrão	CRIAR TABELA e ALTER TABLE agora atribuem codificação de compactação LZO à maioria das colunas novas. As colunas definidas como chaves de classificação, as colunas definidas como tipos de dados BOOLEAN, REAL ou DOUBLE PRECISION recebem a codificação RAW, por padrão. Para obter mais informações, consulte ENCODE.	6 de fevereiro de 2017

Alteração	Descrição	Alterado em
Codificação da compactação ZSTD	Agora o Amazon Redshift permite a codificação de compactação de coluna ZSTD.	19 de janeiro de 2017
Funções agregadas PERCENTILE_CONT e MEDIAN	Agora PERCENTILE_CONT e MEDIAN estão disponíveis como funções agregadas, bem como funções janela.	19 de janeiro de 2017
Registro em log do usuário da User-Defined Function (UDF – Função definida pelo usuário)	Você pode usar o módulo de log Python para criar mensagens de erro e alerta definidas pelo usuário em suas UDFs. Depois da execução da consulta, você poderá consultar a exibição do sistema SVL_UDF_LOG para recuperar mensagens registradas. Para obter mais informações sobre mensagens definidas pelo usuário, consulte . Registro em log de erros e alertas em UDFs do Python	8 de dezembro de 2016
Redução de ANALYZE COMPRESSION estimada	Agora o comando ANALYZE COMPRESSION informa uma estimativa da redução de porcentagem no espaço em disco para cada coluna. Para obter mais informações, consulte ANALYZE COMPRESSION.	10 de novembro de 2016
Limites de conexão	Agora você pode definir um limite para o número de conexões de banco de dados que um usuário pode abrir simultaneamente. Você também pode limitar o número de conexões simultâneas para um banco de dados. Para obter mais informações, consulte CREATE USER e CREATE DATABASE.	10 de novembro de 2016

Alteração	Descrição	Alterado em
Melhoria na ordem de classificação de COPY	Agora COPY adiciona automaticamente novas linhas à região classificada da tabela quando você carrega os dados na ordem da chave de classificação. Para saber os requisitos específicos a fim de permitir essa melhoria, consulte Carregar os dados por ordem de chave de classificação	10 de novembro de 2016
CTAS com compactação	Agora CREATE TABLE AS (CTAS) distribui automatic amente codificações de compactação para novas tabelas com base no tipo de dados da coluna. Para obter mais informações, consulte Herança de atributos de coluna e tabela.	28 de outubro de 2016
Time stamp com tipo de dados de fuso horário	Agora o Amazon Redshift dá suporte ao timestamp com tipo de dados de fuso horário (TIMESTAMPTZ). Além disso, várias funções novas foram adicionadas para dar suporte ao novo tipo de dados. Para obter mais informações, consulte Perfis de data e hora.	29 de setembro de 2016
Limite de análise	Para reduzir o tempo de processamento e aprimorar a performance geral do sistema para operações ANALYZE, o Amazon Redshift vai ignorar a análise de uma tabela se a porcentagem de linhas que foram alteradas desde a última vez em que o comando ANALYZE tiver sido executado for menor que o limite de análise especificado pelo parâmetro analyze_threshold_percent. Por padrão, `analyze_threshold_percent` é 10.	9 de agosto de 2016
Nova tabela do sistema STL_RESTARTED_SESSIONS	Quando o Amazon Redshift reiniciar uma sessão, STL_RESTARTED_SESSIONS registrará o novo ID de processo (PID) e o PID anterior.	9 de agosto de 2016

