aws

管理ガイド

Amazon Redshift



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

Amazon Redshift: 管理ガイド

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

Amazon の商標およびトレードドレスは Amazon 以外の製品およびサービスに使用することはできません。また、お客様に誤解を与える可能性がある形式で、または Amazon の信用を損なう形式で使用することもできません。Amazon が所有しない商標はすべてそれぞれの所有者に所属します。所有者は必ずしも Amazon と提携していたり、関連しているわけではありません。また、Amazon 後援を受けているとはかぎりません。

Table of Contents

..... xvii
Amazon Redshift とは 1
Amazon Redshift を初めてご使用になるユーザー向けの情報 1
Amazon Redshift サーバーレス機能の概要 2
Amazon Redshift でプロビジョニングされたクラスターの概要 5
クラスターの管理 5
クラスターのアクセスとセキュリティ 6
クラスターのモニタリング 8
データベース 8
Amazon Redshift サーバーレスと Amazon Redshift のプロビジョニングされたデータウェアハウスの比較 9
プロビジョニングされたクラスター用の Amazon Redshift 管理インターフェイスの使用 34
AWS SDK の操作 35
HTTP リクエストへの署名 36
Amazon Redshift CLI のセットアップ 42
Amazon Redshift Serverless 44
Amazon Redshift Serverless とは 44
Amazon Redshift Serverless コンソール 45
Amazon Redshift サーバーレスを使用する場合の考慮事項 49
Amazon Redshift Serverless 容量を計算する 53
サーバーレスエンドポイント容量に関する考慮事項と制限 55
AI 主導のスケーリングと最適化 57
Amazon Redshift Serverless での請求 63
コンピューティング性能に対する請求 63
オンデマンドのコンピューティング容量に対する請求 63
サーバーレス予約に対する請求 68
ストレージの請求 74
Amazon Redshift Serverless 無料トライアルの利用 75
使用料の請求についての注記 75
接続プーリングを使用した Amazon Redshift Serverless の請求 77
ゼロ ETL を使用した Amazon Redshift Serverless のコスト最適化 78
Amazon Redshift Serverless への接続 79
Amazon Redshift Serverless への接続 79
JDBC ドライバー経由での Amazon Redshift Serverless への接続 79

Data APIを使用した Amazon Redshift Serverless への接続 ... 81
SSL での Amazon Redshift Serverless への接続 ... 81
Amazon Redshift マネージド VPC エンドポイントから Amazon Redshift Serverlessへの接続 ... 84
インターフェイス VPC エンドポイントからの Amazon Redshift Serverless への接続 (AWS PrivateLink) ... 84
別のアカウントの Redshift VPC エンドポイントから Amazon Redshift Serverless への接続 ... 84
その他のリソース ... 89
フェデレーションユーザーに付与するデータベースロールを定義する ... 90
データベースロールの定義 ... 91
フェデレーションユーザーに付与するデータベースロールを定義するユースケース ... 91
その他のリソース ... 93
Amazon Redshift Serverless での Identity and Access Management ... 93
アクセス許可の付与 ... 93
IAM 認証情報の使用開始 ... 95
データベースロールのアクセス許可を持つデータベースオブジェクトへのアクセス ... 96
プロビジョニングされたクラスターを Amazon Redshift Serverless へ移行する ... 98
プロビジョニングされたクラスターのスナップショットを作成する ... 98
ドライバーでの Amazon Redshift Serverless への接続 ... 100
Amazon Redshift Serverless SDK を使用する ... 105
ワークグループと名前空間 ... 106
コンソールを使用したワークグループと名前空間 ... 106
AWS Command Line Interface と Amazon Redshift Serverless API を使用したワークグループと名前空間 ... 107
ワークグループ ... 108
名前空間 ... 114
クエリとワークロードのモニタリング ... 118
クエリモニタリングポリシーを追加する ... 119
ユーザーにクエリモニタリング許可を付与する ... 121
ロールにクエリモニタリング許可を付与する ... 122
使用制限の設定 ... 122
クエリ制限の設定 ... 123
クエリキューの設定 ... 123
ダッシュボードを使用した概要データのチェック ... 128
監査ログ ... 129

CloudWatch でのログイベント 129
CloudWatch メトリクス 130
スナップショットと復旧ポイント 138
AWS Backup の統合 139
スナップショットの作成 140
最終スナップショットの作成 141
スナップショットを共有またはスナップショットのアクセス許可を削除 141
スナップショットをスケジュールする 142
スナップショットの保持期間を更新する 145
スナップショットの削除 145
スナップショットの復元 146
復旧ポイントの変換 147
復旧ポイントの復元 147
別の AWS リージョン へのバックアップのコピー 148
テーブルの復元 149
データ共有 151
考慮事項 151
データ共有表示のアクセス権を付与する 152
AWS Glue Data Catalog への名前空間の登録 152
リソースのタギング 153
Amazon Redshift でプロビジョニングされたクラスター 155
クラスターおよびノード 155
ノードタイプの詳細 157
ノードの数の決定 162
EC2 を使用してクラスターを作成する 163
Amazon Virtual Private Cloud (Amazon VPC) 163
デフォルトのディスク容量アラーム 163
クラスターステータス 164
プロビジョニングされたクラスターを使用する際の考慮事項 167
リージョンとアベイラビリティーゾーンの考慮事項 167
クラスターのメンテナンス 167
クラスターオペレーション 184
クラスターの作成 184
ディスク容量アラームの作成 188
クラスターの表示 188
クラスターの変更 189

クラスターのサイズ変更 ... 190
クラスターの名前変更 ... 208
クラスターのリリースバージョンのアップグレード ... 209
クラスターの一時停止と再開 ... 210
クラスターの再起動 ... 212
クラスターの再配置 ... 213
使用制限の設定 ... 217
クラスターのシャットダウンと削除 ... 219
スナップショットとバックアップ ... 220
AWS Glue Data Catalog にクラスターを登録する ... 247
マルチ AZ デプロイ ... 247
マルチ AZ 配置のセットアップ ... 248
新しいクラスターを作成する際のマルチ AZ の設定 ... 251
スナップショットから復元されたデータウェアハウスのマルチ AZ の設定 ... 254
シングル AZ データウェアハウスのマルチ AZ データウェアハウスへの変換 ... 256
マルチ AZ データウェアハウスのシングル AZ データウェアハウスへの変換 ... 257
マルチ AZ データウェアハウスのサイズ変更 ... 258
マルチ AZ 配置のフェールオーバー ... 259
マルチ AZ データウェアハウスのクエリとロードの表示 ... 261
マルチ AZ 配置でのクエリのモニタリング ... 262
クラスターのクエリを終了する ... 263
クラスターのパフォーマンスをモニタリングする ... 264
パフォーマンスデータ ... 265
パフォーマンスデータの表示 ... 280
クエリ実行の分析 ... 304
アラームを作成する ... 306
実行中クエリの終了 ... 307
CloudWatch コンソールでのパフォーマンスメトリクス ... 308
Query profiler ... 309
クエリとデータベースのモニタリング ... 319
SYS ビューベースのクエリとデータベースのモニタリング ... 326
トラック ... 329
トラックの切り替え ... 330
トラックと復元 ... 330
サーバーレスワークグループの更新 ... 331
バージョンの管理 ... 332

ワークグループバージョンまたはクラスターのバージョンの確認 332
ゼロ ETL 統合 334
考慮事項 337
ターゲットで履歴モードを使用する場合の考慮事項 338
ゼロ ETL 統合ソースが Aurora または Amazon RDS である場合の考慮事項 340
ゼロ ETL 統合ソースが DynamoDB である場合の考慮事項 340
ゼロ ETL 統合のソースが Salesforce、SAP、ServiceNow、Zendesk などのアプリケーションである場合の考慮事項 341
ゼロ ETL 統合ターゲットを復元する際の考慮事項 342
ゼロ ETL 統合ターゲットのサイズ変更時の考慮事項 343
ゼロ ETL 統合の開始方法 344
ターゲット Amazon Redshift データウェアハウスを作成して設定する 344
大文字と小文字の区別を有効にする 345
Amazon Redshift で認可を設定する 347
ゼロ ETL 統合の作成 352
デスティネーションデータベースの作成 364
レプリケートされたデータのクエリ 367
レプリケートしたデータのマテリアライズドビューでのクエリ 367
DynamoDB からのレプリケートされたデータのクエリ 369
ゼロ ETL 統合の表示 370
履歴モード 374
データの共有 376
ゼロ ETL 統合のモニタリング 377
Amazon Redshift システムビューとのゼロ ETL 統合のモニタリング 377
Amazon EventBridge とのゼロ ETL 統合のモニタリング 377
ゼロ ETL 統合のメトリクス 378
DynamoDB のゼロ ETL 統合を変更する 381
DynamoDB のゼロ ETL 統合を削除する 382
サポート対象のリージョン 383
Aurora MySQL 383
Aurora PostgreSQL 385
Amazon DynamoDB 387
Amazon RDS for MySQL 389
エンタープライズアプリケーション 391
ゼロ ETL 統合のトラブルシューティング 393
データベースのクエリ 410

Amazon Redshift に接続する 411
Amazon Redshift クエリエディタ v2 を使用したデータベースのクエリの実行 411
AWS アカウントの設定 412
クエリエディタ v2 を開く 419
Amazon Redshift データベースに接続する 425
Amazon Redshift データベースの閲覧 427
データベースオブジェクトの作成 429
クエリとタブの履歴を表示する 437
Amazon Q 生成 SQL の操作 438
データベースにデータをロードする 451
クエリの作成 461
ノートブック 466
AWS Glue Data Catalog に対するクエリの実行 470
データレイクのクエリ 474
データ共有 476
スケジュールされたクエリ 479
結果の視覚化 490
チームとしての共同作業と共有 495
クエリエディタ v1 を使用してデータベースのクエリを実行する 497
考慮事項 498
SQL クライアントツールを使用して データウェアハウスクラスターに接続する 499
クライアントツールとの接続に関する推奨事項 499
Amazon Redshift での接続の設定 500
メタデータ検出のベストプラクティス 714
接続のセキュリティオプションを設定する 721
クライアントツールおよびコードからの接続 730
認証プロファイルを使用して Amazon Redshift に接続する 779
Amazon Redshift での接続の問題のトラブルシューティング 782
Data API の使用 790
Data API でデータを使用する 791
Data API の呼び出し時の考慮事項 791
データベース認証用の認証情報の選択 794
JDBC データ型のマッピング 795
パラメータを使用した SQL ステートメントの実行 796
冪等性トークンを使用して SQL ステートメントを実行する 798
セッションの再利用による SQL ステートメントの実行 800

結果の取得 ... 801
アクセスの承認 ... 804
信頼できる ID の伝播 ... 814
Data API の呼び出し ... 817
Data API の問題のトラブルシューティング ... 853
Amazon EventBridge を使用したデータ API オペレーションのスケジューリング ... 854
データ API のモニタリング ... 858
Data API での AWS KMS を使用する ... 860
Amazon Sagemaker Unified Studio の使用 ... 863
パラメータグループ ... 865
デフォルトパラメータ値 ... 866
ワークロード管理 ... 868
WLM の動的プロパティと静的プロパティ ... 868
WLM 設定パラメータのプロパティ ... 868
AWS CLI を使用した WLM パラメータの設定 ... 875
パラメータグループを作成する ... 884
パラメータグループを変更する ... 884
クエリモニタリングルールを作成する ... 889
パラメータグループを削除する ... 890
AWS パートナーとの統合 ... 891
AWS パートナーによるデータの読み込み ... 892
リザーブドノード ... 894
リザーブドノードサービス ... 894
リザーブドノードサービス間の価格の比較 ... 895
リザーブドノードの動作 ... 896
リザーブドノードと一括請求 (コンソリデーティッドビリング) ... 898
リザーブドインスタンスを使用するクラスターのノードタイプを変更する前に考慮すべき事項 ... 898
リザーブドノードの例 ... 898
例 1 ... 899
例 2 ... 899
例 3 ... 899
例 4 ... 899
例 5 ... 900
例 6 ... 900
リザーブドノードの購入 ... 900

セキュリティ 903
データ保護 905
データ暗号化 906
データのトークナイゼーション 925
インターネットワークトラフィックのルーティング 926
Identity and access management 927
アイデンティティを使用した認証 927
サービス間の混乱した代理の防止 929
アクセスコントロール 930
アクセス管理の概要 931
アイデンティティベースのポリシー (IAM ポリシー) の使用 938
ネイティブ ID プロバイダー (IdP) フェデレーション 1003
サービスにリンクされたロールの使用 1040
IAM 認証を使用したデータベースユーザー認証情報の生成 1048
AWS のサービスへのアクセスの承認 1106
管理者パスワードの管理 1143
AWS Secrets Manager の統合に必要となるアクセス許可 1144
管理者パスワードシークレットのローテーション 1145
Amazon Redshift で AWS Secrets Manager を使用する場合の考慮事項 1145
シークレットの ARN の取得 1146
データベース接続認証情報のシークレットの作成 1146
ログ作成とモニタリング 1150
データベース監査ログ作成 1150
CloudTrail によるログ記録 1167
S3 Tables とのシステムテーブル統合 1179
コンプライアンス検証 1193
耐障害性 1195
インフラストラクチャセキュリティ 1196
ネットワークの隔離 926
セキュリティグループ 1197
インターフェイス VPC エンドポイント 1197
設定と脆弱性の分析 1205
Identity Center 認証の使用 1207
前提条件 1207
Identity Center 認証の仕組み 1208
GetIdentityCenterAuthToken API オペレーション 1209

プロビジョニングされたクラスターの GetIdentityCenterAuthToken ... 1209
サーバーレスワークグループ用の GetIdentityCenterAuthToken ... 1210
レスポンスの構文 ... 1211
ドライバーの統合 ... 1212
トークンの直接使用 ... 1212
Java コード例 ... 1212
IAM ポリシーの要件 ... 1213
API アクセス許可 ... 1213
IAM ポリシーの例 ... 1213
リージョン別の可用性 ... 1215
トラブルシューティング ... 1215
API エラー ... 1215
接続エラー ... 1216
ネットワークタスク ... 1217
クライアント接続のカスタムドメイン名 ... 1217
ドメイン名の登録 ... 1218
ドメイン名の証明書のリクエスト ... 1219
カスタムドメインの設定 ... 1220
プロビジョニングされたクラスターまたはワークグループに接続する ... 1222
カスタムドメインが割り当てられているクラスターの名前を変更する ... 1223
カスタムドメインの関連付けを説明する ... 1223
カスタムドメインを別の証明書に関連付ける ... 1224
カスタムドメインを削除する ... 1225
RedShift が管理する VPC エンドポイント ... 1226
考慮事項 ... 1227
VPC へのアクセス権の付与 ... 1228
Redshift で管理される VPC エンドポイントの作成 ... 1229
VPC での Redshift リソース ... 1230
VPC でクラスターまたはワークグループを作成する ... 1233
VPC セキュリティグループ ... 1234
クラスターまたはワークグループのセキュリティ設定を構成する ... 1236
Redshift リソースのサブネット ... 1239
VPC とサブネットへのパブリックアクセスをブロックする ... 1242
拡張 VPC ルーティングを使用して Redshift でネットワークトラフィックを制御する ... 1244
VPC エンドポイントを使用したデータベーストラフィックの制御 ... 1246
拡張 VPC ルーティングを有効にする ... 1247

拡張された VPC ルーティングを使用したデータレイクテーブルのクエリ 1249
イベント 1255
クラスターイベント通知サブスクリプション 1255
イベント通知サブスクリプションを作成する 1258
プロビジョニングされたクラスターのイベント通知 1259
Amazon Redshift Serverless イベント通知 1283
イベント通知のゼロ ETL 統合 1296
クォータと制限 1306
Amazon Redshift オブジェクトのクォータ 1306
Amazon Redshift Serverless オブジェクトのクォータ 1315
Amazon Redshift Data API のクォータ 1317
クエリエディタ v2 オブジェクトのクォータ 1319
Amazon Redshift Spectrum のクォータと制限 1321
命名に関する制約 1322
リソースのタグ付け 1326
タグ付け要件 1327
リソースのタグの管理 1327
AWS Backup の統合 1329
考慮事項 1330
制限事項 1331
AWS Backup を Amazon Redshift で使用する 1331
クラスターバージョン 1333
パッチ 205 1334
新機能 1335
パッチ 204 1337
新機能 1337
パッチ 203 1341
新機能 1341
パッチ 202 1344
新機能 1345
パッチ 201 1347
新機能 1347
パッチ 200 1349
新機能 1350
パッチ 199 1352
新機能 1353

パッチ 198 1354
新機能 1355
パッチ 197 1356
新機能 1357
パッチ 196 1359
新機能 1359
パッチ 195 1360
新機能 1360
パッチ 194 1361
新機能 1361
パッチ 193 1362
新機能 1362
パッチ 192 1362
新機能 1363
パッチ 191 1363
新機能 1364
パッチ 190 1364
新機能 1364
パッチ 189 1365
新機能 1365
パッチ 188 1366
新機能 1366
パッチ 187 1367
新機能 1368
パッチ 186 1369
新機能 1370
パッチ 185 1371
新機能 1371
パッチ 184 1372
新機能 1373
パッチ 183 1374
新機能 1374
パッチ 182 1375
新機能 1376
パッチ 181 1377
新機能 1378

パッチ 180 1379
新機能 1380
パッチ 179 1381
新機能 1382
パッチ 178 1383
新機能 1383
パッチ 177 1386
新機能 1387
パッチ 176 1388
新機能 1388
パッチ 175 1390
新機能 1390
パッチ 174 1391
このバージョンの新機能 1391
このバージョンの新機能 1391
このバージョンの新機能 1391
このバージョンの新機能 1391
このバージョンの新機能 1391
このバージョンの新機能 1391
このバージョンの新機能 1391
パッチ 173 1392
このバージョンの新機能 1392
このバージョンの新機能 1392
このバージョンの新機能 1392
このバージョンの新機能 1392
このバージョンの新機能 1392
このバージョンの新機能 1392
このバージョンの新機能 1392
このバージョンの新機能 1392
このバージョンの新機能 1392
このバージョンの新機能 1392
パッチ 172 1394
新機能 1394
パッチ 171 1395
新機能 1395
パッチ 170 1395

新機能 1396
パッチ 169 1396
新機能 1396
パッチ 168 1396
新機能 1397
動作の変更 1398
今後予定されている動作変更 1398
Amazon Redshift は、2026 年 9 月 30 日以降、一時停止中のデータ共有プロデューサークラスターの自動スナップショットを作成 1399
SUPER データ型は、パッチ 206 以降、より大きな個々の文字列をサポート 1399
2026 年 8 月 17 日以降の Amazon Redshift Serverless API における AWS KMS キーアクセス許可の適用 1401
Amazon Redshift では、パッチ 208 以降、ログイン試行が複数回失敗した後にユーザーロックアウトを適用します 1402
2026 年 6 月 8 日に、Amazon Redshift Serverless および Amazon Redshift RG インスタンスの手動スナップショットの請求モデルを強化 1402
Lake Formation テーブルの Iceberg DELETE には、パッチ 202 で始まる DELETE アクセス許可が必要です 1403
Amazon Redshift Serverless は、パッチ 202 以降のスナップショット復元でゼロ ETL および S3 イベント統合を保持 1404
2026 年 12 月 31 日の Amazon Redshift ODBC 1.x ドライバーのサポート終了 1404
スカラー Python UDFは 2026 年 6 月 30 日にサポートが終了します 1405
2026 年 2 月 27 日以降のマテリアライズドビュー (MV) の自動 REFRESH の動作変更 1406
Amazon Redshift は、2026 年 2 月 16 日以降、データ共有を通じてコンシューマー情報にアクセスする関数をサポートしません 1406
2026 年 10 月 31 日以降に有効となる Transport Layer Security (TLS) の最小バージョンに関する変更 1406
Amazon Redshift は、2025 年 10 月 30 日以降、新しいスカラー Python UDF の作成をサポートしません 1407
最近の動作の変更 1408
2025 年 8 月 26 日以降、Amazon Redshift は最新の IANA タイムゾーンデータベースを使用します 1408
2025 年 8 月 15 日以降に有効になる Amazon Redshift Serverless RPU の変更 1409
2025 年 8 月 10 日以降に有効となるデータベース監査ログ記録に関する変更 1409
2025 年 6 月 27 日以降に有効になるサーバーレスワークグループの仮想プライベートクラウドエンドポイントの変更 1410

2025 年 5 月 2 日以降に有効になる、クエリモニタリングの変更 1410
2025 年 1 月 10 日以降に有効になるセキュリティ変更 1411
コードの例 1414
基本 1415
Hello Amazon Redshift 1415
基本を学ぶ 1420
アクション 1487
シナリオ 1547
Amazon Redshift データ追跡用のウェブアプリケーションの作成 1547
サーバーレスデータウェアハウスの使用開始 1548
プロビジョニングされたデータウェアハウスクラスターの使用開始 1566
ドキュメント履歴 1577

Amazon Redshift は、2026 年 6 月 30 日以降、Python UDF の使用をサポートしなくなります。これは段階的に実施されます。Python のサポート終了と移行オプションの詳細については、2025 年 6 月 30 日に公開されたブログ記事を参照してください。

Amazon Redshift とは

「Amazon Redshift 管理ガイド」をご覧いただき、ありがとうございます。Amazon Redshift は、クラウド内でのフルマネージド型、ペタバイトスケールのデータウェアハウスサービスです。Amazon Redshift Serverless を使用すると、プロビジョニングされたデータウェアハウスをすべて設定しなくても、データにアクセスして分析することができます。リソースは自動的にプロビジョニングされて、データウェアハウス容量はインテリジェントにスケーリングされ、要求が厳しく、予測不可能なワークロードであっても高速なパフォーマンスを実現します。データウェアハウスがアイドル状態のときには課金されず、使用した分のみ支払います。Amazon Redshift クエリエディタ v2 またはお好みのビジネスインテリジェンス (BI) ツールで、データをロードしてクエリを直ちに開始することができます。使いやすい管理不要の環境で、最高のコストパフォーマンスと使い慣れた SQL 機能をお楽しみください。

データセットのサイズに関係なく、Amazon Redshift は現在使用しているのと同じ SQL ベースのツールとビジネスインテリジェンスアプリケーションを使用して、高速なクエリパフォーマンスを提供します。

Amazon Redshift を初めてご使用になるユーザー向けの情報

Amazon Redshift を初めて使用する方には、以下のセクションを初めに読むことをお勧めします。

- 主なサービスと料金設定 — この製品詳細ページには、Amazon Redshift の価値提案、主なサービス、料金設定が掲載されています。
- Amazon Redshift Serverless データウェアハウスの使用を開始 — このトピックでは、サーバーレスデータウェアハウスのセットアップ、リソースの作成、サンプルデータのクエリのプロセスを順を追って説明します。
- Amazon Redshift データベースデベロッパーガイド – このガイドでは、データベースデベロッパーを対象に、データウェアハウスを構成するデータベースの設計、構築、クエリ、および保守の方法について説明します。

Amazon Redshift リソースを手動で管理したい場合は、データクエリのニーズに合わせてプロビジョニングされたクラスターを作成することができます。詳細については、「Amazon Redshift クラスター」を参照してください。

アプリケーションデベロッパーは、Amazon Redshift API または AWS Software Development Kit (SDK) のライブラリを使用して、クラスターをプログラムで管理できます。Amazon Redshift API を

使用する場合、署名することですべての HTTP または HTTPS リクエストを認証する必要があります。リクエストの署名に関する詳細については、「HTTP リクエストへの署名」を参照してください。

API、CLI、SDK の詳細については、次のリンクを参照してください。

- Amazon Redshift Serverless API リファレンス
- Amazon Redshift API リファレンス
- Amazon Redshift Data API リファレンス
- AWS CLI コマンドリファレンス
- SDK に関しては、「Amazon Web Services のツール」を参照します。

Amazon Redshift サーバーレス機能の概要

Amazon Redshift のプロビジョニングされたデータウェアハウスでサポートされている機能のほとんどは、Amazon Redshift サーバーレスでもサポートされています。次にその主な機能の一部を示します。

機能	説明
スナップショット	Amazon Redshift サーバーレスのスナップショットまたはプロビジョニングされたデータウェアハウスは、Amazon Redshift サーバーレスに復元できます。詳細については、「スナップショットと復旧ポイント」を参照してください。
復旧ポイント	Amazon Redshift サーバーレスは 30 分ごとに自動的に復旧ポイントを作成します。これらの復旧ポイントは 24 時間保持されます。この復旧ポイントを使用することで、誤って書き込んだり削除した後でも復元できます。復旧ポイントにより、Amazon Redshift Serverless のデータベース内のすべてのデータを、そのポイントの時刻の状態に復元できます。復旧時刻の状態を長期間保持する必要がある場合は、その復旧ポイントからスナップショットを作成することもできます。詳細については、「スナップショットと復旧ポイント」を参照してください。
基本 RPU 容量	基本の容量は、Redshift 処理ユニット (RPU) で設定できます。1 つの RPU が 16 GB のメモリを提供します。この設定により、使用中のリソースとワークロードのコストのバランスを制御できます。使用可能なリソースを増やしてクエリパフォーマンスを向上させるには、この値を大きくします。使用を制限するには、値を小さくします。デフォルト値は 128 RPU です。1 日あたりに使用する RPU

機能	説明
	などの使用制限を設定して、コストを管理することもできます。詳細については、「Amazon Redshift Serverless での請求」を参照してください。
データ共有の使用制限	プロデューサーリージョンからコンシューマーリージョンに転送されるデータの量は、コンソールまたは API を使用して制限できます。これらのデータ転送の料金は AWS リージョン によって異なり、テラバイト単位で測定されます。データ共有の詳細については、「Amazon Redshift データベースデベロッパーガイド」の「コンソールを使用したデータ共有の開始方法」を参照してください。
ユーザー定義関数 (UDF)	Amazon Redshift サーバーレスでは、ユーザー定義関数 (UDF) を実行できます。詳細については、Amazon Redshift データベース開発者ガイドの「ユーザー定義関数の作成」を参照してください。
ストアドプロシージャ	Amazon Redshift サーバーレスでは、ストアドプロシージャを実行できます。詳細については、Amazon Redshift データベース開発者ガイドの「Amazon Redshift のストアドプロシージャの作成」を参照してください。
マテリアライズドビュー	Amazon Redshift サーバーレスでは、マテリアライズドビューを作成できます。詳細については、Amazon Redshift データベース開発者ガイドの「Amazon Redshift でのマテリアライズドビューの作成」を参照してください。
空間関数	Amazon Redshift サーバーレスでは、空間関数を実行できます。詳細については、Amazon Redshift データベース開発者ガイドの「Amazon Redshift での空間データのクエリ」を参照してください。
横串検索	クエリを実行して、Amazon Redshift Serverless から Aurora DB データベースと Amazon RDS データベースのデータを結合できます。詳細については、Amazon Redshift データベース開発者ガイドの「Amazon Redshift での横串検索を使用したデータのクエリの実行」を参照してください。
データレイククエリ	クエリを実行して、Amazon S3 データレイクのデータを Amazon Redshift サーバーレスと結合できます。詳細については、「Amazon Redshift 管理ガイド」の「データレイクのクエリ」を参照してください。
HyperLogLog	Amazon Redshift サーバーレスでは、HyperLogLog 関数を実行できます。詳細については、Amazon Redshift データベース開発者ガイドの「Amazon Redshift での HyperLogLog スケッチの使用」を参照してください。

機能	説明
データベース間でのデータのクエリ	Amazon Redshift サーバーレスのデータベース間でデータをクエリできます。詳細については、Amazon Redshift データベース開発者ガイドの「データベース間でのデータのクエリ」を参照してください。
データ共有	Amazon Redshift サーバーレスを使用して、プロビジョニングされたデータウェアハウスのデータ共有にアクセスできます。詳細については、Amazon Redshift データベース開発者ガイドの「Amazon Redshift でのクラスター間のデータの共有」を参照してください。
半構造化データのクエリ	SUPER データ型の半構造化データを、Amazon Redshift Serverless に取り込んで保存できます。詳細については、Amazon Redshift データベース開発者ガイドの「Amazon Redshift 半構造化データの取り込みとクエリ」を参照してください。
リソースのタグ付け	AWS CLI または Amazon Redshift サーバーレス API を使用して、リソースに関連するメタデータをリソースにタグ付けできます。詳細については、「リソースにタグを付ける」を参照してください。
機械学習	Amazon Redshift サーバーレスで Amazon Redshift 機械学習を使用できます。詳細については、Amazon Redshift データベース開発者ガイドの「Amazon Redshift での機械学習の使用」を参照してください。
SQL コマンドと関数	Amazon Redshift サーバーレスでは、いくつかの例外 (REBOOT_CLUSTER など) を除き、Amazon Redshift SQL コマンドと関数を使用できます。詳細については、Amazon Redshift データベース開発者ガイドの「SQL リファレンス」を参照してください。
CloudFormation リソース	CloudFormation テンプレートを使用して、Amazon Redshift サーバーレスのリソースをデプロイおよび更新できます。この統合により、リソースの管理に費やす時間を短縮し、アプリケーションに集中することができます。Amazon Redshift Serverless の CloudFormation リソースの詳細については、「Amazon Redshift Serverless resource type reference」(Amazon Redshift Serverless リソースタイプのリファレンス) を参照してください。

機能	説明
CloudTrail リソース	Amazon Redshift Serverless は AWS CloudTrail と統合され、Amazon Redshift Serverless で実行されたアクションの記録を提供します。CloudTrail のすべての API コールをイベントとして Amazon Redshift Serverless にキャプチャします。詳細については、「Amazon Redshift サーバーレスの　CloudTrail」を参照してください。

Amazon Redshift でプロビジョニングされたクラスターの概要

Amazon Redshift サービスは、データウェアハウスを設定、操作、および拡張するためのすべての作業を管理します。これらのタスクには、クラスターの容量のプロビジョニング、モニタリング、およびバックアップのほか、Amazon Redshift エンジンへのパッチおよびアップグレードの適用が含まれます。

次の動画は、Amazon Redshift クエリエディタ v2 を使用してクラスターを作成する方法とデータをクエリする方法を説明しています。

クラスターの管理

Amazon Redshift クラスターは、リーダーノードと 1 つ以上のコンピューティングノードから構成される一連のノードです。必要になるコンピューティングノードのタイプと数は、データのサイズ、実行するクエリの数、必要とされるクエリランタイムパフォーマンスに応じて異なります。

クラスターの作成と管理

データウェアハウスのニーズに応じて、小規模な単一ノードのクラスターから始め、要件が変わったら、より規模の大きい複数ノードのクラスターに簡単に拡張できます。サービスを中断することなく、コンピューティングノードをクラスターに追加または削除できます。詳細については、「Amazon Redshift でプロビジョニングされたクラスター」を参照してください。

コンピューティングノードの予約

クラスターを 1 年以上連続して実行する場合、1 年間または 3 年間コンピューティングノードを予約することでお金を節約できます。コンピューティングノードを予約すれば、要求に応じてコンピューティングノードをプロビジョニングするときに時間単位で支払う場合に比べて大幅な節約になります。詳細については、「リザーブドノード」を参照してください。

クラススナップショットの作成

スナップショットはクラスターのポイントインタイムバックアップです。スナップショットには、自動と手動の 2 つのタイプがあります。Amazon Redshift は、暗号化された Secure Sockets Layer (SSL) 接続を使用して、これらのスナップショットを Amazon Simple Storage Service (Amazon S3) の内部に保存できます。スナップショットから復元する場合は、Amazon Redshift で新しいクラスターを作成し、指定したスナップショットからデータをインポートします。 スナップショットの詳細については、「Amazon Redshift スナップショットとバックアップ」を参照してください。

クラスターのアクセスとセキュリティ

Amazon Redshift には、クラスターのアクセスとセキュリティに関連する複数の機能があります。これらの機能は、クラスターへのアクセスのコントロール、接続ルールの定義、データおよび接続の暗号化に役立ちます。これらの機能は、Amazon Redshift のデータベースアクセスおよびセキュリティに関連する機能に追加されています。データベースセキュリティの詳細については、Amazon Redshift データベースデベロッパーガイドのデータベースセキュリティの管理を参照してください。

AWS アカウントと IAM 認証情報

デフォルトでは Amazon Redshift クラスターは、クラスターを作成する AWS アカウントにのみアクセスできます。クラスターは誰もアクセスできないようにロックされています。AWS アカウント内では、AWS Identity and Access Management (IAM) サービスを使用してユーザーアカウントを作成し、それらのアカウントがクラスター操作を制御するためのアクセス許可を管理します。詳細については、「Amazon Redshift のセキュリティ」を参照してください。IAM ロールのガイダンスやベストプラクティスを含む、IAM ID の管理の詳細については、「Amazon Redshift での Identity and Access Management」を参照してください。

セキュリティグループ

デフォルトでは、作成したクラスターはすべてのユーザーに対して閉じられます。IAM 認証情報は、Amazon Redshift API 関連リソース (Amazon Redshift コンソール、コマンドラインインターフェイス (CLI)、API および SDK へのアクセスのみコントロールします。JDBC または ODBC 経由で SQL クライアントツールからクラスターへのアクセスを有効にするには、セキュリティグループを使用します。

- Amazon Redshift クラスターに EC2-VPC プラットフォームを使用している場合、VPC セキュリティグループを使用する必要があります。クラスターは EC2-VPC プラットフォームで起動することをお勧めします。

クラスターを EC2-Classic で起動した後に VPC に移行することはできません。ただし、Amazon Redshift コンソールを使用して EC2-Classic スナップショットを EC2-VPC クラスターに復元できます。詳細については、「スナップショットからのクラスターの復元」を参照してください。

- Amazon Redshift クラスターに EC2-Classic プラットフォームを使用している場合、Amazon Redshift セキュリティグループを使用する必要があります。

いずれの場合でも、CIDR/IP アドレスの特定の範囲に明示的なインバウンドアクセスを付与するためにセキュリティグループにルールを追加するか、SQL クライアントが Amazon EC2 インスタンスで実行されている場合には、Amazon Elastic Compute Cloud (Amazon EC2) セキュリティグループにルールを追加します。詳細については、「Amazon Redshift セキュリティグループ」を参照してください。

インバウンドアクセスルールに加えて、クラスター内でデータベース認証を行うための認証情報を提供するために、データベースのユーザーを作成します。詳細については、このトピックの「データベース」を参照してください。

暗号化

クラスターをプロビジョニングする際には、オプションで、セキュリティ強化のためにクラスター暗号化オプションを選択できます。暗号化を有効にすると、Amazon Redshift は、暗号化されたフォーマットでユーザーが作成したテーブルにすべてのデータを保存します。AWS Key Management Service (AWS KMS) を使用して、Amazon Redshift の暗号化キーを管理できます。

暗号化は、クラスターの変更不可能なプロパティです。暗号化されたクラスターから、暗号化されていないクラスターに切り替える唯一の方法は、データをアンロードして新しいクラスターに再ロードすることです。暗号化は、クラスターとすべてのバックアップに適用されます。暗号化されたスナップショットからクラスターを復元すると、新しいクラスターも暗号化されます。

暗号化、キー、およびハードウェアセキュリティモジュールに関する詳細については、「Amazon Redshift データベース暗号化」を参照してください。

SSL 接続

SQL クライアントとクラスター間の接続を暗号化するには、Secure Sockets Layer (SSL) 暗号化を使用できます。詳細については、「接続のセキュリティオプションを設定する」を参照してください。

クラスターのモニタリング

Amazon Redshift でのモニタリングに関連するさまざまな機能があります。データベース監査ログ記録を使って、アクティビティログを生成して、イベントと通知サブスクリプションの設定を行い、関心のある情報をトラッキングすることができます。Amazon Redshift と Amazon CloudWatch でメトリクスを使用し、クラスターとデータベースのヘルスとパフォーマンスを把握する方法について説明します。

データベース監査ログ作成

認証の試み、接続、切断、データベースユーザー定義の変更、データベースで実行されるクエリに関する情報を追跡するために、データベース監査ログ作成機能を使用できます。この情報は、Amazon Redshift 内でセキュリティおよびトラブルシューティングの目的で役立ちます。このログは Amazon S3 バケットに格納されます。詳細については、「データベース監査ログ作成」を参照してください。

イベントと通知

Amazon Redshift ではイベントが追跡され、AWS アカウントでは数週間、そのイベントに関する情報が保持されます。Amazon Redshift はイベントごとに、イベントが発生した日付、説明、イベントのソース (クラスター、パラメータグループ、またはスナップショット)、ソース ID などの情報を報告します。一連のイベントフィルターを指定する、Amazon Redshift イベント通知サブスクリプションを作成できます。フィルター条件に一致するイベントが発生すると、Amazon Redshift は Amazon Simple Notification Service を使用して、イベントが発生したことを通知します。イベントのタイプと通知の詳細については、「Amazon Redshift のイベント」を参照してください。

パフォーマンス

Amazon Redshift では、クラスターとデータベースの状態やパフォーマンスを追跡できるようにするため、パフォーマンスメトリクスとデータが提供されます。Amazon Redshift は、Amazon CloudWatch メトリクスを使用して、CPU 使用率、レイテンシー、スループットなど、クラスターの物理的な側面をモニタリングするのに役立ちます。Amazon Redshift は、クラスター内のデータベースアクティビティを監視するのに役立つクエリおよびロードパフォーマンスデータも提供します。パフォーマンスメトリクスとモニタリングの詳細については、「Amazon Redshift クラスターパフォーマンスのモニタリング」を参照してください。

データベース

Amazon Redshift は、クラスターをプロビジョニングするときにデータベースを 1 つ作成します。これは、データをロードし、データのクエリを実行するために使用するデータベースです。必要

に応じて SQL コマンドを実行することで、追加のデータベースを作成できます。詳細については、Amazon Redshift データベースデベロッパーガイドのステップ 1: データベースを作成するを参照してください。

クラスターをプロビジョニングする際に、クラスター内に作成されるすべてデータベースにアクセスが可能な管理者ユーザーを指定します。この管理者ユーザーは、最初にデータベースにアクセスできる唯一のユーザーであるスーパーユーザーです。ただし、このユーザーは追加のスーパーユーザーとユーザーを作成できます。詳細については、Amazon Redshift データベースデベロッパーガイドのスーパーユーザーおよびユーザーを参照してください。

Amazon Redshift はパラメータグループを使用して、日付のプレゼンテーションスタイルや小数点の精度など、クラスター内のすべてのデータベースの動作を定義します。クラスターをプロビジョニングする際にパラメータグループを指定しない場合、Amazon Redshift は、デフォルトのパラメータグループをクラスターに関連付けます。詳細については、「Amazon Redshift パラメータグループを作成します。」を参照してください。

Amazon Redshift のデータベースの詳細については、Amazon Redshift データベースデベロッパーガイドを参照してください。

Amazon Redshift サーバーレスと Amazon Redshift のプロビジョニングされたデータウェアハウスの比較

Amazon Redshift サーバーレスの一部の概念と機能は、Amazon Redshift のプロビジョニングされたデータウェアハウスの対応する機能とは異なります。例えば、1 つの明らかな違いとして、Amazon Redshift サーバーレスには、クラスターまたはノードという概念がありません。次の表では、Amazon Redshift サーバーレスの機能と動作を示し、プロビジョニングされたデータウェアハウスの同等の機能との違いについて説明します。

機能	説明	サーバーレス	プロビジョンド
ワークグループと名前空間	Amazon Redshift サーバーレスでワークロードを分離し、	名前空間は、データベースオブジェクトとユーザーのコレク	プロビジョニングされたクラスターは、ユーザーが直接管理するコンピューティングノードとリーダーノードのコレクションです。詳細については、「Amazon Redshift でプロビジョニングされたクラスター」を参照してください。

機能	説明	サーバーレス	プロビジョンド
	さまざまなリソースを管理するには、名前空間とワークグループを作成してストレージリソースとコンピューティングリソースを別々に管理できます。	ションです。ワークグループは、コンピューティングリソースのコレクションです。詳細については、「Amazon Redshift Serverless」を参照し、Amazon Redshift サーバーレスの設計を理解してください。	

機能	説明	サーバーレス	プロビジョンド
ノードタイプ	Amazon Redshift サーバーレスでは、Amazon Redshift のプロビジョニングされたクラスターとは異なり、ノードタイプを選択したり、ノード数を指定したりしません。	Amazon Redshift Serverless は、自動的に容量をプロビジョニングおよび管理します。必要に応じて、ベースデータウェアハウスの容量を指定して、ワークロードに適した価格とパフォーマンスのバランスを選択できます。また、最大 RPU 時間を指定してコスト管理を設定し、コストが予測可能になるようにすることもできます。詳細については、	コストとパフォーマンスの仕様を満たすノードタイプを使用してクラスターを構築します。詳細については、「Amazon Redshift でプロビジョニングされたクラスター」を参照してください。

機能	説明	サーバーレス	プロビジョンド
		「Amazon Redshift Serverless 容量を計算する」を参照してください。	
ワークロード管理と同時実行スケーリング	Amazon Redshift は負荷の高い時間帯に応じてスケールできます。Amazon Redshift サーバーレスも、断続的に発生する負荷の高い時間帯に応じてスケールできます。	Amazon Redshift Serverless は、リソースを自動的かつ効率的に管理し、ワークロードに基づいてコスト管理のしきい値内でスケーリングします。詳細については、「コンピューティング性能に対する請求」を参照してください。	プロビジョニングされたデータウェアハウスを使用すると、クラスターでの同時実行スケーリングを有効にして、負荷の高い時間帯に対応できます。詳細については、「同時実行スケーリング」を参照してください。

機能	説明	サーバーレス	プロビジョンド
ポート –	接続に使用するポート番号。	Amazon Redshift Serverless では、5431-5455 または 8191-8215 のポート範囲から別のポートに変更できます。詳細については、「Amazon Redshift Serverless への接続」を参照してください。	プロビジョニングされたデータウェアハウスでは、任意のポートを選択して接続できます。