Alteração	Descrição	Alterado em
Atualizada a documentação das funções de data e a hora	Adicionado um resumo de funções com links para Perfis de data e hora, e atualizadas as referências da função para fins de consistência.	24 de junho de 2016
Novas colunas em STL_CONNECTION_LOG	A tabela do sistema STL_CONNECTION_LOG tem duas colunas novas para acompanhar conexões SSL. Se sempre carregar logs de auditoria em uma tabela do Amazon Redshift, você precisará adicionar as novas colunas a seguir à tabela de destino: sslcompression e sslexpansion.	5 de maio de 2016
Senha hash MD5	Você pode especificar a senha para um comando CREATE USER ou ALTER USER fornecendo a string de hash MD5 da senha e do nome do usuário.	21 de abril de 2016
Nova coluna em STV_TBL_PERM	A coluna `backup` na exibição do sistema STV_TBL_PERM indica se a tabela está incluída em snapshots do cluster. Para obter mais informações, consulte BACKUP.	21 de abril de 2016
Sem tabelas de backup	Para tabelas que não contêm dados críticos, como tabelas de teste, você pode especificar BACKUP NO em sua instrução CRIAR TABELA ou CREATE TABLE AS ou para evitar que o Amazon Redshift inclua a tabela em snapshots automatizados ou manuais. O uso de tabelas sem backup economiza tempo de processamento ao criar e restaurar snapshots e reduz o espaço de armazenamento no Amazon S3.	7 de abril de 2016

Alteração	Descrição	Alterado em
Limite de exclusão de VACUUM	Por padrão, o comando VACUUM agora recupera o espaço, de maneira que pelo menos 95 por cento das linhas restantes não sejam marcadas para exclusão. Dessa forma, VACUUM normalmente precisa de muito menos tempo para a fase de exclusão, em comparação com a recuperação de 100 por cento das linhas excluídas. Você pode alterar o limite padrão de uma tabela única incluindo o parâmetro TO threshold PERCENT ao executar o comando VACUUM.	7 de abril de 2016
Tabela do sistema SVV_TRANSACTIONS	O sistema SVV_TRANSACTIONS exibe informaçõ es de registros sobre as transações que mantêm bloqueios em tabelas no banco de dados no momento.	7 de abril de 2016
Usar funções do IAM para acessar outros recursos da AWS	Para mover dados entre seu cluster e outro recurso da AWS, como Amazon S3, DynamoDB, Amazon EMR ou Amazon EC2, seu cluster deve ter permissão para acessar o recurso e executar as ações necessári as. Como uma alternativa mais segura para fornecer um par de chaves de acesso com comandos COPY, UNLOAD ou CREATE LIBRARY, agora você pode especificar uma função do IAM usada pelo cluster na autenticação e na autorização. Para obter mais informações, consulte Controle de acesso com base em função.	29 de março de 2016
Limite de classific ação de VACUUM	Agora o comando VACUUM ignora a fase de classific ação para qualquer tabela em que mais de 95 por cento das linhas da tabela já estejam classificadas. Você pode alterar o limite de classificação padrão de uma tabela única incluindo o parâmetro TO threshold PERCENT ao executar o comando VACUUM.	17 de março de 2016

Alteração	Descrição	Alterado em
Novas colunas em STL_CONNECTION_LOG	A tabela do sistema STL_CONNECTION_LOG tem três colunas novas. Se você carregar logs de auditoria rotineiramente em uma tabela do Amazon Redshift, precisará adicionar as seguintes novas colunas à tabela de destino: sslversion, sslcipher e mtu.	17 de março de 2016
UNLOAD com compactação bzip2	Agora você tem a opção de usar o comando UNLOAD utilizando a compactação bzip2.	8 de fevereiro de 2016
ALTER TABLE APPEND	ALTER TABLE APPEND acrescenta linhas a uma tabela de destino movendo dados de uma tabela de origem existente. ALTER TABLE APPEND geralmente é muito mais rápido do que as operações CREATE TABLE AS ou INSERT semelhantes, pois os dados são movidos, não duplicados.	8 de fevereiro de 2016
Salto na fila de consultas do WLM	Se o Workload Management (WLM – Gerenciamento do workload) cancelar uma consulta somente leitura, como uma instrução SELECT, o WLM tentará rotear a consulta para a próxima fila correspondente. Para obter mais informações, consulte Salto na fila de consultas do WLM.	7 de janeiro de 2016
ALTER DEFAULT PRIVILEGES	Você pode usar o comando ALTER DEFAULT PRIVILEGES para definir o conjunto padrão de privilégios de acesso a ser aplicado a objetos criados no futuro pelo usuário especificado.	10 de dezembro de 2015
compactação de arquivo bzip2	O comando COPY dá suporte ao carregamento de dados de arquivos que foram compactados usando-se bzip2.	10 de dezembro de 2015