機能	説明	サーバーレス	プロビジョンド
サイズ変更	コンピュートリソースを追加または削除して、ワークロードに適したパフォーマンスを実現します。	Amazon Redshift Serverless では、サイズ変更が適用されません。ただし、基本データウェアハウスの RPU 容量は、価格とパフォーマンスの要件に基づいて変更できます。詳細については、「Amazon Redshift Serverless 容量を計算する」を参照してください。	プロビジョニングされたクラスターでは、クラスターをサイズ変更してノードを追加または削除します。詳細については、「Amazon Redshift でのクラスター管理の概要」を参照してください。

機能	説明	サーバーレス	プロビジョンド
一時停止と再開	実行するワークロードがないときは、プロビジョニングされたクラスターを一時停止してコストを節約できます。	Amazon Redshift サーバーレスでは、クエリの実行時にのみ料金が発生するため、一時停止や再開は不要です。詳細については、「コンピューティング性能に対する請求」を参照してください。	さまざまな時間帯のワークロードの評価に応じて、クラスターの一時停止および再開を手動で行います。詳細については、「Amazon Redshift でのクラスター管理の概要」を参照してください。

機能	説明	サーバーレス	プロビジョンド
外部データのクエリ	Amazon S3 バケット内のデータを、JSON、Parquet、または Iceberg などのさまざまな形式でクエリできます。	Amazon Redshift Serverless には、ワークグループのコンピューティングリソースで実行される統合データレイククエリエンジンが含まれており、AWS リージョン内または異なる AWS リージョン間での Amazon S3 データのクエリをサポートします。データレイククエリのパフォーマンスを調整するには、ワークグループの基本 RPU 容	RA3 および DC2 でプロビジョニングされたクラスターでは、データレイククエリは、クラスターから独立した専有サーバーを使用して Redshift Spectrum で実行されます。Redshift Spectrum では、Amazon S3 バケットがクラスターと同じ AWS リージョンにある必要があります。詳細については、「Amazon Redshift Spectrum を使用した外部データのクエリ」を参照してください。 RG プロビジョニングクラスターには、統合されたデータレイククエリエンジンが含まれています。データレイククエリは、Redshift Serverless と同様にクラスター独自のコンピューティングリソースで直接実行され、同じ AWS リージョンまたは別の AWS リージョンの Amazon S3 データのクエリをサポートします。詳細については、「データレイクのクエリ」を参照してください。

機能	説明	サーバーレス	プロビジョンド
		量を調整します。	
コンピューティングリソースの請求	Amazon Redshift と Amazon Redshift サーバーレスでの料金の発生方法の違い。	Amazon Redshift Serverless では、実行するワークロードに対して秒単位の RPU 時間 (60 秒の最低料金) で課金されます。これには、Amazon S3 のオープンファイル形式のデータにアクセスするクエリが含まれます。詳細については、「コンピューティング性能に対する請求」を参照してください。	プロビジョニングされたクラスターでは、クラスターが一時停止されていない場合、1 秒ごとに料金が発生します。RG クラスターの場合、データレイククエリは、個別の Spectrum 料金なしでノード時間のコンピューティング料金に含まれます。RA3 クラスターと DC2 クラスターの場合、データレイククエリには追加の Redshift Spectrum 料金が発生します。

機能	説明	サーバーレス	プロビジョンド
メンテナンスウィンドウ	サーバーメンテナンスの仕組み	Amazon Redshift Serverless では、メンテナンスウィンドウがありません。更新はシームレスに処理されます。詳細については、「Amazon Redshift サーバーレスとは」を参照してください。	プロビジョニングされたクラスターでは、パッチ適用が発生するときに、メンテナンスウィンドウを指定します。(通常、使用量が少ない場合は定期的な時間を選択します)。
暗号化:	データベースの暗号化を有効にできます。	Amazon Redshift Serverless は、常に AWS KMS と AWS マネージドキーまたはカスタマーマネージドキーで暗号化されます。	プロビジョニングされたデータウェアハウスのデータは、AWS KMS (AWS マネージドキーまたはカスタマーマネージドキー) で暗号化するか、暗号化しないままにすることができます。「Amazon Redshift データベース暗号化」を参照してください。

機能	説明	サーバーレス	プロビジョンド
ストレージ請求	ストレージの請求の仕組み。	Amazon Redshift サーバーレスの場合、このレートは、1 か月あたりの GB に基づいて計算されます。「コンピューティング性能に対する請求」を参照してください。	ストレージは、RG または RA3 ノードを持つプロビジョニングされたクラスターのコンピューティングリソースとは別に課金されます。

機能	説明	サーバーレス	プロビジョンド
ユーザー管理	ユーザーの管理方法。	Amazon Redshift Serverless の場合、ユーザーは IAM ユーザーまたは Redshift ユーザーです。詳細については、「Amazon Redshift Serverless での Identity and Access Management」を参照してください。 IAM ロールのベストプラクティスを含む IAM ID の管理の詳細については、「Amazon Redshift での Identity and	プロビジョニングされたデータウェアハウスの場合、ユーザーは IAM ユーザーまたは Redshift ユーザーです。詳細については、「Amazon Redshift データベース開発者ガイド」の「データベースセキュリティの管理」を参照してください。 IAM ロールのベストプラクティスを含む IAM ID の管理の詳細については、「Amazon Redshift での Identity and Access Management」を参照してください。

機能	説明	サーバーレス	プロビジョンド
		Access Management」を参照してください。	

機能	説明	サーバーレス	プロビジョンド
JDBC および ODBC のツールと互換性	クライアント接続の仕組み。	Amazon Redshift Serverless は、JDBC または ODBC 準拠のツールやクライアントアプリケーションと互換性があります。ドライバーの詳細については、「Amazon Redshift 管理ガイド」の「接続の設定」を参照してください。Amazon Redshift Serverless への接続の詳細については、「Redshift Serverless への接続」を参照してください。	Amazon Redshift プロビジョンドは、JDBC または ODBC 準拠のツールやクライアントアプリケーションと互換性があります。ドライバーの詳細については、「Amazon Redshift 管理ガイド」の「接続の設定」を参照してください。クラスターへの接続については、「SQL クライアントツールを使用して Amazon Redshift データウェアハウスクラスターに接続する」を参照してください。

機能	説明	サーバーレス	プロビジョンド
サインイン時の認証情報の要件	認証情報の処理方法。	Amazon Redshift Serverless では、すべてのインスタンスで認証情報を入力する必要はありません。詳細については、「Amazon Redshift Serverless への接続」を参照してください。	Amazon Redshift にアクセスするには、IAM ロールに関連付けられたユーザーからのサインイン認証情報が必要です。IAM ロールには、プロビジョニングされたデータウェアハウスへのアクセス許可がアタッチされています。認証されると、ユーザーはデータベース、Redshift コンソール、クエリエディタ v2 に直接接続できます。

機能	説明	サーバーレス	プロビジョンド
Data API	ウェブサービスやその他のアプリケーションからデータにアクセスできます。	Amazon Redshift サーバーレスは、Amazon Redshift Data API をサポートしています。Amazon Redshift サーバーレスでは、`cluster-identity` パラメータではなく `workgroup-name` パラメータを使用します。Data API を呼び出す方法の詳細については、「Amazon Redshift Data API の使用」を参照してください。	Amazon Redshift プロビジョンドは、Amazon Redshift Data API をサポートしています。Amazon Redshift クラスターでは、`workgroup-name` パラメータではなく、`cluster-identity` パラメータを使用します。Data API を呼び出す方法の詳細については、「Amazon Redshift Data API の使用」を参照してください。

機能	説明	サーバーレス	プロビジョンド
スナップショット	ポイントインタイムリカバリを提供します。	Amazon Redshift Serverless では、スナップショットと復旧ポイントをサポートします。名前空間のスナップショットと復旧ポイントの詳細については、「スナップショットと復旧ポイント」を参照してください。	プロビジョニングされたクラスターはスナップショットをサポートします。詳細については、「コンソールを使用したスナップショットの管理」を参照してください。

機能	説明	サーバーレス	プロビジョンド
データ共有	同じアカウントまたは異なるアカウントのデータベース間でデータを共有できるようにします。	Amazon Redshift Serverless は、プロビジョニングされたデータウェアハウスがサポートするすべてのデータ共有機能をサポートします。また、Amazon Redshift Serverless とプロビジョニングされたデータウェアハウス、ツール、またはクライアントアプリケーション間のデータ共有もサポートします。	プロビジョニングされたクラスターは、クロスデータベース、クロスアカウント、クロスリージョン、AWS Data Exchange データ共有をサポートします。詳細については、「Amazon Redshift でのクラスター間のデータの共有」を参照してください。

機能	説明	サーバーレス	プロビジョンド
トラック	ソフトウェアアップデートのスケジュールを提供します。	Amazon Redshift Serverless にはトラックの概念はありません。バージョンと更新は、サービスが処理します。Amazon Redshift サーバーレスの設計の詳細については、「スナップショットと復旧ポイント」を参照してください。	プロビジョニングされたクラスターは、現在のトラックと後続トラックの切り替えをサポートします。

機能	説明	サーバーレス	プロビジョンド
システムテーブルとビュー	リソースとシステムメタデータをモニタリングする方法を提供します。	Amazon Redshift Serverless では、新しいシステムテーブルとビューをサポートします。システムテーブルの詳細については、「Amazon Redshift Serverless でのクエリとワークロードのモニタリング」を参照してください。プロビジョニングされた古いシステムテーブルとビューから新しいビューにクエリを移行する方法については、「SYS モ	プロビジョニングされたデータウェアハウスでは、モニタリングやシステムメタデータを必要とする他のタスクのために、既存のシステムテーブルとビューのセットをサポートします。

機能	説明	サーバーレス	プロビジョンド
		ニタリングビューへの移行」を参照してください。	
パラメータグループ	これは、クラスター内で作成したすべてのデータベースに適用されるパラメータのグループです。これらのパラメータは、クエリのタイムアウトやデータスタイルなどのデータベース設定を定義します。	Amazon Redshift Serverless には、パラメータグループの概念がありません。	プロビジョニングされたデータウェアハウスは、パラメータグループをサポートします。プロビジョニングされたクラスターのパラメータグループについての詳細は、「Amazon Redshift パラメータグループを作成します。」を参照してください。

機能	説明	サーバーレス	プロビジョンド
クエリのモニタリング	実行したクエリを時間ベースで表示します。	Amazon Redshift Serverless のクエリモニタリングでは、システムテーブルを使用するには、ユーザーによってデータベースに接続する必要があります。そのため、クエリモニタリングとシステムテーブルは同期しています。Amazon Redshift Serverless のシステムテーブルのクエリでは、IAM ユーザーにマッピングされたデータベー	プロビジョニングされたクラスターでのクエリのモニタリングは、システムテーブルのすべてのデータを表示するわけではありません。

機能	説明	サーバーレス	プロビジョンド
		スユーザーを使用して、クエリモニタリングを使用します。クエリのモニタリングの詳細については、「Amazon Redshift サーバーレスでのクエリとワークロードのモニタリング」を参照してください。	

機能	説明	サーバーレス	プロビジョンド
監査ログ	データベースの接続とユーザーアクティビティに関する情報を提供します。	Amazon Redshift サーバーレスでは、CloudWatch が監査ログの送信先になります。Amazon S3 ベースの監査ログ配信は、Amazon Redshift Serverless ではサポートされていません。詳細については、「Amazon Redshift サーバーレスの監査ログ」を参照してください。	プロビジョニングされたクラスターでは、Amazon S3 ベースの監査ログ配信が標準となっています。今回、CloudWatch への監査ログの配信が拡張されて、プロビジョニングされたデータウェアハウスをカバーするようになりました。

機能	説明	サーバーレス	プロビジョンド
イベント通知	Amazon EventBridge は、アプリケーションをさまざまなソースからのデータに接続するために使用できるサーバーレスのイベントバスサービスです。	Amazon Redshift Serverless は Amazon EventBridge を使用してイベント通知を管理し、データウェアハウスの変更に関する最新情報を伝えます。詳細については、「Amazon EventBridge による Amazon Redshift Serverless イベント通知」を参照してください。	プロビジョニングされたクラスターでは、イベント通知を管理するために、Amazon Redshift コンソールを使用してイベントサブスクリプションを作成します。詳細については、「イベント通知サブスクリプションを作成する」を参照してください。

機能	説明	サーバーレス	プロビジョンド
カーソルの制約	Amazon Redshift は、カーソルのすべての結果セットのサイズに対して制約を適用します。	Amazon Redshift Serverless では、カーソルの結果セットの最大合計サイズは 150,000 MB です。	プロビジョニング済みクラスターの場合、カーソルの結果セットの最大合計サイズはクラスタータイプによって異なります。詳細については、「カーソルの制約」を参照してください。

プロビジョニングされたクラスター用の Amazon Redshift 管理インターフェイスの使用

> **Note**
>
> このトピックでは、プロビジョニングされたクラスター用の Amazon Redshift 管理インターフェイスの使用について説明します。Amazon Redshift Serverless と Amazon Redshift Data API には、同様の管理インターフェイスがあります。

Amazon Redshift は、AWS SDK、AWS Command Line Interface (AWS CLI)、および Amazon Redshift 管理 API といった、Amazon Redshift クラスターの作成、管理、削除に使用できる複数の管理インターフェイスをサポートしています。

Amazon Redshift API – リクエストを送信することでこの Amazon Redshift 管理 API を呼び出すことができます。リクエストは、`GET` という名前のパラメータで HTTP 動詞 `POST` または `Action` を使用する HTTP または HTTPS リクエストです。Amazon Redshift API の呼び出しは、Amazon Redshift サービスにアクセスする最も直接的な方法です。しかし、エラー処理やリクエストに署名するハッシュの生成など、低レベルの詳細な作業をアプリケーションで処理する必要があります。

- Amazon Redshift API リクエストの構築と署名の詳細については、「HTTP リクエストへの署名」を参照してください。

- Amazon Redshift API アクションと Amazon Redshift のデータ型については、Amazon Redshift API リファレンスを参照してください。

AWS SDK – AWS SDK は Amazon Redshift クラスター関連のオペレーションに使用できます。一部の SDK ライブラリは基礎となる Amazon Redshift API をラップします。このようなライブラリは、特定のプログラミング言語に API 機能を統合し、多くの低レベルの詳細な作業 (署名の計算、リクエスト再試行の処理、エラー処理など) を処理します。SDK ライブラリのラッパー関数を呼び出すことで、Amazon Redshift クラスターを管理するアプリケーションの作成プロセスが大幅に簡易化されます。

- Amazon Redshift は、AWS SDKs for Java、.NET、PHP、Python、Ruby、および Node.js でサポートされています。Amazon Redshift のラッパー関数の詳細については、各 SDK のリファレンスマニュアルを参照してください。AWS SDK のリストとドキュメントのリンクについては、「Tools for Amazon Web Services」を参照してください。
- このガイドでは、Java SDK を使用して Amazon Redshift を操作する例について説明します。より一般的な AWS SDK コード例については、「AWS SDK を使用した Amazon Redshift のコード例」を参照してください。

AWS CLI – CLI では、Windows、Mac、および Linux コンピュータから AWS サービスを管理する際に使用できるコマンドラインツールのセットが提供されています。AWS CLI には、Amazon Redshift API アクションに基づくコマンドが含まれています。

- Amazon Redshift CLI のインストールとセットアップの詳細については、Amazon Redshift CLI のセットアップ を参照してください。
- Amazon Redshift CLI コマンドのリファレンス資料については、AWS CLI リファレンスの「Amazon Redshift」を参照してください。

このサービスを AWS SDK で使用する

AWS Software Development Kit (SDK) は、多くの一般的なプログラミング言語で使用できます。各 SDK には、デベロッパーが好みの言語でアプリケーションを簡単に構築できるようになる API、コード例、およびドキュメントが提供されています。

SDK ドキュメント	コードの例
AWS SDK for C++	AWS SDK for C++ コードの例

SDK ドキュメント	コードの例
AWS CLI	AWS CLI コードの例
AWS SDK for Go	AWS SDK for Go コードの例
AWS SDK for Java	AWS SDK for Java コードの例
AWS SDK for JavaScript	AWS SDK for JavaScript コードの例
AWS SDK for Kotlin	AWS SDK for Kotlin コードの例
AWS SDK for .NET	AWS SDK for .NET コードの例
AWS SDK for PHP	AWS SDK for PHP コードの例
AWS Tools for PowerShell	AWS Tools for PowerShell コードの例
AWS SDK for Python v4	AWS SDK for Python v4 のコード例
AWS SDK for Python (Boto3)	AWS SDK for Python (Boto3) コードの例
AWS SDK for Ruby	AWS SDK for Ruby コードの例
AWS SDK for Rust	AWS SDK for Rust コードの例
AWS SDK for SAP ABAP	AWS SDK for SAP ABAP コードの例
AWS SDK for Swift	AWS SDK for Swift コードの例

可用性の例

必要なものが見つからなかった場合。このページの下側にある [Provide feedback] リンクから、コードの例をリクエストしてください。

HTTP リクエストへの署名

Amazon Redshift では、管理 API に送信するすべてのリクエストが署名によって認証されている必要があります。このトピックでは、リクエストに署名する方法について説明します。

いずれかの AWS ソフトウェア開発キット (SDK) または AWS Command Line Interface を使用する場合、リクエストは自動的に署名されるため、このセクションは省略してかまいません。AWS SDK の使用方法の詳細については、「プロビジョニングされたクラスター用の Amazon Redshift 管理インターフェイスの使用」を参照してください。Amazon Redshift Command Line Interface の使用に関する詳細については、Amazon Redshift コマンドラインリファレンスをご覧ください。

リクエストに署名するには、暗号化ハッシュ関数を使用してデジタル署名を計算します。暗号化ハッシュは、入力データから一意のハッシュ値生成して返す関数です。ハッシュ関数への入力には、リクエストのテキストと、一時的な認証情報から取得できるシークレットアクセスキーが含まれます。ハッシュ関数から返されるハッシュ値をリクエストに署名として含めます。署名は、リクエストの `Authorization` ヘッダーの一部です。

> **Note**
>
> AWS マネジメントコンソール の外部で AWS を操作するには、プログラマチックアクセス権が必要です。プログラムによるアクセス権を付与する方法は、AWS にアクセスしているユーザーのタイプによって異なります。
> ユーザーにプログラムによるアクセス権を付与するには、以下のいずれかのオプションを選択します。

プログラムによるアクセス権を必要とするユーザー	目的	方法
IAM	(推奨) 一時的な認証情報としてコンソール認証情報を使用して、AWS CLI、AWS SDK、または AWS API へのプログラマチックリクエストに署名します。	使用するインターフェイスの指示に従ってください。 • AWS CLI については、「AWS Command Line Interface ユーザーガイド」の「AWS ローカル開発用のログイン」を参照してください。 • AWS SDK については、「AWS SDK およびツールリファレンスガイド」の「AWS ローカル開発

プログラムによるアクセス権を必要とするユーザー	目的	方法
		用のログイン」を参照してください。
ワークフォースアイデンティティ (IAM アイデンティティセンターで管理されているユーザー)	一時的な認証情報を使用して、AWS CLI、AWS SDK、または AWS API へのプログラムによるリクエストに署名します。	使用するインターフェイスの指示に従ってください。 • AWS CLI については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM アイデンティティセンター を使用するための AWS CLI の設定」を参照してください。 • AWS SDK、ツール、および AWS API については、AWS SDK とツールリファレンスガイドの「IAM アイデンティティセンター認証」を参照してください。
IAM	一時的な認証情報を使用して、AWS CLI、AWS SDK、または AWS API へのプログラムによるリクエストに署名します。	IAM ユーザーガイドの「AWS リソースでの一時的な認証情報の使用」の指示に従ってください。

プログラムによるアクセス権を必要とするユーザー	目的	方法
IAM	(非推奨) 長期的な認証情報を使用して、AWS CLI、AWS SDK、または AWS API へのプログラムによるリクエストに署名します。	使用するインターフェイスの指示に従ってください。 • AWS CLI については、「AWS Command Line Interface ユーザーガイド」の「IAM ユーザー認証情報を使用した認証」を参照してください。 • AWS SDK とツールについては、「AWS SDK とツールリファレンスガイド」の「長期認証情報を使用して認証する」を参照してください。 • AWS API については、「IAM ユーザーガイド」の「IAM ユーザーのアクセスキーの管理」を参照してください。

Amazon Redshift は、リクエストを受け取ると、リクエストの署名に使用されたものと同じハッシュ関数と入力データを使用して署名を再計算します。再計算された署名がリクエストに含まれる署名と一致した場合、Amazon Redshift はリクエストを処理します。それ以外の場合、リクエストは拒否されます。

Amazon Redshift では、AWS 署名バージョン 4 を使用した認証がサポートされています。署名を計算するプロセスは 3 つのタスクで構成されます。これらのタスクをこの後の例に示します。

- タスク 1: 正規リクエストを作成する

  HTTP リクエストを正規形式に変換します。正規形式を使用する必要がある理由は、送信した署名と比較するために、Amazon Redshift で同じ正規形式を使用して署名が計算されるためです。

- タスク 2: 署名文字列を作成する

  暗号化ハッシュ関数への入力値の 1 つとして使用する文字列を作成します。署名文字列と呼ばれる文字列は、ハッシュアルゴリズムの名前、要求日付、認証情報スコープの文字列、および前のタスクで正規化されたリクエストを結合したものです。認証情報スコープの文字列自体は、日付、リージョン、およびサービス情報を結合したものです。

- タスク 3: 署名を計算する

  2 つの入力文字列 (署名文字列と派生キー) を受け取る暗号化ハッシュ関数を使用して、リクエストの署名を計算します。シークレットアクセスキーから、認証情報スコープの文字列を使用して一連のハッシュベースのメッセージ認証コード (HMAC-SHA256) を作成することで、派生キーが計算されます。

署名の計算例

次の例に、CreateCluster リクエストの署名を作成する詳細な手順を示します。この例を参考にして実際の署名の計算方法を確認できます。その他の参考計算例は、「IAM ユーザーガイド」の「リクエスト署名の例」セクションに記載されています。

GET または POST リクエストを使用してリクエストを Amazon Redshift に送信できます。この 2 つの違いとして、GET リクエストではパラメータがクエリ文字列パラメータとして送信されるのに対し、POST リクエストではパラメータがリクエストの本文に含まれます。次の例は POST リクエストを示しています。

例では、次のように想定しています。

- リクエストのタイムスタンプは `Fri, 07 Dec 2012 00:00:00 GMT`
- エンドポイントは米国東部 (バージニア北部) リージョン `us-east-1`

汎用リクエスト構文:

```
https://redshift.us-east-1.amazonaws.com/
   ?Action=CreateCluster
   &ClusterIdentifier=examplecluster
   &MasterUsername=masteruser
   &MasterUserPassword=12345678Aa
   &NumberOfNode=2
   &NodeType=dc2.large
   &Version=2012-12-01
```

```
   &x-amz-algorithm=AWS4-HMAC-SHA256
   &x-amz-credential=AKIAIOSFODNN7EXAMPLE/20121207/us-east-1/redshift/aws4_request
   &x-amz-date=20121207T000000Z
   &x-amz-signedheaders=content-type;host;x-amz-date
```

「タスク 1: 正規リクエストを作成する」で計算されるリクエストの正規形式は次のとおりです。

```
POST
/

content-type:application/x-www-form-urlencoded; charset=utf-8
host:redshift.us-east-1.amazonaws.com
x-amz-date:20121207T000000Z

content-type;host;x-amz-date
55141b5d2aff6042ccd9d2af808fdf95ac78255e25b823d2dbd720226de1625d
```

正規リクエストの最後の行はリクエストボディのハッシュです。この API にはクエリパラメータがないため、正規リクエストの 3 行目は空です。

「タスク 2: 署名する文字列を作成する」で署名する文字列は次のとおりです。

```
AWS4-HMAC-SHA256
20121207T000000Z
20121207/us-east-1/redshift/aws4_request
06b6bef4f4f060a5558b60c627cc6c5b5b5a959b9902b5ac2187be80cbac0714
```

署名する文字列の最初の行はアルゴリズム、2 行目はタイムスタンプ、3 行目は認証情報スコープ、最後の行は「タスク 1: 正規リクエストを作成する」で作成した正規リクエストのハッシュです。認証情報スコープで使用するサービス名は `redshift` です。

「タスク 3: 署名を計算する」について、派生キーは次のように表されます。

```
derived key = HMAC(HMAC(HMAC(HMAC("AWS4" + YourSecretAccessKey,"20121207"),"us-
east-1"),"redshift"),"aws4_request")
```

派生キーは連続するハッシュ関数として計算されます。上記の式の内側の HMAC ステートメントでは、**AWS4** というフレーズをシークレットアクセスキーと連結し、それをキーとして使用して「us-east-1」データをハッシュします。このハッシュの結果が次のハッシュ関数のキーになります。

派生キーを計算した後、署名する文字列と派生キーという 2 つの入力文字列を受け取るハッシュ関数でそのキーを使用します。例えば、シークレットアクセスキー wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY を使用し、前述の文字列に署名する場合、計算された署名は次のようになります。

```
9a6b557aa9f38dea83d9215d8f0eae54100877f3e0735d38498d7ae489117920
```

最後のステップは、Authorization ヘッダーの構築です。デモンストレーションのアクセスキー AKIAIOSFODNN7EXAMPLE の場合、ヘッダーは次のとおりです (読みやすいように改行しています)。

```
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20121207/us-east-1/
redshift/aws4_request,
SignedHeaders=content-type;host;x-amz-date,
Signature=9a6b557aa9f38dea83d9215d8f0eae54100877f3e0735d38498d7ae489117920
```

Amazon Redshift CLI のセットアップ

ここでは、Amazon Redshift の管理に使用するために、AWS CLI コマンドラインツールをセットアップして実行する方法について説明します。Amazon Redshift コマンドラインツールは、AWS Command Line Interface (AWS CLI) で実行されており、Python (https://www.python.org/) が順番に使用されます。AWS CLI は、Python をサポートするあらゆるオペレーティングシステムで実行できます。

AWS Command Line Interface のインストール

Amazon Redshift コマンドラインツールを初めて使用する場合、まず AWS CLI をセットアップし、Amazon Redshift CLI オプションを定義する設定ファイルを追加します。

別の AWS CLI サービス用に AWS を既にインストールして設定している場合は、この手順をスキップできます。

AWS Command Line Interfaceをインストールするには

1. 「Install or update to the latest version of the AWS CLI」に進み、AWS CLI のインストール手順に従います。

   CLI アクセスには、アクセスキー ID とシークレットアクセスキーが必要です。長期のアクセスキーの代わりに一時的な認証情報をできるだけ使用します。一時的な認証情報には、アクセス

キー ID、シークレットアクセスキー、および認証情報の失効を示すセキュリティトークンが含まれています。詳細については、IAM ユーザーガイドの「AWS リソースを使用した一時的なセキュリティ認証情報の使用」を参照してください。

2. アクセスキー、デフォルトのリージョン、コマンドの出力形式など、設定情報を含むファイルを作成します。次に、そのファイルを参照する AWS_CONFIG_FILE 環境変数を設定します。詳細な手順については、AWS Command Line Interface ユーザーガイドの「Configuring the AWS CLI」にアクセスしてください。
3. テストコマンドを実行して、AWS CLI インターフェイスが機能していることを確認します。たとえば、次のコマンドを実行すると、AWS CLI のヘルプ情報が表示されます。

```
aws help
```

次のコマンドを実行すると、Amazon Redshift のヘルプ情報が表示されます。

```
aws redshift help
```

Amazon Redshift CLI コマンドのリファレンス資料については、AWS CLI リファレンスの「Amazon Redshift」にアクセスしてください。

Amazon Redshift Serverless

Amazon Redshift Serverless を使用すると、オンプレミスのデータウェアハウスをプロビジョンして管理する必要なく、分析を実行してスケールできます。Amazon Redshift Serverless を使用すると、データアナリスト、デベロッパー、データサイエンティストは、Amazon Redshift でデータウェアハウスにデータをロードしてレコードをクエリし、データからのインサイトを数秒で取得できるようになりました。Amazon Redshift は、データウェアハウス容量のプロビジョニングとスケーリングを自動的に実行して、要求が厳しく、予測不可能なワークロードのための高速なパフォーマンスを実現します。料金については、使用した容量のみのお支払いです。既存の分析アプリケーションやビジネスインテリジェンスアプリケーションに変更を加えることなく、この単純化によるメリットを得ることができます。

Amazon Redshift Serverless の詳細については、「Amazon Redshift サービスレベルアグリーメント」を参照してください。

Amazon Redshift Serverless とは

Amazon Redshift Serverless は、データウェアハウスの容量を自動的にプロビジョニングし、基盤となるリソースをインテリジェントにスケーリングします。Amazon Redshift Serverless による容量の調整は数秒以内に行われます。最も要求の厳しいワークロードや揮発性ワークロードに対しても、一貫して高いパフォーマンスとシンプルなオペレーションを実現します。

Amazon Redshift Serverless を使用すると、以下の機能によるメリットを得られます。

- Amazon Redshift でプロビジョニングされたクラスターで、データにアクセスして分析を行う際に、セットアップや調整そして管理の必要はありません。
- Amazon Redshift SQL の優れた機能、業界でもトップクラスのパフォーマンス、さらにデータレイクとの統合などを使用して、データウェアハウス、データレイク、オペレーショナルなデータソースをシームレスにクエリできます。
- インテリジェントなオートスケーリングにより、最も要求の厳しい不安定なワークロードでも、高いパフォーマンスとシンプルなオペレーションを実現できます。
- ワークグループと名前空間を使用して、きめ細かなコスト管理でコンピューティングリソースとデータを整理できます。
- 料金は、データウェアハウスが使用されている場合にのみ発生します。

Amazon Redshift Serverless では、コンソールインターフェイスを使用してサーバーレスデータウェアハウスまたは API にアクセスしてアプリケーションを構築します。データウェアハウスを通じて、Amazon Redshift マネージドのストレージや Amazon S3 データレイクにアクセスできます。

この動画では、Amazon Redshift Serverless が、データウェアハウスのインフラストラクチャを管理することなく、分析を簡単に実行してスケールする方法を示します。

Amazon Redshift Serverless コンソール

Amazon Redshift Serverless コンソールの使用開始については、次の動画、Amazon Redshift Serverless の開始方法をご覧ください。

Serverless ダッシュボード

[Serverless dashboard] (サーバーレスダッシュボード) ページでは、リソースの概要に加え使用状況のグラフを表示できます。

- [Namespace overview] (名前空間の概要) — このセクションには、名前空間内のスナップショットとデータ共有の量が表示されます。
- [Workgroups] (ワークグループ) — このセクションには、Amazon Redshift Serverless 内のすべてのワークグループが表示されます。
- [Queries metrics] (クエリのメトリクス) – このセクションには、過去 1 時間のクエリによるアクティビティが表示されます。
- [RPU capacity used] (使用された RPU キャパシティー) – このセクションには、過去 1 時間に使用されたキャパシティーが表示されます。
- [Free trial] (無料トライアル) — このセクションには、AWS アカウントに残っている無料トライアルクレジットが表示されます。これは、同じアカウントでのスナップショット、ストレージ、ワークグループなど、Amazon Redshift Serverless リソースとオペレーションのすべての使用率をカバーします。
- [Alarms] (アラーム) — このセクションには、Amazon Redshift Serverless で設定したアラームが表示されます。

データバックアップ

[Data backup] (データバックアップ) タブでは、次の操作が行えます。

- [Snapshots] (スナップショット) – Amazon Redshift Serverless のデータについて、スナップショットの作成、削除、管理ができます。デフォルトの保持期間は `indefinitely` ですが、保存

期間は 1～3653 日の任意の値に設定できます。スナップショットから名前空間を復元するように AWS アカウント に許可できます。

- [Recovery points] (復旧ポイント) – 過去 24 時間以内に発生した、偶発的な書き込みまたは削除からの復旧のために自動的に作成された復旧ポイントが表示されます。データを復元するには、使用可能な任意の名前空間に復旧ポイントを復元できます。復旧ポイントからスナップショットを作成することで、復旧のための時点をより長い期間保持できます。デフォルトの保持期間は `indefinitely` ですが、保存期間は 1 ~ 3653 日の任意の値に設定できます。

データアクセス

[Data access] (データアクセス) タブでは、次の操作が行えます。

- ネットワークとセキュリティ設定 – VPC 関連の値、AWS KMS 暗号化のための値、および監査ログ記録の値を表示できます。更新できるのは、監査ログのみです。
- AWS KMS key — Amazon Redshift Serverless でリソースを暗号化するために使用される AWS KMS key。
- [Permissions] (アクセス許可) – ユーザーに代わってリソースを使用するために Amazon Redshift Serverless が引き受けることができる、IAM ロールの管理を行います。詳細については、「Amazon Redshift Serverless での Identity and Access Management」を参照してください。
- [Redshift-managed VPC endpoints] (RedShift 管理の VPC エンドポイント) – 別の VPC またはサブネットから Amazon Redshift Serverless インスタンスにアクセスできます。詳細については、「他の VPC エンドポイントからの Amazon Redshift Serverless への接続」を参照してください。

制限

[Limits] (制限) タブでは、以下の操作が行えます。

- [Base capacity in Redshift processing units (RPUs)] (Redshift プロセッシング単位 (RPU) の基本キャパシティ) – ワークロードの処理に使用する基本容量を設定できます。クエリのパフォーマンスを向上させるには、RPU の値を増やします。
- [Usage limits] (使用制限) – アクションが開始される前の期間に、Amazon Redshift Serverless インスタンスが使用できる最大のコンピューティングリソース。Amazon Redshift Serverless がワークロードを実行するために使用するリソースの量を制限します。使用量の単位は、Redshift プロセッシング単位 (RPU) 時間で表されます。RPU 時間は、1 時間に使用された RPU の数です。設定したしきい値に達した際に次のとおりのアクションを実行するように定義できます。
  - アラートを送信します。

- エントリをシステムテーブルにログ記録します。
- ユーザークエリを無効にします。

制限最大 4 つまで設定できます。

- [Query limits] (クエリの制限) — 制限を追加してパフォーマンスと制限をモニタリングできます。クエリモニタリングの制限の詳細については、「WLM クエリモニタリングルール」を参照してください。

詳細については、「Amazon Redshift Serverless 容量を計算する」を参照してください。

データ共有

[Datashares] (データ共有) タブでは、次の操作が行えます。

- [Datashares created in my namespace] (自分の名前空間で作成されたデータ共有) 設定 – データ共有を作成し、それを他の名前空間や AWS アカウント と共有できます。
- 他の名前空間と AWS アカウント からのデータ共有 — 他の名前空間や AWS アカウント からのデータ共有から、データベースを作成できます。

データ共有の詳細については、「Amazon Redshift Serverless でのデータ共有」を参照してください。

クエリとデータベースのモニタリング

[Query and database monitoring] (クエリとデータベースのモニタリング) ページでは、[Query history] (クエリ履歴)、および [Database performance] (データベースパフォーマンス) についてのグラフを表示できます。

[Query history] (クエリ履歴) タブには、以下のようなグラフが表示されます ([Query list] (クエリリスト) か [Resource metrics] (リソースメトリクス) のどちらかを選択します)。

- [Query runtime] (クエリランタイム) – このグラフでは、同じ時間フレームで実行されているクエリが表示されます。グラフないのバーを選択すると、クエリ実行の詳細を表示できます。
- [Queries and loads] (クエリとロード) – このセクションでは、[Query ID] (クエリ ID) ごとにクエリとロードが一覧表示されます。
- [RPU capacity used] (使用された RPU キャパシティー) – このグラフは、Redshift プロセッシング単位 (RPU) により全体的なキャパシティーを表示します。

- [Database connections] (データベース接続) – このグラフは、アクティブなデータベース接続の数を表示します。

データベースパフォーマンス

[Database performance] (データベースパフォーマンス) タブには、以下のグラフが表示されます。

- [Queries completed per second] (1 秒あたりに完了したクエリ数) – このグラフには、1 秒の間に完了したクエリの平均数が表示されます。
- [Queries duration] (クエリの期間) – このグラフには、クエリを完了するまでの平均時間が表示されます。
- [Database connections] (データベース接続) – このグラフは、アクティブなデータベース接続の数を表示します。
- [Running queries] (実行中のクエリ数) – このグラフでは、特定の時間中に実行されているクエリの総数が表示されます。
- [Queued queries] (キューに追加済みのクエリ) – このグラフは、特定の時間中にキューに格納されたクエリの総数を示します。
- [Query run time breakdown] (クエリ実行時の内訳) – このグラフには、クエリのタイプごとに、クエリの実行にかかった合計時間が表示されます。

リソースのモニタリング

[Resource monitoring] (リソースのモニタリング) ページでは、消費しているリソースに関するグラフを表示できます。複数のファセットに基づいてデータをフィルタリングできます。

- [Metric filter] (メトリクスフィルター) — メトリクスフィルターを使用して、特定のワークグループのフィルターを選択したり、時間範囲と時間間隔を選択できます。
- [RPU capacity used] (使用された RPU キャパシティー) – このグラフは、全体的なキャパシティーを、Redshift プロセッシング単位 (RPU) で表示します。
- [コンピューティングの使用状況] – このグラフは、選択した時間範囲における期間ごとに、RPU 時間の使用量を示します。6 時間未満の時間範囲の場合、RPU 時間は正確な時間で表示されます。6 時間以上の時間範囲の場合、RPU 時間は平均で表示されます。
- 自動最適化用の追加コンピューティングの課金秒数 - このグラフには、選択した時間範囲のデータベースの自動最適化に対して課金された RPU 秒数が表示されます。Amazon Redshift が追加のコンピューティングリソースを使用して最適化を実行すると、自動最適化は課金されます。詳細につ

いては、「データベースの自動最適化のための追加のコンピューティングリソースの割り当て」を参照してください。

データ共有

[Datashares] (データ共有) ページでは、[In my account] (自分のアカウント内)、および [From other accounts] (他のアカウントから) のデータ共有を管理できます。データ共有の詳細については、「Amazon Redshift Serverless でのデータ共有」を参照してください。

AWS Glue Data Catalog

[AWS Glue Data Catalog] タブでは、AWS Glue Data Catalog への名前空間の登録ステータスを確認できます。このタブは、登録プロセスを開始した後にのみ表示されます。名前空間の AWS Glue Data Catalog への登録の詳細については、「Amazon Redshift データベースデベロッパーガイド」の「Apache Iceberg の Amazon Redshift との互換性」を参照してください。

Amazon Redshift サーバーレスを使用する場合の考慮事項

Amazon Redshift Serverless が利用できる AWS リージョン の一覧については、「Amazon Web Services 全般のリファレンス」の「Redshift Serverless API」に記載されているエンドポイントを参照してください。

Amazon Redshift サーバーレスによって使用される一部のリソースは、クォータの対象となります。詳細については、「Amazon Redshift Serverless オブジェクトのクォータ」を参照してください。

カーソルを宣言すると、Amazon Redshift サーバーレスの結果セットサイズの指定が DECLARE に指定されます。Amazon Redshift Serverless では、カーソルの結果セットの最大合計サイズは 150,000 MB です。

オンラインパッチ適用 – Amazon Redshift Serverless は、従来のメンテナンスウィンドウを必要とせずに、自動ソフトウェア更新を提供します。新しい更新が利用可能になると、システムはアイドル期間中にリリースから 14 日以内に更新を適用します。この更新プロセスには最大 15 分かかることがあります。14 日以内に 15 分間のアイドル期間が発生しない場合、サーバーレスエンドポイントが短時間使用できなくなる可能性があります。この間、エンドポイントへのアプリケーション接続が失敗する可能性があります。Redshift パッチリリースは、「Amazon Redshift のクラスターバージョン」ドキュメントでモニタリングできます。Amazon Redshift Serverless の詳細については、「Amazon Redshift サービスレベルアグリーメント」を参照してください。

トラック — Amazon Redshift が新しいワークグループバージョンをリリースすると、ワークグループは自動的に更新されます。ワークグループを最新のリリースに更新するか、前のリリースに更新す

るかを指定できます。トラックの詳細については、「Amazon Redshift でプロビジョニングされたクラスターとサーバーレスワークグループのトラック」を参照してください。

アベイラビリティゾーン ID - Amazon Redshift Serverless インスタンスを設定する際には、[その他の考慮事項] を開き、[サブネット] に表示されているサブネット ID に、サポートされているアベイラビリティゾーン ID が少なくとも 2 つ含まれていることを確認します。

- 拡張された VPC のルーティング (EVR) のないワークグループの場合、2 つのアベイラビリティーゾーン (AZ) が必要です。
- EVR を使用するワークグループの場合、3 つの AZ が必要です。

サブネットからアベイラビリティーゾーン ID へのマッピングを確認するには、VPC コンソールを開いて [Subnets] (サブネット) をクリックし、アベイラビリティーゾーン ID とサブネット ID のリストを表示します。サポートされているアベイラビリティーゾーン ID に、サブネットがマッピングされていることを確認します。サブネットの作成については、Amazon VPC ユーザーガイドの「VPC にサブネットを作成する」を参照してください。

2 つのサブネット (EVR なし) - 少なくとも 2 つのサブネットが必要で、それらが 2 つのアベイラビリティーゾーンにまたがっている必要があります。

3 つのサブネット (EVR を使用する場合のみ) - EVR を使用する場合、少なくとも 3 つのサブネットが必要で、それらが 3 つ以上のアベイラビリティーゾーンにまたがっている必要があります。

空き IP アドレスの要件 - 拡張された VPC ルーティング (EVR) を有効にせずに Redshift Serverless を使用する場合は、各サブネットで 3 つ以上の空き IP アドレスが必要です。これは、サービスが適切に機能するための要件です。

Redshift Serverless のデプロイ用に RPU を更新する場合、サービスの運用要件に対応するために、各サブネットに 3 つ以上の空き IP アドレスが必要です。

Amazon VPC での IP アドレスの割り当てと IP アドレス指定の詳細については、Amazon VPC ユーザーガイドの「VPC とサブネットの IP アドレス指定」を参照してください。