Alteração	Descrição	Alterado em
NULLS FIRST e NULLS LAST	Você pode especificar se uma cláusula ORDER BY deve classificar NULLS antes ou depois no conjunto de resultados. Para obter mais informações, consulte Cláusula ORDER BY e Resumo da sintaxe de funções da janela.	19 de novembro de 2015
Palavra-c have REGION para CREATE LIBRARY	Se o bucket do Amazon S3 que contém os arquivos da biblioteca UDF não residir na mesma região da AWS que seu cluster do Amazon Redshift, você pode usar a opção REGION para especificar a região na qual os dados estão localizados. Para obter mais informações, consulte CREATE LIBRARY.	19 de novembro de 2015
User-Defined Functions (UDFs – Funções definidas pelo usuário) escalares	Agora você pode criar funções escalares personali zadas definidas pelo usuário para implementar a funcionalidade de processamento não SQL fornecida pelos módulos com suporte do Amazon Redshift na biblioteca padrão do Python 2.7 ou seus próprios UDFs personalizados com base na linguagem de programação Python. Para obter mais informações, consulte Funções definidas pelo usuário (UDFs) no Amazon Redshift.	11 de setembro de 2015
Propriedades dinâmicas na configuração do WLM	O parâmetro de configuração do WLM agora dá suporte à aplicação de algumas propriedades dinamicamente. Outras propriedades permanecem alterações estáticas e exigem que clusters associados sejam reiniciados, de maneira que as alterações feitas na configuração possam ser aplicadas. Para obter mais informações, consulte Propriedades de configura ção dinâmicas e estáticas do WLM e Gerenciamento do workload.	3 de agosto de 2015

Alteração	Descrição	Alterado em
Função LISTAGG	Função LISTAGG e Função de janela LISTAGG retornam uma string criada concatenando-se um conjunto de valores de coluna.	30 de julho de 2015
Parâmetro obsoleto	O parâmetro de configuração max_cursor_result_set_size está obsoleto. O tamanho dos conjuntos de resultados do cursor é restringido com base no tipo de nó do cluster. Para obter mais informações, consulte Restrições de cursor.	24 de julho de 2015
Revisada documentação do comando COPY	A referência de comando COPY foi amplamente revisada para tornar o material o mais amigável e acessível possível.	15 de julho de 2015
COPY no formato Avro	O comando COPY suporta o carregamento de dados no formato Avro de arquivos de dados no Amazon S3, Amazon EMR e de hosts remotos usando SSH. Para obter mais informações, consulte AVRO e Copiar de exemplos Avro.	8 de julho de 2015
STV_STARTUP_RECOVERY_STATE	A tabela do sistema STV_STARTUP_RECOVERY_STATE registra o estado das tabelas que estão temporariamente bloqueadas durante as operações de reinicialização do cluster. O Amazon Redshift aplica um bloqueio temporário nas tabelas, enquanto elas estão sendo processadas, para resolver transações velhas depois que um cluster é reiniciado.	25 de maio de 2015
ORDER BY opcional para funções de classificação	Agora a cláusula ORDER é opcional para determina das funções de classificação de janela. Para obter mais informações, consulte Função de janela CUME_DIST, Função de janela DENSE_RANK, Função de janela RANK, Função de janela NTILE, Função de janela PERCENT_RANK e Função de janela ROW_NUMBER.	25 de maio de 2015