Without EVR

拡張された VPC ルーティングを使用しない場合でも、ベース RPU のサイズ (4 から 1024 RPU) やワークグループの RPU 使用量またはワークグループに関係なく、各サブネットに 3 つ以上の空き IP アドレスが必要です。AI 主導のスケーリングと最適化機能が有効なワークグループには、3 つ以上の空き IP アドレスが必要になります。

With Enhanced VPC Routing (EVR)

Redshift Serverless で拡張された VPC ルーティングを使用する場合、ワークグループの作成時に必要な IP アドレスの最小数は次のとおりです。

Redshift プロセッシング単位 (RPU)	必要な空き IP アドレス	最小 CIDR サイズ
4	9	/27
8	9	/27
16	13	/27
32	13	/27
64	21	/27
128	37	/26
256	69	/25
512	133	/24
1024	261	/23

EVR を使用する場合、ワークグループを更新して使用する RPU を増やす際にも、空き IP アドレスが必要です。ワークグループのサブネットを更新する際に必要な空き IP アドレスの数は次のとおりです。

Redshift プロセッシング単位 (RPU)	更新された Redshift プロセッシング単位 (RPU)	必要な空き IP アドレス
4	8	10
8	16	10
16	32	13
32	64	16

Redshift プロセッシング単位 (RPU)	更新された Redshift プロセッシング単位 (RPU)	必要な空き IP アドレス
64	128	28
128	256	52
256	512	100
512	1024	197

> **Note**
>
> 1024 の最大ベース RPU 容量は、次の AWS リージョンでのみ使用できます。
>
> - 米国東部 (バージニア北部)
> - 米国東部 (オハイオ)
> - 米国西部 (オレゴン)
> - 欧州 (アイルランド)
> - 欧州 (ロンドン)

IP アドレスの割り当ての詳細については、「Amazon VPC ユーザーガイド」の「IP アドレス指定」を参照してください。

移行後のストレージスペース - 小さな Amazon Redshift プロビジョニングクラスターを Amazon Redshift Serverless に移行すると、移行後にストレージ領域の割り当てが増加することがあります。これは、ストレージ領域の割り当てが最適化され、ストレージ領域が事前に割り当てられたためです。この領域は、Amazon Redshift Serverless でデータが増加するにつれ、一定期間使用されます。

Amazon Redshift Serverless と Amazon Redshift でプロビジョニングされたクラスター間のデータ共有 - Amazon Redshift Serverless がプロデューサーで、プロビジョニングされたクラスターがコンシューマーであるデータ共有の場合、プロビジョニングされたクラスターのクラスターバージョンは 1.0.38214 以降である必要があります。これ以前のクラスターバージョンを使用する場合、クエリを実行するとエラーが発生します。クラスターのバージョンは、Amazon Redshift コンソールの [メンテナンス] タブで確認できます。`SELECT version();` を実行することもできます。

最大クエリ実行時間 - 経過したクエリ実行時間 (秒)。キューでの待機時間は実行時間に含まれません。クエリが設定された実行時間を超えると、Amazon Redshift Serverless はクエリを中止します。有効な値は 0 ~ 86,399 です。

インターリーブソートキーを含むテーブルの移行 — Amazon Redshift でプロビジョニングされたクラスターを Amazon Redshift サーバーレスに移行すると、Redshift はインターリーブされたソートキーと DISTSTYLE KEY を含むテーブルを複合ソートキーに変換します。DISTSTYLE は変更されません。ディストリビューションスタイルの詳細については、「Amazon Redshift 開発者ガイド」の「データディストリビューションスタイルの操作」を参照してください。ソートキーの詳細については、「ソートキーの使用」を参照してください。

VPC 共有 – 共有 VPC 内に Amazon Redshift Serverless ワークグループを作成できます。作成する場合は、リソース共有を削除しないことをお勧めします。ワークグループが使用できなくなる可能性があります。

IPv6 のサポート

Amazon Redshift Serverless は、AWS 仮想プライベートクラウド (VPC) において IPv4 アドレスと IPv6 アドレスの両方 (デュアルスタック) または IPv4 専用設定で Amazon Redshift ワークグループの設定をサポートします。新しい Amazon Redshift Serverless ワークグループを作成するときに IPv6 サポートを有効にしたり、IPv6 アドレス指定をサポートするように既存のワークグループを変更したりできます。この機能を使用すると、Amazon Redshift Serverless ウェアハウスを IPv6 対応 VPC サブネットにデプロイし、アプリケーションのアドレス空間要件の拡大をサポートするようにネットワーク設定を構成することができます。アプリケーションは、IPv4 プロトコルまたは IPv6 プロトコルを使用して Amazon Redshift Serverless ウェアハウスと通信できるようになりました。これにより、既存のネットワークアーキテクチャと将来のネットワークアーキテクチャの両方との互換性が確保されます。

Amazon Redshift Serverless 容量を計算する

Amazon Redshift Serverless では、ワークロードの要件に合わせてコンピューティングキャパシティが自動的にスケールアップおよびスケールダウンします。コンピューティングキャパシティとは、Amazon Redshift Serverless ワークロードに割り当てられた処理能力とメモリを指します。一般的なユースケースには、ピークトラフィック期間の処理、複雑な分析の実行、大量のデータの効率的な処理などがあります。以下の用語は、Amazon Redshift がコンピューティングキャパシティを管理する方法の詳細を示します。

RPU

Amazon Redshift Serverless では、Redshift プロセッシング単位 (RPU) で、データウェアハウスの容量が測定されています。RPU は、ワークロードの処理に使用されるリソースです。1 つの RPU が 16 GB のメモリを提供します。

基本容量

この設定は、Amazon Redshift Serverless がクエリの処理に使用するデータウェアハウスの基本容量を指定します。基本容量は、RPU で指定します。基本の容量は、Redshift 処理ユニット (RPU) で設定できます。ベース容量を大きく設定することで、特に大量のリソースを必要とするデータ処理ジョブでは、クエリのパフォーマンスが向上します。Amazon Redshift Serverless のデフォルトの基本容量は、128 RPU です。ベース容量設定は、4 RPU ~ 512 RPU の間で調整できます。この値は、4 RPU に設定するか、8 RPU 以上に 8 の増分で (8、16、24...512) 設定できます。この値は、AWS コンソール、`UpdateWorkgroup` API オペレーション、または AWS CLI の `update-workgroup` オペレーションを使用して設定できます。

最小ベース容量の 4 RPU では、データウェアハウスのコストと容量の要件に基づいて、より単純なワークロードからより複雑なワークロードまで柔軟に実行できます。ベース容量の 4 RPU は 32 TB 未満のデータを含むウェアハウスを対象とし、ベース容量の 8、16、24 RPU は 128 TB 未満のデータを必要とするワークロードを対象としています。データ要件が 128 TB を超える場合は、最低 32 RPU のベース容量を使用する必要があります。さらに、列数が多くて同時実行性が高いテーブルがあるワークロードでは、32 RPU 以上のベース容量を使用することをお勧めします。

使用可能な最大ベース容量である 1024 RPU は、ワークロードに最高レベルのコンピューティングリソースを追加します。これにより、複雑性の高いワークロードをサポートする柔軟性が向上し、データのロードとクエリが高速化されます。

> **Note**
>
> 拡張された最大ベース容量の 1024 RPU は、次の AWS リージョンで使用できます。他のリージョンの場合、最大ベース容量は 512 RPU です。
>
> - 米国東部 (バージニア北部)
> - 米国東部 (オハイオ)
> - 米国西部 (オレゴン)
> - 欧州 (アイルランド)
> - 欧州 (フランクフルト)

基本容量を 512-1024 に設定すると、RPU を 32 単位で増減できます。

大規模で複雑なワークロードを管理する場合は、Redshift Serverless データウェアハウスのサイズを増やすことを検討してください。大規模なウェアハウスは、より多くのコンピューティングリソースにアクセスできるため、クエリをより効率的に処理できます。

基本容量を増やすことが有益であるインスタンスを次に示します。

- 実行に時間がかかる複雑なクエリがある
- テーブルに多数の列がある
- クエリに多数の JOIN がある
- クエリが、データレイクなどの外部ソースから大量のデータを集約またはスキャンする

Amazon Redshift Serverless のクォータと制限の詳細については、「Amazon Redshift Serverless オブジェクトのクォータ」を参照してください。

Amazon Redshift Serverless 容量に関する考慮事項と制限

Amazon Redshift Serverless 容量に関する考慮事項と制限事項は次のとおりです。Redshift Serverless の一般的な考慮事項については、「Amazon Redshift サーバーレスを使用する場合の考慮事項」を参照してください。

- ベース容量が 4 RPU の設定では、最大 32 TB のマネージドストレージ容量をサポートします。32 TB を超えるマネージドストレージを使用している場合は、8 RPU 未満にベース RPU を設定することはできません。
- ベース容量が 8 RPU または 16 RPU の設定では、最大 128 TB の Redshift マネージドストレージ容量をサポートします。128 TB を超えるマネージドストレージを使用している場合は、32 RPU 未満にベースを設定することはできません。
- ワークグループのベース容量を編集すると、ワークグループで実行されているクエリの一部がキャンセルされる場合があります。
- Redshift Serverless は、以下の増分を使用してデータウェアハウスの RPU をスケールします。
  - 4 ~ 8 RPU: 4 RPU ずつ増加します。
  - 8 ~ 512 RPU: 8 RPU ずつ増加します。
  - 512 ~ 1024 RPU: 32 RPU ずつ増加します。

- バキュームブーストは 8 RPU 以上でのみサポートされています。8 RPU 以下の場合は、代わりに次のコマンドを使用します。

```
VACUUM [FULL | SORT ONLY | DELETE ONLY | REINDEX | RECLUSTER] [table_name] [TO
 threshold PERCENT]
```

4 つの Redshift 処理ユニット (RPU) 容量を持つ Redshift Serverless

ベース容量として 4 RPU を持つ Redshift Serverless は、小規模または要求の少ないワークロードに最適です。このエントリポイントは、柔軟で費用対効果の高いソリューションを提供します。このエントリレベルの設定は、以下のリソース (最大) を持つデータウェアハウスをサポートします。

- 最大 32 TB の Redshift マネージドストレージ。
- テーブルあたり最大 100 列
- 64 GB のメモリ

これらの制限を超える必要がある場合は、自動スケーリングに依存せずに、ベース容量を手動で増やす必要があります。データウェアハウスを 4 RPU を超えてスケールすると、データウェアハウスは引き続きより多くの RPU を使用し、Amazon Redshift はデータウェアハウスを 4 RPU にスケールダウンしなくなります。

> **Note**
>
> ベース容量として 4 RPU を使用する場合、100 列を超えるテーブルを作成できますが、テーブルは 100 列に制限することをお勧めします。この制限を超えると、クエリの実行中にデータウェアハウスのメモリが枯渇してパフォーマンスが低下する場合があります。

以下の AWS リージョンでは、4 RPU を使用するデータウェアハウスを作成できます。

- 米国東部(オハイオ)
- 米国東部 (バージニア北部)
- 米国西部 (北カリフォルニア)
- 米国西部 (オレゴン)
- アジアパシフィック (ムンバイ)
- アジアパシフィック (シンガポール)

- アジアパシフィック (シドニー)
- アジアパシフィック (東京)
- 欧州 (アイルランド)
- 欧州 (ストックホルム)

AI 主導のスケーリングと最適化

> **Note**
>
> AI 主導のスケーリングと最適化は、Amazon Redshift Serverless が利用可能なすべての AWS リージョンで利用できます。

Amazon Redshift Serverless には、さまざまなワークロード要件に対応するため、AI 主導の高度なスケーリングと最適化の機能が用意されています。データウェアハウスは、以下のプロビジョニングの課題を抱える場合があります。

- リソースを大量に消費するクエリのパフォーマンス向上のため、データウェアハウスが過剰にプロビジョニングされる場合がある
- コスト削減のため、データウェアハウスがプロビジョニング不足になる可能性がある

データウェアハウスのワークロードのパフォーマンスとコストの適切なバランスを取るのは、特にアドホッククエリやデータ量が増加する場合には困難です。リソースの消費量が多いクエリと少ないクエリが混在するワークロードを実行している場合には、インテリジェントなスケーリングが必要です。AI 主導のスケーリングと最適化の機能は、データの増加に応じて Serverless コンピューティング (RPU) を自動的にスケールします。また、コストパフォーマンス目標の範囲内でクエリのパフォーマンスを維持するためにも、この機能が役立ちます。データ量が増えるに従ってコンピューティングリソースが動的に割り当てられるので、クエリが引き続きパフォーマンス目標を達成できます。AI 主導のスケーリングと最適化のおかげで、サービスは変動的なワークロード要件にシームレスに適応することができ、手動による介入や複雑なキャパシティプランニングは必要ありません。

Amazon Redshift Serverless は、クエリの複雑さやデータ量などの要因に基づいて、より包括的で応答性の高いスケーリングソリューションを提供します。この機能により、多様なワークロードや増大するデータセットを柔軟かつ効率的に処理しつつ、ワークロードのコストパフォーマンスを最適化できます。Amazon Redshift Serverless は、Serverless ワークグループに対して指定された料金パフォーマンス目標を達成するため、Amazon Redshift Serverless エンドポイントに対して AI 主導の

スケーリングと最適化を自動的に適用します。このような料金パフォーマンスの自動最適化は、ワークロードに設定すべきベース容量が不明な場合や、割り当てられたリソースを増やすことでワークロードの一部で利点が得られる可能性がある場合に特に役立ちます。

例

組織で通常実行しているワークロードは 32 RPU しか必要としないのに、突如さらに複雑なクエリが導入された場合、適切なベース容量がわからない場合があります。ベース容量を高く設定すればパフォーマンスは向上しますが、コストも高くなり、コスト予想から外れる可能性があります。Amazon Redshift Serverless は、AI 主導のスケーリングと最適化を使用して、組織に応じて最適化されたコストを維持しながら、料金パフォーマンスの目標を満たすように RPU を自動的に調整します。この自動最適化は、ワークロードのサイズを問わず役に立ちます。自動最適化を使用すると、複雑なクエリが多数ある場合に、組織の料金パフォーマンス目標の達成につながります。

> **Note**
>
> 料金パフォーマンス目標は、ワークグループ独自の設定です。ワークグループによって、料金パフォーマンス目標が異なる場合があります。

料金パフォーマンスを設定するには、AWS コンソールを使用します。デフォルトでは、すべての新しい Serverless ワークグループに対して料金パフォーマンス目標が有効になり、[Balanced] に設定されます。Serverless ワークグループを作成した後で、料金パフォーマンス目標を変更することができます。

ワークグループのコストパフォーマンス目標を編集するには

1. Amazon Redshift Serverless コンソールで、[ワークグループの設定] を選択します。
2. 料金パフォーマンス目標を編集するワークグループを選択します。[パフォーマンス] タブをクリックして、[編集] を選択します。
3. [コストパフォーマンス目標] を選択し、スライダーを希望の設定に調整します。
4. [変更の保存] をクリックします。
5. Amazon Redshift Serverless がワークロードに割り当てることができる RPU の最大量を変更するには、[ワークグループの設定] の [制限] タブを選択します。

[コストパフォーマンス目標] スライダーを使用して、コストとパフォーマンスの必要なバランスを設定できます。スライダーを動かして、次のいずれかのオプションを選択できます。

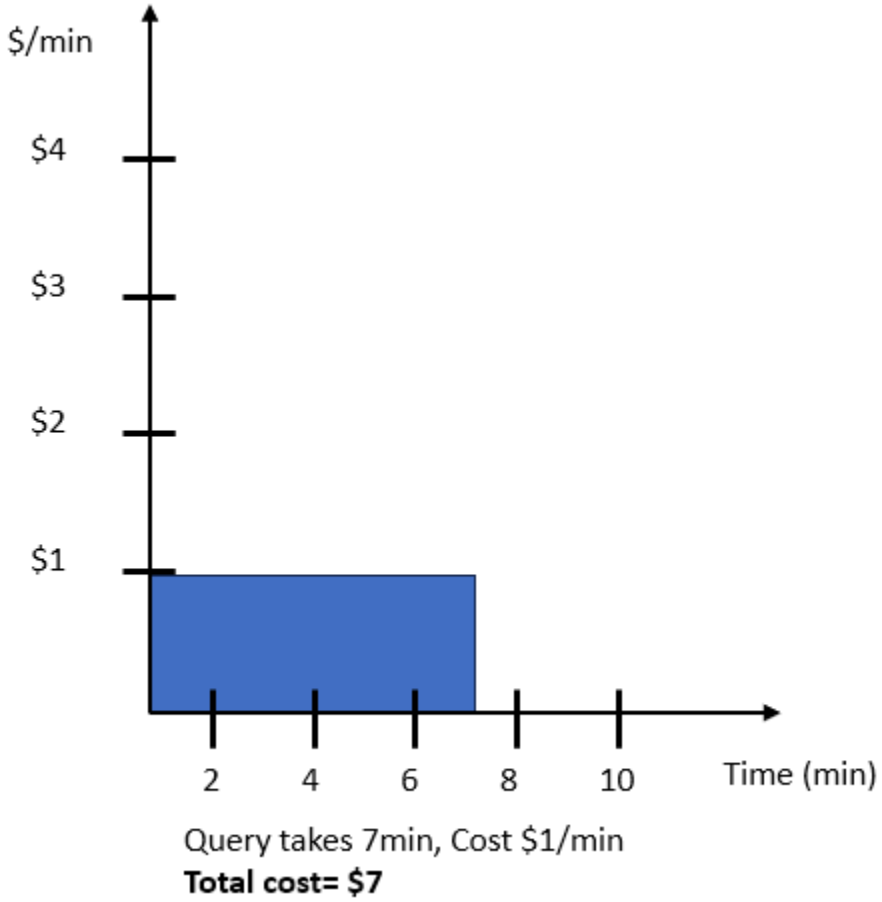
- [コストの最適化] - コスト削減を優先します。Amazon Redshift Serverless がコンピューティングキャパシティを自動的にスケールアップしようとするのは、そうすることでコストを削減できる可能性がある場合のみです。コンピューティングリソースのスケールダウンも試みてコストを削減し、クエリランタイムの向上も図ります。
- [バランスを取る] - パフォーマンスとコストのバランスを図ります。Amazon Redshift Serverless がパフォーマンスに配慮してスケールされる分、コストが若干増減する可能性があります。ほとんどの Amazon Redshift Serverless データウェアハウスでは、この設定が推奨されます。
- [パフォーマンスの最適化] - パフォーマンスを優先します。Amazon Redshift はパフォーマンス向上のために積極的にスケールするので、コスト増を招く場合があります。
- 中間位置: スライダーを [バランスを取る] と [コストの最適化] との中間位置、または [パフォーマンスの最適化] との中間位置に設定することもできます。コストまたはパフォーマンスの最適化に振り切るとやりすぎになる場合は、これらの設定を使用してください。

コストパフォーマンス目標を選択する際の考慮事項

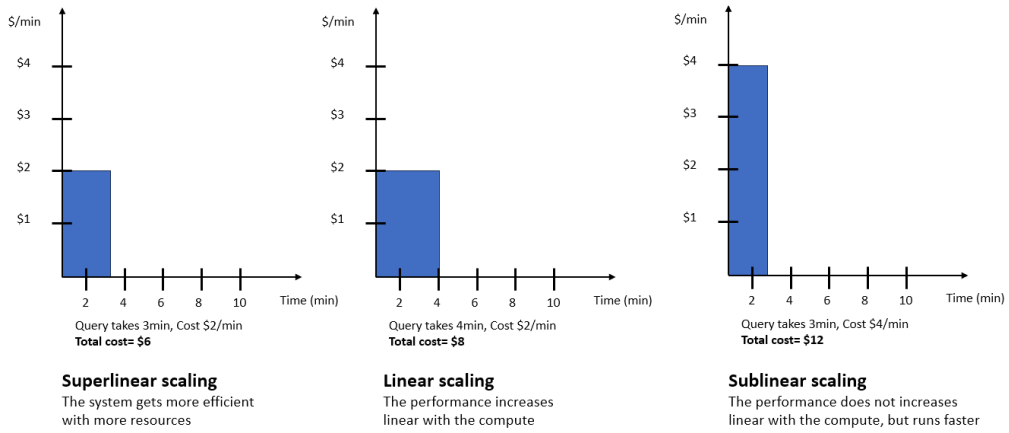
コストパフォーマンスのスライダーを使用して、ワークロードに必要なコストパフォーマンス目標を選択できます。AI 主導のスケーリングと最適化のアルゴリズムは、時間が経つにつれてワークロード履歴から学習するので、予測と決断の精度が向上します。

例

この例では、所要時間が 7 分、コストが 7 USD のクエリについて考えてみます。次の図は、スケーリングなしの場合のクエリのランタイムとコストを示しています。



次に示すように、クエリはいくつかの異なる方法でスケールされます。選択した料金パフォーマンス目標に基づいて、AI 主導のスケーリングと最適化は、クエリのパフォーマンスとコストのトレードオフを予測し、それに応じてスケールします。さまざまなスライダーオプションを選択した場合の結果は、次のようになります。