Alteração	Descrição	Alterado em
Chaves de classificação intercalada	As chaves de classificação intercalada dão peso igual a cada coluna da chave de classificação. Usar chaves de tipo intercalado, em vez de chaves compostas padrão, melhora significativamente a performance em consultas que usam predicados restritivos em colunas de classificação secundária, especialmente em tabelas grandes. A classificação intercalada também melhora a performance geral quando várias consultas filtram colunas diferentes na mesma tabela. Para obter mais informações, consulte Chaves de classificação e CRIAR TABELA.	11 de maio de 2015
Revisado tópico sobre ajustar performance da consulta	Ajustar o desempenho da consulta foi expandido a fim de incluir novas consultas para analisar a performan ce da consulta e mais exemplos. Além disso, o tópico foi revisado para ser mais claro e mais completo. Práticas recomendadas do Amazon Redshift para criar consultas tem mais informações sobre como escrever consultar para melhorar a performance.	23 de março de 2015
SVL_QUERY _QUEUE_INFO	A exibição SVL_QUERY_QUEUE_INFO resume detalhes de consultas que passam um tempo em uma fila de consultas do WLM ou em uma fila de confirmaç ões.	19 de fevereiro de 2015
SVV_TABLE _INFO	Você pode usar a exibição SVV_TABLE_INFO para diagnosticar e resolver problemas no projeto da tabela que possam afetar a performance de uma consulta, inclusive problemas de codificação da compactaç ão, chaves de distribuição, estilo de classificação, distorção da distribuição de dados, tamanho da tabela e estatísticas.	19 de fevereiro de 2015

Alteração	Descrição	Alterado em
UNLOAD usa criptografia de arquivos no lado do servidor	O comando UNLOAD agora usa automaticamente a criptografia do lado do servidor (SSE) do Amazon S3 para criptografar todos os arquivos de dados de descarregamento. A criptografia no lado do servidor adiciona outra camada de segurança de dados com pouca ou nenhuma alteração na performance.	31 de outubro de 2014
Função de janela CUME_DIST	O Função de janela CUME_DIST calcula a distribui ção cumulativa de um valor dentro de uma janela ou partição.	31 de outubro de 2014
Função MONTHS_BE TWEEN	O Função MONTHS_BETWEEN determina o número de meses entre duas datas.	31 de outubro de 2014
Função NEXT_DAY	O Função NEXT_DAY retorna a data da primeira instância de um dia especificado posterior à data indicada.	31 de outubro de 2014
Função de janela PERCENT_RANK	O Função de janela PERCENT_RANK calcula a classificação percentual de uma determinada linha.	31 de outubro de 2014
Função de janela RATIO_TO_ REPORT	O Função de janela RATIO_TO_REPORT calcula a proporção de um valor para a soma dos valores em uma janela ou partição.	31 de outubro de 2014
Função TRANSLATE	O Função TRANSLATE substitui todas as ocorrências de caracteres especificados dentro de uma determina da expressão pelos substitutos especificados.	31 de outubro de 2014
Função NVL2	O Função NVL2 retorna um de dois valores, dependendo de uma expressão especificada ser avaliada como NULL ou NOT NULL.	16 de outubro de 2014
Função de janela MEDIAN	O Função de janela MEDIAN calcula o valor mediano para o intervalo de valores em uma janela ou partição.	16 de outubro de 2014

Alteração	Descrição	Alterado em
Cláusula ON ALL TABLES IN SCHEMA schema_name para comandos GRANT e REVOKE	Os comandos GRANT e REVOKE foram atualizados com uma cláusula ON ALL TABLES IN SCHEMA schema_name. Essa cláusula permite usar um único comando para alterar privilégios de todas as tabelas em um esquema.	16 de outubro de 2014
Cláusula IF EXISTS para comandos DROP SCHEMA, DROP TABLE, DROP USER e DROP VIEW	Os comandos DROP SCHEMA, DESCARTAR TABELA, DROP USER e DROP VIEW foram atualizados com uma cláusula IF EXISTS. Essa cláusula faz o comando não aplicar alterações e retornar uma mensagem, em vez de encerrar com um erro se o objeto especificado não existir.	16 de outubro de 2014
Cláusula IF NOT EXISTS para comandos CREATE SCHEMA e CREATE TABLE	Os comandos CREATE SCHEMA e CRIAR TABELA foram atualizados com uma cláusula IF NOT EXISTS. Essa cláusula faz o comando não aplicar alterações e retornar uma mensagem, em vez de encerrar com um erro se o objeto especificado já existir.	16 de outubro de 2014
Suporte a COPY para codificação UTF-16	Agora o comando COPY dá suporte ao carregamento de arquivos de dados que usam a codificação UTF-16, bem como a codificação UTF-8. Para obter mais informações, consulte ENCODING.	29 de setembro de 2014
Novo tutorial de gerenciamento do workload	Tutorial: Configuração de filas de gerenciamento do workload (WLM) manual orienta você em meio ao processo de configuração das filas de gerenciamento de workload (WLM) para melhorar o processamento da consulta e a alocação do recurso.	25 de setembro de 2014