- [コストの最適化] - [コストの最適化] オプションを選択した場合、データウェアハウスはコスト削減の選択肢を優先してスケールします。前の例では、超線形スケーリングアプローチがこの動作を示しています。スケーリングは、スケーリングモデル予測に従って、費用対効果の高い方法で実行できる場合にのみ行われます。スケーリングモデルが、特定のワークロードに対してコスト最適化スケーリングができないと予測した場合、データウェアハウスはスケールしません。
- [バランスを取る] - [バランスを取る] を選択した場合、システムはコストとパフォーマンスの両方の考慮事項のバランスを取りながらスケールし、コストは限られた範囲で増加する場合があります。[バランスを取る] オプションは、超線形、線形、場合によっては劣線形のワークロードスケーリングを実行します。
- [パフォーマンスの最適化] - [パフォーマンスの最適化] オプションを選択した場合は、パフォーマンスを向上させる前述の方法に加えて、コストが高くなり、ランタイムが釣り合う形で改善されない場合でもスケールされます。[パフォーマンスの最適化] では、システムは可能であれば超線形スケーリング、線形スケーリング、劣線形スケーリングを実行します。スライダーの位置が [パフォーマンスの最適化] 位置に近いほど、Amazon Redshift Serverless で劣線形スケーリングが許容されやすくなります。

[コストパフォーマンス] スライダーを設定するときは、次の点に注意してください。

- コストパフォーマンスの設定はいつでも変更できますが、ワークロードのスケーリングはすぐには変更されません。スケーリングは、システムが現状のワークロードについて学習する従い、時間の経過とともに変化します。Serverless ワークグループを 1～3 日間モニタリングして、新しい設定の影響を確認することをお勧めします。
- コストパフォーマンススライダーのオプション [最大容量] と [最大 RPU 時間] は連携します。[最大容量] と [最大 RPU 時間] は、Amazon Redshift Serverless でデータウェアハウスがスケールできる最大 RPU と、Amazon Redshift Serverless でデータウェアハウスが消費できる最大 RPU 時間を制限するコントロールです。Amazon Redshift Serverless では、コストパフォーマンス目標の設定に関係なく、常にこれらの設定が尊重および適用されます。

リソースの自動スケーリングのモニタリング

AI 主導のスケーリングと最適化は、次の方法でモニタリングできます。

- Amazon Redshift コンソールで RPU の使用済み容量のグラフを確認します。
- CloudWatch で `AWS/Redshift-Serverless` および `Workgroup` の `ComputeCapacity` メトリクスをモニタリングします。

- SYS_QUERY_HISTORY ビューをクエリします。特定のクエリ ID またはクエリテキストを指定して、期間を特定します。この期間を使用して SYS_SERVERLESS_USAGE システムビューをクエリし、compute_capacity 値を見つけます。compute_capacity フィールドには、クエリランタイム中にスケールされた RPU が表示されます。

次の例を参考にして、SYS_QUERY_HISTORY ビューをクエリします。サンプル値を実際のクエリテキストに置き換えてください。

```
select query_id,query_text,start_time,end_time, elapsed_time/1000000.0
 duration_in_seconds
from sys_query_history
where query_text like '<query_text>'
and query_text not like '%sys_query_history%'
order by start_time desc
```

次のクエリを実行して、compute_capacity が start_time から end_time の期間中にどのようにスケールされたかを確認します。次のクエリの start_time と end_time を前述のクエリの出力に置き換えます。

```
select * from sys_serverless_usage
where end_time >= 'start_time'
and end_time <= DATEADD(minute,1,'end_time')
order by end_time asc
```

これらの機能を使用する具体的な手順については、「Configure monitoring, limits, and alarms in Amazon Redshift Serverless to keep costs predictable」を参照してください。

AI 主導のスケーリングと最適化を使用する際の考慮事項

AI 主導のスケーリングと最適化を使用する場合は、次の点を考慮してください。

- 8 ~ 512 のベース RPU を必要とする Amazon Redshift Serverless の既存のワークロードでは、最適な結果を得るために AI 主導のスケーリングと最適化を使用してください。この機能は、この RPU 範囲外のワークロードに使用することはお勧めしません。
- 料金パフォーマンス目標はワークロードを自動的に最適化しますが、その結果は異なる場合があります。システムが特定のパターンを学習できるように、代表的なワークロードを実行して、この機能をしばらく時間をかけて使用してください。

- AI 主導のスケーリングと最適化は、Amazon Redshift Serverless インスタンスで実行されているワークロードに基づいて、最適な時間を使用して Serverless ワークグループに最適化を適用します。

AI 主導のスケーリングと最適化の詳細については、次の動画をご覧ください。

Amazon Redshift Serverless での請求

コンピューティング性能に対する請求

Amazon Redshift Serverless の容量は、次の 2 つの方法で購入できます。

- オンデマンド容量を購入できます – オンデマンドコンピューティング容量を選択した場合、リソースは従量制料金となります。これは、Amazon Redshift Serverless を使い始めたばかりの場合や、安定した使用パターンがまだ明確でない場合に最適です。オンデマンドは、最も高い柔軟性を提供します。詳細については、「オンデマンドのコンピューティング容量に対する請求」を参照してください。
- 予約を購入できます – 特定の期間 (1 年間など)、事前設定された量のコンピューティングリソースを購入する場合、予約には割引があります。これは、一定量の容量を着実に使用することがわかっている場合に最適です。容量のニーズをある程度予測できる場合、コスト削減に役立ちます。詳細については、「サーバーレス予約に対する請求」を参照してください。

予約とオンデマンドリソースを一緒に使用できます。どちらか一方だけを使用する必要はありません。

料金の詳細については、「Amazon Redshift の料金」を参照してください。

オンデマンドのコンピューティング容量に対する請求

基本容量と課金への影響

クエリを実行すると、所定の期間に使用された容量に基づいて、秒単位の RPU 時間で請求されます。クエリが実行されていない場合、コンピューティング容量に対して請求されません。Redshift Managed Storage (RMS) についても、保存されているデータ量に基づいて課金されます。

ワークグループを作成するときに、コンピューティングの基本容量を設定するオプションを使用できます。ワークグループレベルでワークロードの料金/パフォーマンス要件を満たすには、既存

のワークグループの基本容量を増減して調整します。コンソールを使用して基本容量を変更するには、[ワークグループの設定] からワークグループを選択し、[制限] タブを選択します。

クエリ数が増えると、Amazon Redshift Serverless は自動的にスケールして一貫したパフォーマンスを保持します。

最大 RPU 時間の使用制限

Amazon Redshift Serverless のコストを予測できるようにしておくには、1 日、週、または月ごとに使用される [Maximum RPU hours] (RPU の最大時間) を設定します。これは、コンソールまたは API を使用して設定できます。制限に達した場合、ログエントリをシステムテーブルに書き込むか、アラートを受信するか、またはユーザークエリを無効にすることを指定できます。RPU の最大時間を設定すると、コストを管理することができます。RPU の最大時間の設定は、データウェアハウス内のデータにアクセスするクエリと、Amazon S3 内の外部テーブルなどの外部データにアクセスするクエリの両方について、ワークグループに適用されます。

以下に例を示します。

1 週間につき 100 時間の制限を設定すると仮定します。コンソールでこれを行うには、次の手順を実行します。

1. ワークグループを選択して、[制限] タブの [使用制限を管理] をクリックします。
2. 使用制限を追加して、[毎週] の頻度、[100] 時間の期間、アクションは [ユーザークエリを無効にする]] と選択します。

この例では、1 週間で 100 RPU 時間の制限に達すると、クエリは無効になります。

ワークグループに対して最大 RPU 時間を設定しても、ワークグループのパフォーマンスやコンピューティングリソースは制限されません。この設定は、クエリ処理への影響なしにいつでも調整できます。最大 RPU 時間を設定する目的は、コストとパフォーマンスの要件を満たすことです。サーバーレス料金の詳細については、Amazon Redshift の料金を参照してください。

Amazon Redshift Serverless のコストを予測可能な状態に維持するもう 1 つの方法には、AWS コスト異常検出を使用して、予期しない請求の発生可能性を低減し、コスト管理を向上することがあります。

> **Note**
>
> Amazon Redshift 料金計算ツールは料金の見積もりに役立ちます。必要なコンピューティングリソースを入力すると、コストのプレビューが表示されます。

コンピューティングリソースのコスト管理のための最大容量の設定

最大容量設定は、Amazon Redshift Serverless がスケールアップできる RPU の上限として機能します。これにより、コンピュートリソースのコスト制御につながります。ベース容量で利用可能なコンピューティングリソースの最小量を設定するのと同様に、最大容量で RPU 使用量の上限を設定します。これは、計画に沿った支出につながります。最大容量は各ワークグループ個別に適用され、コンピューティングの使用状況は常に制限されます。

最大容量と RPU 時間の使用制限との違い

最大 RPU 時間制限と最大容量はどちらもコストの制御に役立ちますが、機能する方法は異なります。違いは次の点で説明されるとおりです。

- 最大容量 – この設定は、Amazon Redshift Serverless が自動スケーリングに使用できる RPU の最大数を指定します。
- 最大 RPU 時間使用制限 – 最大容量とは異なり、この設定ではキャパシティの上限は設定されません。ただし、コスト抑制につながるその他のアクションが実行されます。ログへのエントリの追加、ユーザーへの通知、クエリの実行の停止などを選択できます。

最大容量のみを使用することも、最大 RPU 時間使用制限を超過した場合のアクションでこれを補完することもできます。

最大容量のユースケース

ワークグループごとに最大容量の設定が異なる場合があります。これにより、予算要件の強制に役立ちます。この仕組みを説明するために、次を想定します。

- ベース容量が 256 RPU に設定されたワークグループがあります。毎月のほとんどの期間、ワークロードは 256 RPU をわずかに超える安定した状態にあります。
- 最大容量は 512 RPU に設定されています。

アドホックな統計レポートを生成するために、3 日間にわたり、予想外に使用率が高くなったと仮定します。この場合、512 RPU を超えるコンピューティングコストを回避するために最大容量を設定しています。これにより、確実にコンピューティング容量がこの上限を超えないようにすることができます。

最大容量の使用に関する注意事項

最大容量を適切に設定するうえで、次の注意事項が役立ちます。

- Amazon Redshift Serverless ワークグループごとに異なる最大容量を設定できます。
- リソース使用率が非常に高い期間があり、最大容量が低い RPU レベルで設定されている場合、ワークロードの処理に遅延が発生し、最適なユーザーエクスペリエンスが得られない可能性があります。
- 一定期間に使用された容量は、Amazon Redshift Serverless ダッシュボードで確認できます。概要データの確認の詳細については、「ダッシュボードを使用して Amazon Redshift Serverless 概要データをチェックする」を参照してください。
- 通常、最大容量の変更によって実行中のクエリが中断されることはありません。ただし、最大容量を現在のベース容量未満に設定すると、Amazon Redshift Serverless は直ちに容量を減らします。これは、実行中のワークロードに影響する可能性があります。
- 最大容量の設定の上限は 5632 RPU です。

最大容量の設定方法

最大容量は、コンソールで設定できます。既存のワークグループについては、[ワークグループの設定] で設定を変更できます。CLI を使用して、次のサンプルのようなコマンドを使用して設定することもできます。

```
aws redshift-serverless update-workgroup --workgroup-name myworkgroup --max-capacity
 512
```

これにより、指定した名前のワークグループの最大容量設定を指定できます。設定を完了したら、コンソールで値を調べて検証できます。値は、CLI を使用して `get-workgroup` コマンドを実行しても確認できます。

最大容量設定を次のとおり `-1` に指定すると、無効にできます。

```
aws redshift-serverless update-workgroup --workgroup-name myworkgroup --max-capacity -1
```

Amazon Redshift Serverless の使用状況とコストのモニタリング

Amazon Redshift Serverless の使用量と請求額を見積もるには、複数の方法があります。システムビューは、クエリのための設定を特にしなくても、クエリデータや使用状況のデータなどのシステムメタデータがタイムリーに表示できます。CloudWatch は、Amazon Redshift Serverless の使用状況のモニタリングにも活用でき、インサイトを提供し、アクションを設定するための追加機能も搭載されています。

システムビューのクエリによる使用状況の可視化

SYS_SERVERLESS_USAGE システムテーブルをクエリして使用状況を追跡し、クエリの料金を取得します。

```
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
```

このクエリでは、Amazon Redshift Serverless で発生する 1 日あたりのコストを使用状況に基づいて提供します。

使用量とコストを決定するための使用に関する注意事項

- 実行するワークロードに対して秒単位の RPU 時間 (60 秒の最低料金) で課金されます。
- sys_serverless_usage システムテーブルのレコードには、1 分間隔で発生したコストが表示されます。以下の列について理解することが重要です。

  charged_seconds 列:

  - 時間間隔中に課金されたコンピュートユニット (RPU) 秒が提供されます。結果には、Amazon Redshift Serverless の最低料金が含まれます。
  - トランザクション完了後の compute-resource の使用量に関する情報があります。したがって、トランザクションが完了していない場合、この列の値は 0 になります。

  compute_seconds 列:

  - コンピューティングの使用状況情報をリアルタイムで提供します。Amazon Redshift Serverless の最低料金は含まれていません。そのため、インターバル中に請求される請求秒数とはある程度異なる場合があります。

- (トランザクションが終了していない場合でも) 各トランザクション中の使用状況情報を表示するため、提供されるデータはリアルタイムで表示されます。
- compute_seconds が 0 であるが charged_seconds が 0 より大きい場合も、その逆の場合もあります。これはシステムビューでのデータの記録方法によって生じる通常の動作です。サーバーレスの使用状況の詳細をより正確に表すには、SYS_SERVERLESS_USAGE のデータを集計することをお勧めします。

テーブルとビューのモニタリングの詳細については、「Amazon Redshift Serverless でのクエリとワークロードのモニタリング」を参照してください。

CloudWatch を使用した使用状況の可視化

CloudWatch で利用可能なメトリクスを使用して、使用状況を追跡できます。CloudWatch 用に生成されるメトリックスには、現在時刻で使用された RPU の総秒数を示す `ComputeSeconds`、その時点の合計のコンピューティング性能を示す `ComputeCapacity` があります。使用量のメトリクスは、Redshift Serverless dashboard の Redshift コンソールでも検索できます。CloudWatch の詳細については、「Amazon CloudWatch とは」を参照してください。

サーバーレス予約に対する請求

Amazon Redshift Serverless では、従量制料金モデルでクラスターをプロビジョニングおよび管理することなく、分析を実行およびスケールできます。サーバーレス予約を使用すると、コンピューティングコストをさらに最適化し、Redshift Serverless の既存および新規のワークロードのコスト予測可能性を向上させることができます。

Amazon Redshift は、AWS 支払者アカウントレベルでサーバーレス予約を管理し、複数の AWS アカウント間で予約を共有できるため、AWS アカウント内のすべての Redshift Serverless ワークロードでコンピューティングコストを最大 24% 削減できます。Amazon Redshift はサーバーレス予約を 1 時間ごとに請求し、1 秒あたりの予約数を計測して、Redshift Serverless が提供する柔軟性を維持しながら、24 時間 365 日、一貫した請求モデルを提供します。Amazon Redshift は、指定された RPU レベルを超える使用量に対して標準のオンデマンドレートで課金します。

> **Note**
>
> オンデマンド使用量を制限する場合は、[最大容量] 設定を使用して、ワークグループのリソース使用量の制限を設定できます。詳細については、「Amazon Redshift Serverless での請求」を参照してください。

サーバーレス予約の利点

サーバーレス予約は、Amazon Redshift Serverless の割引料金オプションです。サーバーレス予約では、1 年間、前払いなしのオンデマンド (OD) 料金の割引により、指定された数の Redshift 処理ユニット (RPU) にコミットできます。前払いにより、割引率が高くなります。サーバーレス予約を使用すると、コンピューティングコストを最適化し、Serverless の既存および新規のワークロードのコスト予測可能性を向上させることができます。

各サーバーレス予約は、AWS アカウントレベルで購入し、同じ支払者アカウント内の複数の Amazon Redshift Serverless ワークグループ間で共有できます。これにより、割引が柔軟に適用されます。ワークロードパターンが異なる複数のワークグループ間で予約を共有できます。

サーバーレス予約の仕組み

RPU の予約は、ほんの数分で完了する簡単なプロセスです。これには、予約する RPU レベルと支払いタイプの指定が含まれます。Amazon Redshift Serverless で使用している標準の AWS 請求およびコスト管理ツールは、必要な予約レベルを決定し、使用状況を継続的にモニタリングするのに役立ちます。サーバーレス予約は AWS 支払者アカウントレベルで管理され、同じ支払者アカウントで共有できるため、AWS アカウント内のすべての Redshift Serverless ワークロードでコンピューティングコストを最大 24% 削減できます。サーバーレス予約は、1 時間ごとの請求と 1 秒ごとの計測により、Redshift Serverless が提供する柔軟性を維持しながら、24 時間 365 日、一貫した請求モデルを提供します。指定された RPU レベルを超える使用量は、標準の Redshift Serverless オンデマンド料金で課金されます。

同じ AWS アカウント内で複数のサーバーレス予約を購入できます。追加で購入したサーバーレス予約は、前の購入と合わせて考慮されます。例えば、2 つの予約を購入し、各予約で 100 RPU を選択した場合、合計 200 RPU が割引料金で提供されます。

> **Note**
>
> オンデマンド使用量に制限を設定する場合は、[制限] タブ、[使用制限を管理] の順に選択して、Amazon Redshift Serverless コンソールでワークグループの最大 RPU 数を設定できます。

サーバーレス予約を購入すると、予約はすぐに有効になり、Redshift コンソールのサーバーレス予約ダッシュボードに表示されます。

RPU (Redshift 処理ユニット) の使用を分析して必要な予約レベルを決定する

Redshift Serverless 予約では、特定の数の Redshift 処理ユニット (RPU) に 1 年間コミットすることで、予測可能で低いコンピューティングコストを確保できるため、オンデマンド料金よりも割引率が高くなります。前払いなしオプションの場合は最大 20%、全額前払いの場合は最大 24% の割引となります。Redshift Serverless 予約は、AWS 支払いアカウントレベルで購入します。割引は AWS にリンクされたすべてのアカウントの Redshift Serverless ワークグループに自動的に適用されるため、複数のチームをサポートしながら予算を一元管理できます。Redshift Serverless は、使用量を秒単位の詳細度で計測し、各時間で平均して時間単位で請求するため、使用した容量に対してのみ料金が発生します。Redshift Serverless 予約は、アカウント間の柔軟な適用と期間ベースの節約を組み合わせることで、Redshift Serverless の俊敏性を犠牲にすることなく、予測可能な分析料金を実現します。

予約の RPU 使用状況の分析

RPU の使用状況は、Redshift Serverless ダッシュボードでの 7 日間表示と、Cost Explorer を使用した長期分析の 2 つの方法のいずれかで確認できます。次の手順は、RPU の使用状況を分析する方法を示しています。

方法 1: Redshift Serverless ダッシュボード (7 日間表示)

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. Serverless ダッシュボードを開きます。
3. ワークグループを選択します。
4. 過去 1 時間から 1 週間までの RPU 容量の使用状況を表示します。

方法 2: AWS Cost Explorer (長期分析)

1. AWS マネジメントコンソールにサインインし、Cost Explorer コンソール (https://console.aws.amazon.com/costmanagement/) を開きます。
2. [詳細度] を[時間単位] に設定します。
3. [使用タイプ] 別にグループ化します。
4. 以下のフィルタを使用できます。

   - サービス: Redshift
   - リージョン: ローカルリージョン

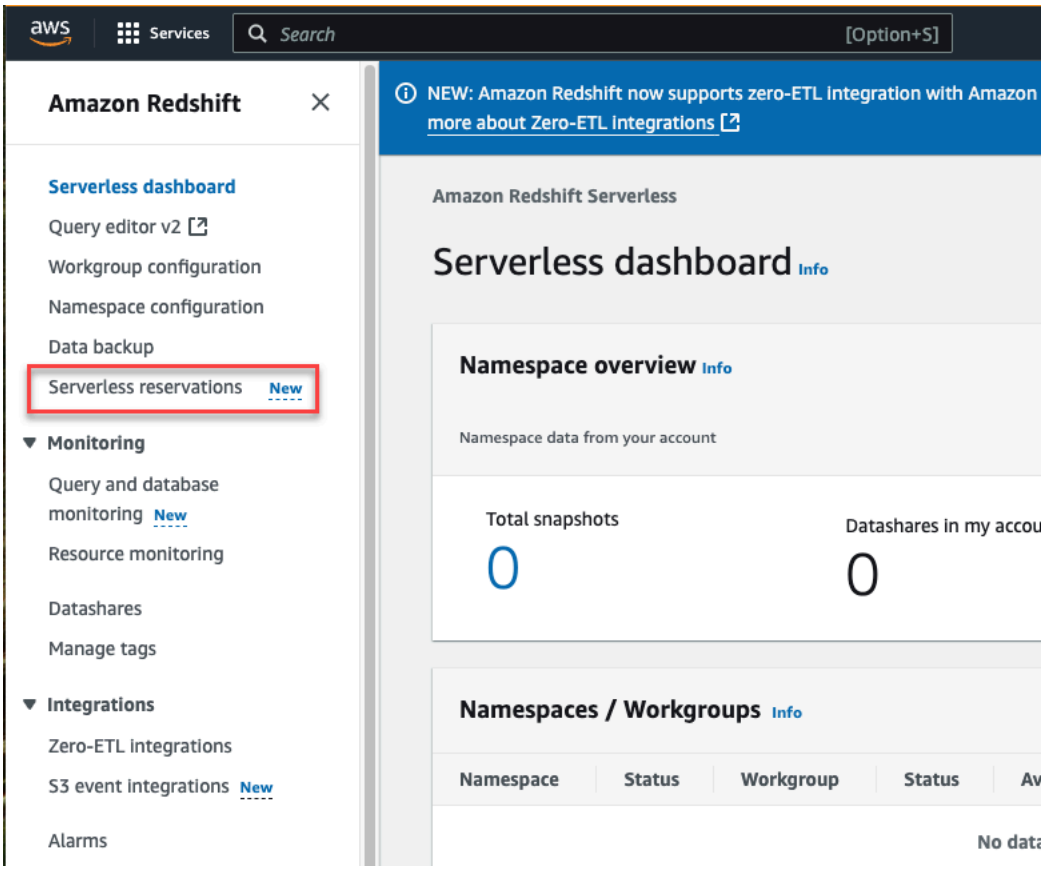
- 使用タイプ: Redshift:ServerlessUsage のフィルター