Alteração	Descrição	Alterado em
Criptografia de 128 bits AES	O comando COPY agora suporta criptografia AES de 128 bits, bem como criptografia AES de 256 bits ao carregar de arquivos de dados criptografados usando a criptografia do cliente Amazon S3. Para obter mais informações, consulte Carregar arquivos de dados criptografados do Amazon S3.	29 de setembro de 2014
Função PG_LAST_UNLOAD_COUNT	A função PG_LAST_UNLOAD_COUNT retorna o número de linhas que foram processadas na operação UNLOAD mais recente. Para obter mais informações, consulte PG_LAST_UNLOAD_COUNT.	15 de setembro de 2014
Nova seção de solução de problemas de consultas	Solução de problemas de consulta fornece uma referência rápida para identificar e resolver alguns dos problemas mais comuns e mais sérios que você provavelmente encontrará com as consultas do Amazon Redshift.	7 de julho de 2014
Novo tutorial de carregamento de dados	Tutorial: Carregar dados do Amazon S3 orienta você em meio ao processo de carregamento de dados nas tabelas de banco de dados do Amazon Redshift de arquivos de dados em um bucket do Amazon S3 do início ao fim.	1º de julho de 2014
Função de janela PERCENTILE_CONT	Função de janela PERCENTILE_CONT é uma função de distribuição inversa que pressupõe um modelo de distribuição contínua. Ela pega um valor percentil e uma especificação de classificação e retorna um valor intercalar que cairia dentro do valor percentil fornecido em relação à especificação de classificação.	30 de junho de 2014
Função de janela PERCENTILE_DISC	Função de janela PERCENTILE_DISC é uma função de distribuição inversa que pressupõe um modelo de distribuição discreta. Ela utiliza um valor percentil e uma especificação de classificação e retorna um elemento do conjunto.	30 de junho de 2014

Alteração	Descrição	Alterado em
Funções GREATEST e LEAST	As funções Funções GREATEST e LEAST retornam o maior ou o menor valor de uma lista de expressões.	30 de junho de 2014
COPY entre regiões	O comando COPY suporta o carregamento de dados de um bucket do Amazon S3 ou tabela do Amazon DynamoDB que está localizado em uma região diferente do cluster do Amazon Redshift. Para obter mais informações, consulte REGION na referência do comando COPY.	30 de junho de 2014
Melhores práticas expandidas	Práticas recomendadas do Amazon Redshift foi expandido, reorganizado e movido para a parte superior da hierarquia de navegação a fim de torná-la detectável.	28 de maio de 2014
UNLOAD para um único arquivo	O comando UNLOAD pode, opcionalmente, descarregar dados da tabela em série para um único arquivo no Amazon S3, adicionando a opção PARALLEL OFF. Se o tamanho dos dados for maior que o tamanho de arquivo máximo de 6,2 GB, UNLOAD criará arquivos adicionais.	6 de maio de 2014
Funções REGEXP	As funções REGEXP_COUNT, REGEXP_INSTR e REGEXP_REPLACE manipulam strings com base na correspondência do padrão de expressão regular.	6 de maio de 2014
COPY do Amazon EMR	O comando COPY dá suporte ao carregamento direto dos dados pelos clusters do Amazon EMR. Para obter mais informações, consulte Carregar dados do Amazon EMR.	18 de abril de 2014

Alteração	Descrição	Alterado em
Aumento do limite de simultaneidade do WLM	Agora você pode configurar o gerenciamento de workload (WLM) para executar até 50 consultas simultâneas em filas de consultas definidas pelo usuário. Esse aumento dá aos usuários mais flexibilidade para gerenciar a performance do sistema modificando configurações do WLM. Para obter mais informações, consulte Implementar o WLM manual.	18 de abril de 2014
Novo parâmetro de configuração para gerenciar o tamanho do cursor	O parâmetro de configuração `max_cursor_result_set_size` define o tamanho máximo de dados, em megabytes, que pode ser retornado por conjunto de resultados do cursor de uma consulta maior. Esse valor de parâmetro também afeta o número de cursores simultâneos do cluster, o que permite configurar um valor que aumente ou diminua o número de cursores do cluster. Para obter mais informações, consulte DECLARE neste guia e "Grupo de parâmetros do Amazon Redshift" no Guia de gerenciamento de clusters do Amazon Redshift.	28 de março de 2014
COPY no formato JSON	O comando COPY suporta o carregamento de dados no formato JSON de arquivos de dados no Amazon S3 e de hosts remotos usando SSH. Para obter mais informações, consulte COPY no formato JSON observações sobre uso.	25 de março de 2014
Nova tabela do sistema STL_PLAN_INFO	A tabela STL_PLAN_INFO suplementa o comando EXPLAIN como outra maneira de observar planos de consulta.	25 de março de 2014
Nova função REGEXP_SUBSTR	O Função REGEXP_SUBSTR retorna os caracteres extraídos de uma string procurando um padrão de expressão regular.	25 de março de 2014

Alteração	Descrição	Alterado em
Novas colunas de STL_COMMIT_STATS	A tabela STL_COMMIT_STATS tem duas colunas novas: `numxids` e `oldestxid` .	6 de março de 2014
Suporte de COPY de SSH para gzip e lzop	O comando COPY dá suporte à compactação de gzip e lzop ao carregar dados por meio de uma conexão SSH.	13 de fevereiro de 2014
Novas funções	O Função de janela ROW_NUMBER retorna o número da linha atual. O Função STRTOL converte a expressão de string de um número da base especificada no valor inteiro equivalente. PG_CANCEL_BACKEND e PG_TERMINATE_BACKEND permitem que os usuários cancelem as consultas e conexões de sessão. A função LAST_DAY foi adicionada para compatibilidade Oracle.	13 de fevereiro de 2014
Nova tabela do sistema	A tabela do sistema STL_COMMIT_STATS fornece métricas relativas à performance da confirmação, inclusive a sincronização dos diversos estágios de confirmação e o número de blocos confirmados.	13 de fevereiro de 2014
FETCH com clusters de nó único	Durante o uso de um cursor em um cluster de único nó, o número máximo de linhas que podem ser buscadas usando-se o comando FETCH é 1000. FETCH FORWARD ALL não é compatível para clusters de nó único.	13 de fevereiro de 2014
Estratégia de redistribuição de DS_DIST_ALL_INNER	DS_DIST_ALL_INNER na saída do plano de explicação indica que toda a tabela interna inteira foi redistribuída para uma única fatia porque a tabela usa DISTSTYLE ALL. Para obter mais informações, consulte Exemplos de tipos de junção e Avaliação do plano de consulta.	13 de janeiro de 2014

Alteração	Descrição	Alterado em
Novas tabelas de sistema para consultas	O Amazon Redshift adicionou novas tabelas de sistema que os clientes podem usar para avaliar a execução de consultas para ajuste e solução de problemas. Para obter mais informações, consulte SVL_COMPILE, STL_SCAN, STL_RETURN, STL_SAVE STL_ALERT_EVENT_LOG.	13 de janeiro de 2014
Cursores de único nó	Agora os cursores dão suporte a clusters de único nó. Um cluster de único nó pode ter dois cursores abertos em um dado momento, com um conjunto de resultados máximo de 32 GB. Em um cluster de único nó, recomendamos definir o parâmetro ODBC Cache Size como 1.000. Para obter mais informações, consulte DECLARE.	13 de dezembro de 2013
Estilo de distribuição ALL	A distribuição ALL pode reduzir drasticamente os tempos de execução para determinados tipos de consultas. Quando uma tabela usa o estilo de distribuição ALL, uma cópia da tabela é distribuída para cada nó. Como a tabela é colocada efetivamente com todas as outras tabelas, nenhuma redistribuição é necessária durante a execução da consulta. A distribuição ALL não é apropriada para todas as tabelas porque ela aprimora requisitos de armazenamento e tempo de carregamento. Para obter mais informações, consulte Distribuição de dados para otimização de consultas.	11 de novembro de 2013