5. 選択したリージョンにおける時間単位のサーバーレス使用状況に関するコストと使用状況グラフを確認します。

コンソールを使用したサーバーレス予約の購入

予約を購入するときは、割引対象となる RPU レベルを選択します。RPU レベルを選択する前に、ベース容量と時間の経過とともに使用するオンデマンド容量を知っておくことが役立ちます。このセクションでは、容量を決定し、サーバーレス予約を予約する方法について説明します。

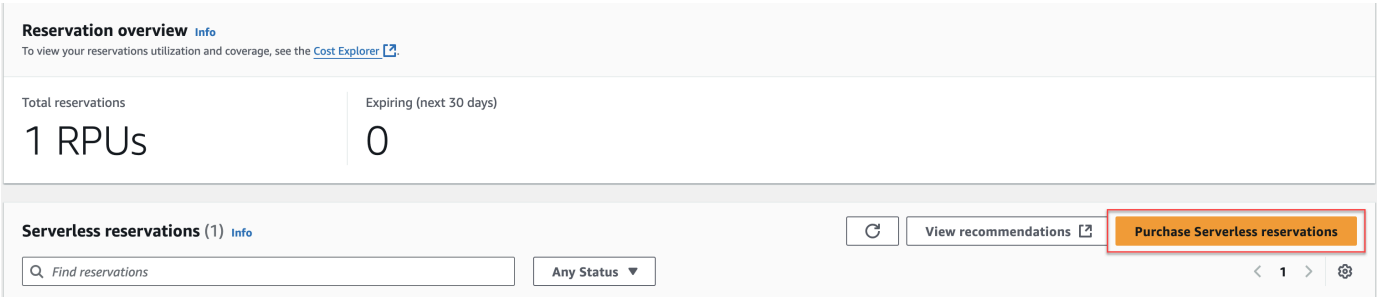
開始するには、Redshift コンソールで [サーバーレス] を選択し、メニューから [サーバーレス予約] を選択します。



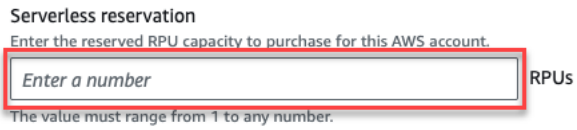
コンソールには、機能の説明と既存の予約のリストが表示されます。ここから予約を購入するか、レポートとモニタリングツールを使用して現在の使用状況を確認できます。これらは、RPU レベルと予約に適した RPU 数を決定するのに役立ちます。

予約を購入するには、次の手順を実行します。

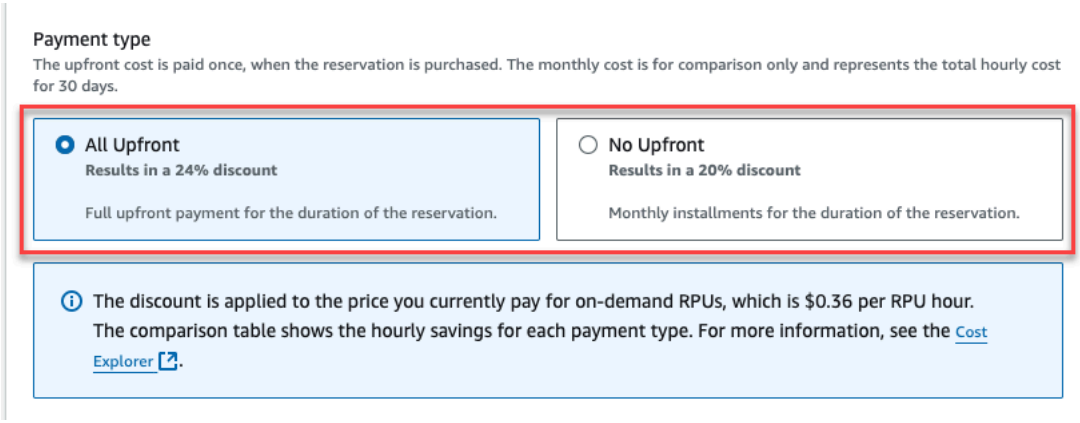
1. [サーバーレス予約を購入] を選択します。



2. ウォークスルーが表示され、一連の選択肢が示されます。[サーバーレス予約] に予約する RPU レベルを入力します。このレベルが不明な場合は、このセクションの後で説明するツールを使用できます。

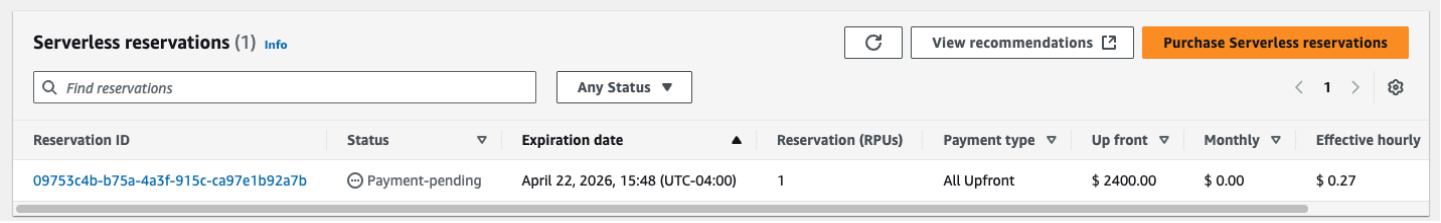


3. 支払いタイプを設定します。予約した RPU の前払い、または月払いを選択できます。前払いを選択すると、割引率が高くなります。



4. 選択が完了したら、[サーバーレス予約を購入]、[確認] の順に選択します。

予約を確認すると、予約のリストに表示されます。



使用に関する注意事項

- 予約を変更または削除することはできません。ただし、追加の予約を作成してカバー範囲を広げることができます。
- Redshift Serverless は、ワークロードの予約した RPU を使用してからオンデマンド RPU を使用することで、コストを削減します。予約した RPU 数を超えると、超えた分の RPU に対して、Redshift Serverless オンデマンド料金で課金されます。
- Amazon Redshift Serverless の無料クレジットは、サーバーレス予約には適用されず、オンデマンドで請求される RPU にのみ適用されます。

サーバーレス予約の例

このシナリオでは、AWS 支払者/連結アカウントに 2 つの Amazon Redshift ワークグループがあります。

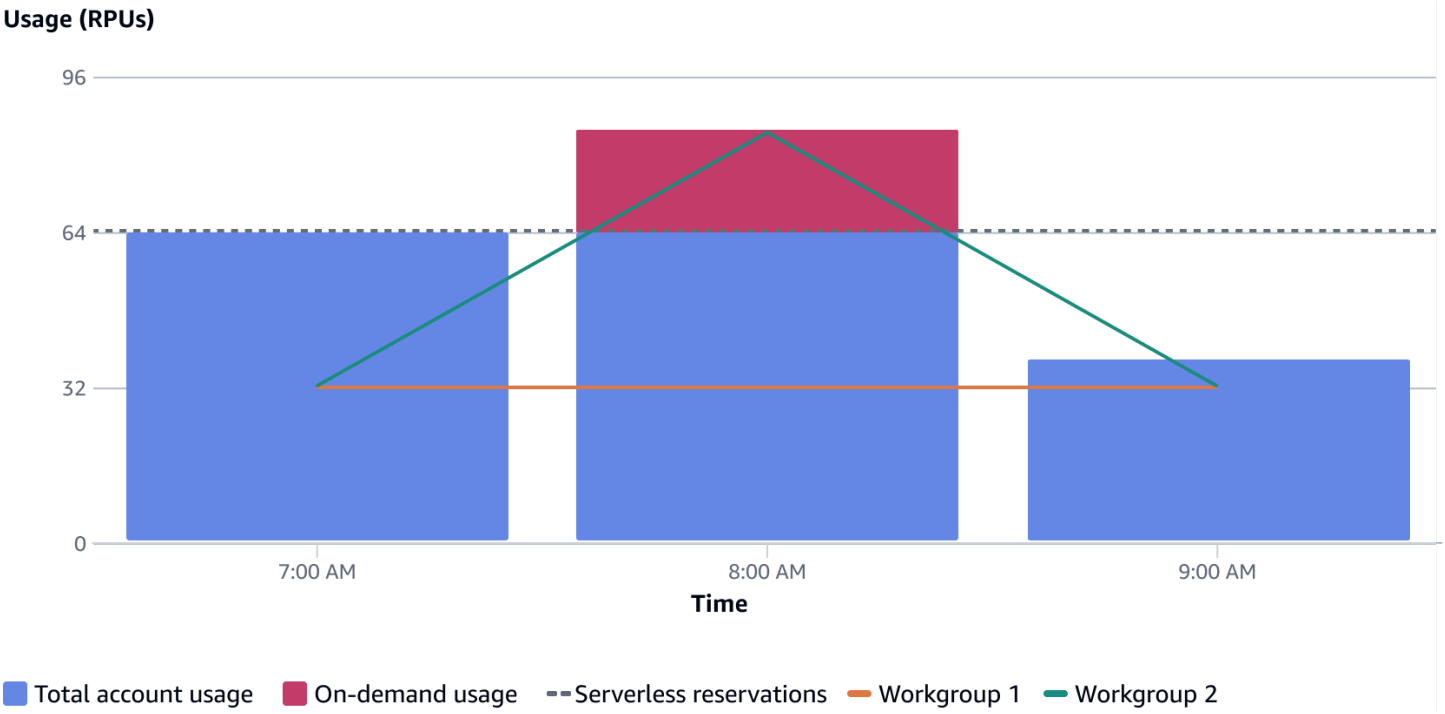
- ワークグループ 1 では、ビジネスインテリジェンスチームの場合など、使用量が定常状態です。
- ワークグループ 2 では、ETL オペレーションなど、ワークロードが予測不可能で、使用量が急増する場合があります。

これらのワークグループのコストを最適化するために、1 年間のサーバーレス予約を購入します。履歴データに基づいて、両方のワークグループは定常状態で 64 RPU を消費すると判断します。ただし、ワークグループ 2 は、短期間に 32 RPU から 48 RPU に増加したり、24 RPU に減少したりすることがあります。予約の RPU レベルを、過去の傾向に合わせて 64 RPU に設定して開始します。時間ごとの請求の詳細は次のとおりです。

- 最初の 1 時間は、過去の使用状況の傾向と同様に、両方のワークグループが 32 RPU を使用し、アカウントの合計使用量が 64 RPU になります。この 1 時間では、すべての RPU がサーバーレス予約の割引料金で課金されます。64 RPU の使用レベルが 64 RPU のサーバーレス予約と等しいためです。
- 2 時間目で、ワークグループ 1 は引き続き 32 RPU を使用します。一方、ワークグループ 2 は 48 RPU に急増し、アカウントの合計使用量は 80 RPU になります。この 1 時間では、64 RPU がサーバーレス予約の割引料金で課金され、16 RPU が Redshift Serverless オンデマンド料金で課金されます。

- 3 時間目で、ワークグループ 1 は引き続き 32 RPU を消費し、ワークグループ 2 は 8 RPU に減少します。この 1 時間では、アカウントの合計が 40 RPU であっても、64 RPU のサーバーレス予約料金がアカウントに課金されます。

ワークグループの使用状況の推移と、オンデマンドおよびサーバーレス予約料金の請求の詳細については、次の図を参照してください。



AWS CLI または Amazon Redshift API を使用したサーバーレス予約の購入

`create-reservation` を使用して RPU 予約を作成します。このコマンドを以下に示します。

```
create-reservation
--capacity
--offering-id
```

`capacity` は、予約する RPU 数に設定します。

ストレージの請求

プライマリストレージ容量は、Redshift マネージドストレージ (RMS) として請求されます。ストレージは GB /月単位で請求されます。ストレージの請求は、コンピューティング能力の請求とは別に行われます。手動スナップショットに使用されるストレージは、すべてのアクティブな手動スナッ

プショットの一意のデータブロックの合計に基づいて、標準のバックアップ請求レートで請求されます。保持時間が 24 時間未満の復旧ポイントは請求されません。

データ転送と機械学習 (ML) の料金は、プロビジョニングされたクラスターと同じように、別々に適用されます。AWS リージョン間のスナップショットレプリケーションとデータ共有は、料金ページに記載されている概略の転送レートで請求されます。詳細については、「Amazon Redshift の料金」を参照してください。

CloudWatch を使用した使用料請求の可視化

スナップショットストレージの使用状況を追跡するメトリクス `ServerlessUniqueSnapshotStorage` が生成され、CloudWatch に送信されます。CloudWatch の詳細については、「Amazon CloudWatch とは」を参照してください。

Amazon Redshift Serverless 無料トライアルの利用

Amazon Redshift Serverless は無料トライアルを提供しています。無料トライアルに参加する場合は、Redshift コンソールで無料トライアルのクレジット残高を表示し、SYS_SERVERLESS_USAGE システムビューで無料トライアルの使用を確認できます。無料トライアルの使用に関する請求の詳細は、請求コンソールに表示されないことに注意してください。使用状況は、無料トライアルの終了後にのみ、請求コンソールで確認できます。Amazon Redshift Serverless 無料トライアルの詳細については、「Amazon Redshift Serverless 無料トライアル」を参照してください。

使用料の請求についての注記

- 使用量の記録 - クエリやトランザクションは、トランザクションの完了、ロールバック、停止後にのみ測定、記録されます。例えば、トランザクションが 2 日間実行された場合、RPU の使用量はトランザクションの完了後に記録されます。`sys_serverless_usage` のクエリを実行することで、使用中の状況をリアルタイムでモニタリングできます。トランザクション記録は、RPU 使用量の変動として反映され、特定の時間や毎日の使用のコストに影響を与える可能性があります。
- 明示的なトランザクションを記述する - トランザクションを終了することは、重要な役割を果たすベストプラクティスです。開いているトランザクションを終了またはロールバックしない場合、Amazon Redshift Serverless は RPU を使用し続けます。例えば、`BEGIN TRAN`を明示的に記述する場合、それに対応した `COMMIT` や `ROLLBACK` の記述があることが重要です。
- クエリのキャンセル - クエリを実行し、終了する前にキャンセルした場合は、クエリの実行時間に対して請求されます。

- スケーリング - Amazon Redshift Serverless インスタンスは、パフォーマンスを一定に保つためにスケーリングを開始して、負荷の高い時間に対応する場合があります。Amazon Redshift Serverless の請求には、同じ RPU レートでの基本のコンピューティング性能とスケール処理容量の両方が含まれます。
- スケールダウン - Amazon Redshift Serverless は、基本 RPU 容量からスケールアップして、負荷の高い時間を処理します。場合によっては、クエリの負荷が低下した後も、RPU 容量が一定期間高い設定のままになる場合があります。予想外のコストを防ぐため、コンソールで最大 RPU 時間を設定することを推奨します。
- システムテーブル - システムテーブルのクエリを実行すると、クエリ時間が請求されます。
- Redshift Spectrum - Amazon Redshift Serverless があり、クエリを実行する場合、データレイククエリには別途の料金は発生しません。Amazon S3 に保存されているデータに対するクエリの場合は、トランザクション時間ごとに、ローカルデータに対するクエリと同じ料金になります。
- フェデレーティッドクエリ - フェデレーションクエリは、データウェアハウスまたはデータレイクでのクエリと同様に、特定の時間で使用される RPU に基づいて課金されます。
- ストレージ - ストレージは GB/月単位で別途請求されます。
- 最低料金 – リソース使用量は最低 60 秒として課金され、1 秒単位で計測されます。
- スナップショットの請求 - スナップショットの請求は変更されません。スナップショットの請求は、GB /月のレートで請求されるストレージに応じて課金されます。データウェアハウスを過去 30 分単位で、24 時間内の特定のポイントに、無料で復元できます。詳細については、Amazon Redshift の料金を参照してください。
- 自動最適化は、追加のコンピューティングリソースを使用して実行されます - Amazon Redshift Serverless は通常、ユーザークエリと共に自動最適化オペレーションを実行します。これらのオペレーションは自律管理と呼ばれ、料金は発生しません。

  追加のコンピューティングリソースの割り当てを有効にすると、Amazon Redshift はユーザーアクティビティが高い期間でも、必要に応じて自律管理を実行します。このような場合は、自律管理の実行に費やされた時間に対して課金されます。詳細については、「Amazon Redshift データベースデベロッパーガイド」の「データベースの自動最適化のための追加のコンピューティングリソースの割り当て」を参照してください。

請求を予測可能にするための Amazon Redshift Serverless のベストプラクティス

一貫した請求を維持するために役立つベストプラクティスと組み込みの設定を以下に示します。

- 各トランザクションを必ず終了してください。BEGIN を使ってトランザクションを開始する場合、それを END することも重要です。
- また、エラー処理のベストプラクティスに従ってエラーに適切に対応し、各トランザクションを終了します。オープントランザクションを最小化することで、不要な RPU の使用を回避できます。
- SESSION TIMEOUT を使用して、開いているトランザクションとアイドルセッションを終了します。これにより、3600 秒 (1 時間) を超えてアイドル状態または非アクティブの状態のセッションがすべてタイムアウトします。これにより、21600 秒 (6 時間) を超えて開いたままの状態または非アクティブの状態のトランザクションがすべてタイムアウトします。このタイムアウト設定は、長時間実行されるクエリに対してセッションを開いたままにする場合など、特定のユーザーに対して明示的に変更できます。CREATE USER のトピックに、ユーザーに対して SESSION TIMEOUT を調整する方法が記載されています。
  - SESSION TIMEOUT の値は、ユースケースで具体的に求められていないかぎり、ほとんどの場合では、延長しないことを推奨します。開いているトランザクションでセッションがアイドル状態のままの場合、セッションが閉じられるまで RPU が使用されることがあります。これによって不要なコストが発生します。
  - Amazon Redshift Serverless では、クエリの実行時間が最大 86,399秒 (24時間) です。Amazon Redshift Serverless がトランザクションに関連付けられたセッションを終了するまでの、オープントランザクションのアイドル状態の最大期間は 6 時間です。詳細については、「Amazon Redshift Serverless オブジェクトのクォータ」を参照してください。

接続プーリングを使用した Amazon Redshift Serverless の請求

Amazon Redshift Serverless は、接続プールによって送信される軽量ヘルスチェッククエリを含む、すべての受信クエリを請求可能なユーザーアクティビティとして扱います。この動作は、クエリのソースがアプリケーション、JDBC/ODBC ドライバー、または接続プーリングフレームワークのいずれであるかに関係なく適用されます。各ヘルスチェッククエリはコンピューティング使用量をトリガーし、クエリの目的やオリジンに関係なく料金が発生します。その結果、オープン接続プールを維持することで、実際のユーザーワークロードが実行されていなくてもコストが発生する可能性があります。

接続プーリングは、アプリケーションと Amazon Redshift Serverless エンドポイント間の永続的な接続のプールを維持します。これらの接続が正常で利用可能であることを確認するために、プーリングメカニズムは多くの場合、軽量または空のクエリ (例: SELECT 1) を定期的に送信します。これらの自動クエリは、接続ステータスを検証します。