Alteração	Descrição	Alterado em
COPY de hosts remotos	Além de carregar tabelas de arquivos de dados no Amazon S3 e nas tabelas do Amazon DynamoDB, o comando COPY pode carregar dados de texto de clusters do Amazon EMR, instâncias do Amazon EC2 e outros hosts remotos usando conexões SSH. O Amazon Redshift usa várias conexões SSH simultâne as para ler e carregar dados em paralelo. Para obter mais informações, consulte Carregamento de dados de hosts remotos.	11 de novembro de 2013
Porcentagem de memória do WLM usada	Você pode equilibrar o workload designando uma porcentagem específica de memória para cada fila na configuração do WLM. Para obter mais informações, consulte Implementar o WLM manual.	11 de novembro de 2013
APPROXIMA TE COUNT(DIS TINCT)	As consultas que usam APPROXIMATE COUNT(DIS TINCT) são executadas muito mais rapidamente, com um baixo relativo de aproximadamente 2%. A função APPROXIMATE COUNT(DISTINCT) usa um algoritmo HyperLogLog. Para obter mais informações, consulte a Função COUNT.	11 de novembro de 2013
Novas funções SQL para recuperar detalhes de consulta recentes	Quatro novas funções SQL recuperam detalhes sobre consultas recentes e comandos COPY. As novas funções facilitam a consulta das tabelas de log do sistema e, em muitos casos, dão detalhes necessári os sem precisar acessar as tabelas de sistema. Para obter mais informações, consulte PG_BACKEND_PID, PG_LAST_COPY_ID, PG_LAST_COPY_COUNT, PG_LAST_QUERY_ID.	1 de novembro de 2013

Alteração	Descrição	Alterado em
Opção MANIFEST para UNLOAD	A opção MANIFEST para o comando UNLOAD complementa a opção MANIFEST do comando COPY. Usar a opção MANIFEST com UNLOAD cria automaticamente um arquivo manifesto que lista explicitamente os arquivos de dados que foram criados no Amazon S3 pela operação de descarregamento. Em seguida, você pode usar o mesmo arquivo manifesto com um comando COPY para carregar os dados. Para obter mais informações, consulte Descarregar dados do Amazon S3 e Exemplos de UNLOAD.	1 de novembro de 2013
Opção MANIFEST para COPY	Você pode usar a opção MANIFEST com o comando COPY para listar explicitamente os arquivos de dados que serão carregados do Amazon S3.	18 de outubro de 2013
Tabelas de sistema para consultas de solução de problemas	Adicionada documentação para tabelas de sistema usadas para solucionar problemas de consultas. Agora a seção Visualizações STL de registro em log contém a documentação das seguintes tabelas de sistema: STL_AGGR, STL_BCAST, STL_DIST, STL_DELETE, STL_HASH, STL_HASHJOIN, STL_INSERT, STL_LIMIT, STL_MERGE, STL_MERGEJOIN, STL_NESTLOOP, STL_PARSE, STL_PROJECT, STL_SCAN, STL_SORT, STL_UNIQUE, STL_WINDOW.	3 de outubro de 2013
Função CONVERT_TIMEZONE	O Função CONVERT_TIMEZONE converte um time stamp de um fuso horário em outro, com a opção de ajuste automático para o horário de verão.	3 de outubro de 2013
Função SPLIT_PART	O Função SPLIT_PART divide uma string no delimitador especificado e retorna a parte na posição especificada.	3 de outubro de 2013

Alteração	Descrição	Alterado em
Tabela do sistema STL_USERLOG	STL_USERLOG registra detalhes das alterações ocorridas quando um usuário do banco de dados é criado, alterado ou excluído.	3 de outubro de 2013
Codificação da coluna LZO e compactação de arquivos LZOP.	A codificação de compactação de coluna LZO oferece uma taxa de compactação extremamente alta e bom desempenho. O COPY do Amazon S3 suporta o carregamento de arquivos compactados usando compactação LZOP.	19 de setembro de 2013
JSON, expressões regulares e cursores	Adicionado suporte para analisar strings JSON, comparar padrões usando expressões regulares e usar cursores para recuperar conjuntos de dados grandes por meio de uma conexão ODBC. Para obter mais informações, consulte Funções JSON, Condições de correspondência de padrões e DECLARE.	10 de setembro de 2013
Opção ACCEPTINVCHAR para COPY	Você pode carregar dados com êxito que contenham caracteres UTF-8 inválidos especificando a opção ACCEPTINVCHAR com o comando COPY.	29 de agosto de 2013
Opção CSV para COPY	Agora comando COPY dá suporte ao carregamento de arquivos de entrada formatados em CSV.	9 de agosto de 2013
CRC32	O Função CRC32 realiza verificações de redundância cíclicas.	9 de agosto de 2013
Curingas do WLM	O gerenciamento de workload (WLM) dá suporte a curingas para adicionar grupos de usuários e consultas a filas. Para obter mais informações, consulte Curingas.	1 de agosto de 2013

Alteração	Descrição	Alterado em
Tempo limite do WLM	Para limitar o valor de tempo em que as consultas em uma determinada fila do WLM têm permissão para serem usadas, você pode definir o valor de tempo limite do WLM para cada fila. Para obter mais informações, consulte Tempo limite do WLM.	1 de agosto de 2013
Novas opções de COPY 'auto' e 'epochsecs'	O comando COPY realiza o reconhecimento automático de formatos de data e a hora. Novos formatos de hora, 'epochsecs' e 'epochmillisecs' permitem que COPY carregue dados em formato epoch.	25 de julho de 2013
Função CONVERT_TIMEZONE	O Função CONVERT_TIMEZONE converte um time stamp de um fuso horário em outro.	25 de julho de 2013
Função FUNC_SHA1	O Função FUNC_SHA1 converte uma string usando o algoritmo SHA1.	15 de julho de 2013
max_execution_time	Para limitar o valor de tempo em que as consultas têm permissão para serem usadas, você pode definir o parâmetro max_execution_time como parte da configuração do WLM. Para obter mais informações, consulte Modificar a configuração do WLM.	22 de julho de 2013
Caracteres UTF-8 de quatro bytes	Agora o tipo de dados VARCHAR dá suporte a caracteres UTF-8 de quatro bytes. Os caracteres UTF-8 de cinco ou mais bytes não são compatíveis. Para obter mais informações, consulte Armazenamento e intervalos.	18 de julho de 2013
SVL_QERROR	A visualização do sistema SVL_QERROR ficou obsoleta.	12 de julho de 2013
Revisado histórico de documento	Agora a página Document History mostra a data em que a documentação foi atualizada.	12 de julho de 2013

Alteração	Descrição	Alterado em
STL_UNLOAD_LOG	STL_UNLOAD_LOG registra os detalhes de uma operação de descarregamento.	5 de julho de 2013
Parâmetro de tamanho da busca JDBC	Para evitar erros de falta de memória no lado do cliente ao recuperar grandes conjuntos de dados usando JDBC, você pode habilitar o cliente para buscar dados em lotes definindo o parâmetro de tamanho da busca JDBC. Para obter mais informaçõ es, consulte Como configurar o parâmetro JDBC para o tamanho da busca.	27 de junho de 2013
Arquivos criptogra fados por UNLOAD	UNLOAD agora suporta o descarregamento de dados da tabela para arquivos criptografados no Amazon S3.	22 de maio de 2013
Credenciais temporárias	Agora COPY e UNLOAD dão suporte ao uso de credenciais temporárias.	11 de abril de 2013
Adicionadas certificações	Esclarecidas e expandidas discussões de Projetar tabelas e carregar dados.	14 de fevereiro de 2013
Adicionadas melhores práticas	Adicionadas Práticas recomendadas do Amazon Redshift para projetar tabelas e Práticas recomenda das do Amazon Redshift para carregamento de dados.	14 de fevereiro de 2013
Esclarecidas restrições de senha	Esclarecidas limitações de senha para CREATE USER e ALTER USER, diversas revisões menores.	14 de fevereiro de 2013
Novo guia	Esta é a primeira versão do Guia do desenvolvedor do Amazon Redshift.	14 de fevereiro de 2013