接続プーリングを使用する場合は、意図しない料金を最小限に抑えるために、次のベストプラクティスを考慮してください。

- 接続プーリング設定でヘルスチェックまたはキープアライブクエリの頻度を確認して最適化することで、ヘルスチェックの頻度を調整します。
- システムのアイドル時間中に不要な接続チャーンやバックグラウンドクエリアクティビティを最小限に抑えるように接続プーリングを設定して、アイドルシステム設定を最適化します。
- オーバーヘッドを削減できる場合は、アプリケーションレベルのプーリングを実装するか、接続ライフサイクル管理を改善します。
- 接続プーリング設定で許可されている場合は、ハートビートクエリまたは検証クエリを無効にします。特定の接続文字列パラメータまたは設定ファイルをチェックして、これらの設定を調整します。
- TCP キープアライブ設定のファインチューニング: ドライバーの内部ハートビートメカニズムを無効にすることができない場合は、オペレーティングシステムまたはアプリケーションレベルで Transmission Control Protocol (TCP) キープアライブ設定を調整して、接続タイムアウトの問題に対処します。詳細については、オペレーティングシステム、JDBC/ODBC ドライバー、または接続プールのドキュメントを参照してください。
- データベース接続プーリングの最適化: 接続を管理し、接続オーバーヘッドを最小限に抑えるように接続プール (HikariCP、Apache データベース接続プール) を設定します。最大接続数、アイドルタイムアウト、検証クエリ (必要な場合) などのパラメータに焦点を当てます。この最適化により、Amazon Redshift Serverless のコンピューティング使用量を実際のワークロード需要に合わせて調整できるため、コスト削減につながる可能性があります。

ゼロ ETL を使用した Amazon Redshift Serverless のコスト最適化

Amazon Redshift Serverless でゼロ ETL 統合を実行しながらコストを最適化するために、環境のサイズを適切に設定し、ワークロードのニーズに基づいて更新設定を調整できます。以下の調整を検討してください。

- ワークロードで使用可能な場合は、8 RPU のより低い基本 RPU 容量を使用します。
- ターゲット Redshift インスタンスの REFRESH_INTERVAL を設定し、鮮度とコストのバランスを取ることができます。間隔を短くすると、ほぼリアルタイムの更新が確保されますが、計算コストが増加します。間隔を長くすると (5 分以上)、レポートや履歴分析など、即時の鮮度が重要ではないワークロードの料金が削減されます。Redshift ターゲット REFRESH_INTERVAL を編集するには、ALTER DATABASE の説明の更新間隔句を参照してください。

- ゼロ ETL データの取り込み中に分析ワークロードを同時に実行することで、Amazon Redshift Serverless 環境の使用率を最大化します。これにより、コンピューティング性能が複数のビジネス目的にアクティブに対応できるようになります。

Amazon Redshift Serverless への接続

Amazon Redshift Serverless インスタンスをセットアップしたら、以下に示すさまざまな方法で接続できます。チームやプロジェクトが複数存在し、それらのコストを別々に管理したい場合は、個別に AWS アカウント を使用できます。

Amazon Redshift Serverless が利用できる AWS リージョン の一覧については、「Amazon Web Services 全般のリファレンス」の「Redshift Serverless API」に記載されているエンドポイントを参照してください。

Amazon Redshift Serverless は、現在の AWS リージョン で AWS アカウント 内のサーバーレス環境に接続します。Amazon Redshift Serverless は、5431 ~ 5455 と 8191 ~ 8215 のポート範囲内の VPC で実行されます。デフォルトは 5439 です。現在、ポートを変更できるのは API オペレーション `UpdateWorkgroup` と AWS CLI オペレーション `update-workgroup` のみです。

Amazon Redshift Serverless への接続

Amazon Redshift Serverless で次の構文を使ってデータベース (dev という名前) に接続できます。

```
workgroup-name.account-number.aws-region.redshift-serverless.amazonaws.com:port/dev
```

例えば、以下の接続文字列では、リージョンに us-east-1 が指定されます。

```
default.123456789012.us-east-1.redshift-serverless.amazonaws.com:5439/dev
```

JDBC ドライバー経由での Amazon Redshift Serverless への接続

次のいずれかの方法で、好みの SQL クライアントから、Amazon RedShift が提供する JDBC ドライバーバージョン 2.x を使用して Amazon Redshift Serverless に接続できます。

JDBC ドライバーバージョン 2.x を使用して、データベース認証のサインイン認証情報で接続するには、次の構文を使用します。ポート番号はオプションです。含まれていない場合、Amazon Redshift Serverless のデフォルトのポート番号は 5439 です。5431-5455 または 8191-8215 のポート範囲から別のポートに変更できます。サーバーレスエンドポイントのデフォルトのポートを変更するには、AWS CLI と Amazon Redshift API を使用します。

```
jdbc:redshift://workgroup-name.account-number.aws-region.redshift-
serverless.amazonaws.com:5439/dev
```

例えば、次の接続文字列では、デフォルトのワークグループ、アカウント ID 123456789012、リージョン us-east-2 を指定しています。

```
jdbc:redshift://default.123456789012.us-east-2.redshift-serverless.amazonaws.com:5439/
dev
```

JDBC ドライバーバージョン 2.x を使用して IAM に接続するには、次の構文を使用します。ポート番号はオプションです。含まれていない場合、Amazon Redshift Serverless のデフォルトのポート番号は 5439 です。5431-5455 または 8191-8215 のポート範囲から別のポートに変更できます。サーバーレスエンドポイントのデフォルトのポートを変更するには、AWS CLI と Amazon Redshift API を使用します。

```
jdbc:redshift:iam://workgroup-name.account-number.aws-region.redshift-
serverless.amazonaws.com:5439/dev
```

例えば、次の接続文字列では、デフォルトのワークグループ、アカウント ID 123456789012、リージョン us-east-2 を指定しています。

```
jdbc:redshift:iam://default.123456789012.us-east-2.redshift-
serverless.amazonaws.com:5439/dev
```

ODBC には、次の構文を使用します。

```
Driver={Amazon Redshift (x64)}; Server=workgroup-name.account-number.aws-
region.redshift-serverless.amazonaws.com; Database=dev
```

バージョン 2.1.0.9 以前の JDBC ドライバーを使用して IAM に接続する場合は、次の構文を使用する必要があります。

```
jdbc:redshift:iam://redshift-serverless-<name>:aws-region/database-name
```

例えば、次の接続文字列では、デフォルトのワークグループとリージョン AWS リージョン us-east-1 を指定しています。

```
jdbc:redshift:iam://redshift-serverless-default:us-east-1/dev
```

ドライバーの詳細については、「Amazon Redshift での接続の設定」を参照してください。

JDBC と ODBC の接続文字列の検索

SQL クライアントツールでワークグループに接続するには、JDBC または ODBC 接続文字列が必要です。接続文字列は、Amazon Redshift Serverless コンソールの、ワークグループの詳細ページにあります。

ワークグループの接続文字列を見つけるには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Redshift Serverless] を選択します。
3. ナビゲーションメニューで、[ワークグループの設定] を選択してから、リストからワークグループ名を選択し、その詳細を開きます。
4. JDBC URL および ODBC URL 接続文字列は、追加の詳細とともに、[General information (一般情報)] セクションで手に入れることができます。各文字列はワークグループが実行される AWS リージョンに基づき決まります。適切な接続文字列の横にあるアイコンを選択して接続文字列をコピーします。

Data APIを使用した Amazon Redshift Serverless への接続

Amazon Redshift Data API を使用して、Amazon Redshift Serverless に接続することもできます。AWS CLI の呼び出しでは、`cluster-identifier` パラメータではなく、`workgroup-name` パラメータを使用します。

Data API の詳細については、「Amazon Redshift Data API の使用」を参照してください。Python で Data API を呼び出すコードの例やその他の例については、「Redshift Data API の開始方法」と、`quick-start` フォルダおよび `use-cases` フォルダを GitHub で参照してください。

SSL での Amazon Redshift Serverless への接続

Amazon Redshift Serverless への安全な接続を設定する

SSL 接続をサポートするために、Redshift Serverless では、AWS Certificate Manager(ACM) によって発行された SSL 証明書をワークグループごとに作成してインストールします。ACM 証明書は、ほとんどのオペレーションシステム、ウェブブラウザ、およびクライアントから公的に信頼されています。SQL クライアントまたはアプリケーションが、`sslmode`接続オプションを`require`、`verify-`

ca、またはverify-fullに設定した SSL を使用して Redshift Serverless に接続する場合は、証明書バンドルをダウンロードする必要があるかもしれません。クライアントが証明書を必要とする場合、Redshift Serverless は以下のようにバンドル証明書を提供します。

- https://s3.amazonaws.com/redshift-downloads/amazon-trust-ca-bundle.crtからバンドルをダウンロードします。
  - 予想される MD5 チェックサム番号は 418dea9b6d5d5de7a8f1ac42e164cdcf です。
  - sha256 のチェックサム番号は 36dba8e4b8041cd14b9d60158893963301bcb92e1c456847784de2acb5bd550 です。

  `https://s3.amazonaws.com/redshift-downloads/redshift-ca-bundle.crt`にあった以前の証明書バンドルは使用しないでください。
- 中国の AWS リージョンでは、https://s3.cn-north-1.amazonaws.com.cn/redshift-downloads-cn/amazon-trust-ca-bundle.crt からバンドルをダウンロードします。
  - 予想される MD5 チェックサム番号は 418dea9b6d5d5de7a8f1ac42e164cdcf です。
  - sha256 のチェックサム番号は 36dba8e4b8041cd14b9d60158893963301bcb92e1c456847784de2acb5bd550 です。

  `https://s3.cn-north-1.amazonaws.com.cn/redshift-downloads-cn/redshift-ca-bundle.crt`および`https://s3.cn-north-1.amazonaws.com.cn/redshift-downloads-cn/redshift-ssl-ca-cert.pem`にあった以前の証明書バンドルは使用しないでください。

> **Important**
>
> Redshift Serverless は、SSL 証明書の管理方法を変更しました。引き続き SSL でワークグループに接続するには、現在の信頼ルート CA 証明書を更新する必要があります。SSL 接続用 ACM 証明書の詳細については、「SSL 接続用 ACM 証明書への移行」を参照してください。

デフォルトでは、ワークグループデータベースは SSL の使用の有無にかかわらず接続を受け入れます。

SSL 接続のみを受け入れる新しいワークグループを作成するには、 `create-workgroup` コマンドを使用して `require_ssl`パラメータを `true` に設定します。次の例を使用するには、*`yourNamespaceName`* を名前空間の名前に置き換え、*`yourWorkgroupName`* をワークグループの名前に置き換えます。

```
aws redshift-serverless create-workgroup \
--namespace-name yourNamespaceName \
--workgroup-name yourWorkgroupName \
--config-parameters parameterKey=require_ssl,parameterValue=true
```

既存のワークグループを更新して SSL 接続のみを受け入れるには、 update-workgroup コマンドを使用して require_sslパラメータを true に設定します。require_ssl パラメータを更新すると、Redshift Serverless によってワークグループが再起動されることに注意してください。次の例では、*yourWorkgroupName* をワークグループの名前に置き換えます。

```
aws redshift-serverless update-workgroup \
--workgroup-name yourWorkgroupName \
--config-parameters parameterKey=require_ssl,parameterValue=true
```

Amazon Redshift では、Elliptic Curve Diffie—Hellman Ephemeral (ECDHE) 鍵契約プロトコルがサポートされます。ECDHE を使用すると、クライアントとサーバーはそれぞれ、安全でないチャネルを介して共有シークレットを確立するために使用される楕円曲線パブリック/プライベートキーペアを持ちます。ECDHE を有効にするために、Amazon Redshift で設定を行う必要はありません。ECDHE を使用してクライアント/サーバー間の通信を暗号化する SQL クライアントツールから接続する場合、Amazon Redshift は提供された暗号リストを使用して適切な接続を行います。詳細については、ウィキペディアの Elliptic curve diffie—hellman と OpenSSL ウェブサイトの Ciphers を参照してください。

Amazon Redshift Serverless への FIPS 準拠の SSL 接続の設定

FIPS 準拠の SSL 接続を使用する新しいワークグループを作成するには、create-workgroup コマンドを使用して use_fips_ssl パラメータと require_ssl パラメータを true に設定します。次の例を使用するには、*yourNamespaceName* を名前空間の名前に置き換え、*yourWorkgroupName* をワークグループの名前に置き換えます。

```
aws redshift-serverless create-workgroup \
--namespace-name yourNamespaceName \
--workgroup-name yourWorkgroupName \
--config-parameters '[{"parameterKey": "require_ssl", "parameterValue": "true"},
 {"parameterKey": "use_fips_ssl", "parameterValue": "true"}]'
```

FIPS 準拠の SSL 接続を使用するように既存のワークグループを更新するには、update-workgroup コマンドを使用して use_fips_ssl パラメータと require_ssl パラメータを true に設定します。use_fips_ssl パラメータを更新すると、Redshift Serverless によってワークグ

ループが再起動されることに注意してください。次の例では、*yourWorkgroupName* をワークグループの名前に置き換えます。

```
aws redshift-serverless update-workgroup \
--workgroup-name yourWorkgroupName \
--config-parameters '[{"parameterKey": "require_ssl", "parameterValue": "true"},
 {"parameterKey": "use_fips_ssl", "parameterValue": "true"}]'
```

FIPS 準拠の接続を使用するように Redshift Serverless を設定する方法の詳細については、「Amazon Redshift データベースデベロッパーガイド」の「use_fips_ssl」を参照してください。

Amazon Redshift マネージド VPC エンドポイントから Amazon Redshift Serverlessへの接続

他の VPC エンドポイントからの Amazon Redshift Serverless への接続

Amazon Redshift Serverless ワークグループのマネージド VPC エンドポイントの構成または設定については、「Amazon Redshift での Redshift 管理の VPC エンドポイントの操作」を参照してください。

インターフェイス VPC エンドポイントからの Amazon Redshift Serverless への接続 (AWS PrivateLink)

インターフェイス VPC エンドポイント (AWS PrivateLink) からの Amazon Redshift Serverless への接続の詳細については、「インターフェイス VPC エンドポイント」を参照してください。

別のアカウントの Redshift VPC エンドポイントから Amazon Redshift Serverless への接続

クロス VPC エンドポイントからの Amazon Redshift Serverless への接続

Amazon Redshift Serverless は VPC でプロビジョニングされます。別のアカウントの VPC にアクセス許可を付与して、自身のアカウントの Amazon Redshift Serverless にアクセスできます。これはマネージド VPC エンドポイントからの接続と似ています。ただし、この場合の接続は、例えば別のアカウントのデータベースクライアントから発信されます。実行できるオペレーションは、次のとおりいくつかあります。

- データベース所有者は、Amazon Redshift Serverless が配置されている VPC へのアクセスを同じリージョンの別のアカウントに付与できます。

- データベース所有者は Amazon Redshift Serverless アクセスを取り消すことができます。

クロスアカウントアクセスの主な利点は、データベースコラボレーションが容易になることです。データベースが配置されているアカウントにユーザーをプロビジョンする必要なくアクセスできるため、設定ステップの低減、時間の節約につながります。

別のアカウントの VPC へのアクセスを付与するために必要なアクセス許可

アクセスを付与したり、許可されているアクセスを変更したりするには、付与者に次のアクセス許可を持つアクセス許可ポリシーが割り当てられている必要があります。

- redshift-serverless:PutResourcePolicy
- redshift-serverless:GetResourcePolicy
- redshift-serverless:DeleteResourcePolicy
- ec2:CreateVpcEndpoint
- ec2:ModifyVpcEndpoint

AWS マネージドポリシー AmazonRedshiftFullAccess で指定されているその他のアクセス許可が必要になる場合があります。詳細については、「Amazon Redshift Serverless へのアクセス許可の付与」を参照してください。

被付与者には、次のとおりのアクセス許可があるアクセス許可ポリシーが割り当てられている必要があります。

- redshift-serverless:ListWorkgroups
- redshift-serverless:CreateEndpointAccess
- redshift-serverless:UpdateEndpointAccess
- redshift-serverless:GetEndpointAccess
- redshift-serverless:ListEndpointAccess
- redshift-serverless:DeleteEndpointAccess

ベストプラクティスとして、アクセス許可ポリシーを IAM ロールにアタッチし、それを必要に応じてユーザーやグループに割り当てることをお勧めします。詳細については、「Amazon Redshift での Identity and Access Management」を参照してください。

クロス VPC アクセスを設定するために使用されるリソースポリシー例は次のとおりです。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "CrossAccountCrossVPCAccess",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "123456789012",
                    "234567890123"
                ]
            },
            "Action": [
                "redshift-serverless:CreateEndpointAccess",
                "redshift-serverless:UpdateEndpointAccess",
                "redshift-serverless:DeleteEndpointAccess",
                "redshift-serverless:GetEndpointAccess"
            ],
            "Resource": "arn:aws:redshift-serverless:*:*:*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestedRegion": "us-east-1"
                },
                "StringLike": {
                    "aws:SourceVpc": [
                       "vpc-11223344556677889",
                       "vpc-12345678"
                    ]
                }
            }
        }
    ]
}
```

このセクションで説明する手順では、アクセスを付与するユーザーに適切なアクセス許可が割り当てられていることを前提としています。例えば、リストされているアクセス許可を持つ IAM ロールが割り当てられている場合などです。この手順では、ワークグループに適切なリソースのアクセス許可が付与された IAM ロールがあることも前提としています。

コンソールを使用した別のアカウントへの VPC アクセスの付与

この手順では、データベースの所有者がデータベースへのアクセスを付与する場合にデータベースのアクセスを設定するステップを説明します。

所有者アカウントからのアクセスの付与

1. Amazon Redshift Serverless ワークグループのプロパティには、[データアクセス] タブに [許可を付与されたアカウント] というリストがあります。このリストにはワークグループへのアクセスを付与されたアカウントと VPC が表示されます。このリストにアカウントを追加するには、このリストを検索して、[アクセス権を付与]] をクリックします。
2. 被付与者の情報を追加できるウィンドウが開きます。アクセスを許可する AWS アカウント ID を入力します。アカウント ID は、12 桁の ID です。
3. すべての VPC へのアクセス権を付与するか、特定の VPC へのアクセスを付与します。特定の VPC へのアクセスのみを許可する場合は、各 ID を入力して [VPC を追加] をクリックして ID を追加します。
4. 完了したら、[変更の保存] をクリックします。

変更を保存すると、追加したアカウントが [許可を付与されたアカウント] リストに表示されます。このエントリには、[アカウント ID] とアクセス権を付与した VPC のリストが表示されます。

データベース所有者は、アカウントへのアクセスを取り消すこともできます。所有者は、いつでもアクセスを取り消すことができます。

アカウントのアクセス権の取り消し

1. 許可を付与されたアカウントのリストから開始します。まず、単一または複数のアカウントを選択します。
2. [アクセスを取り消す] をクリックします。

アクセス権が付与されたら、権限受領者のデータベース管理者はコンソールを調べてアクセス権の有無を確認できます。

コンソールを使用して、別のアカウントにアクセスするためのアクセス許可が付与されていることを確認できます。

1. Amazon Redshift Serverless ワークグループのプロパティには、[データアクセス] タブに [Authorized accounts] というリストがあります。このリストには、このワークグループからアク

セスできるアカウントが表示されます。被付与者は、ワークグループのエンドポイント URL を使用してワークグループに直接アクセスすることはできません。被付与者がワークグループにアクセスするには、[エンドポイント] セクションに移動して、[エンドポイントを作成] を選択します。

2. 次に、被付与者として、ワークグループにアクセスするためのエンドポイント名と VPC を指定します。
3. エンドポイントの作成が正常に完了すると、エンドポイントが [エンドポイント] セクションに表示されます。エンドポイントの URL も表示されます。このエンドポイント URL を使用して、ワークグループにアクセスします。

CLI コマンドを使用した別のアカウントへのアクセスの付与

アクセスを付与するアカウントは、まず put-resource-policy を使用して接続する別のアカウントにアクセス権を付与する必要があります。データベース所有者は、put-resource-policy を呼び出して、別のアカウントがワークグループへの接続を作成することを許可できます。その後、被付与者アカウントは、create-endpoint-authorization を使用して、許可された VPC を介したワークグループへの接続を作成できます。

特定のアカウントと VPC へのアクセスを許可するために呼び出すことができる put-resource-policy のプロパティは次のとおりです。

```
aws redshift-serverless put-resource-policy
--resource-arn <value>
--policy <value>
```

コマンドを呼び出した後、get-resource-policy を呼び出して resource-arn を指定すると、リソースへのアクセスを許可されているアカウントと VPC を確認できます。

被付与者が実行できる呼び出しは、次のとおりです。付与されたアクセス権に関する情報が表示されます。具体的には、アクセス権が付与された VPC を含むリストが返されます。

```
aws redshift-serverless list-workgroups
--owner-account <value>
```

この目的は、被付与者が付与されるアカウントからエンドポイントの承認に関する情報を取得することにあります。owner-account は共有するアカウントです。これを実行すると、ワークグループごとの CrossAccountVpcs、つまり許可されている VPC のリストが返されます。参考までに、ワークグループで使用できるすべてのプロパティは、次のとおりです。

```
Output: workgroup (Object)
workgroupId String,
workgroupArn String,
workgroupName String,
status: String,
namespaceName: String,
baseCapacity: Integer, (Not-applicable)
enhancedVpcRouting: Boolean,
configParameters: List,
securityGroupIds: List,
subnetIds: List,
endpoint: String,
publiclyAccessible: Boolean,
creationDate: Timestamp,
port: Integer,
CrossAccountVpcs: List
```

> **Note**
>
> 念のため、クラスターの再配置は、追加の Redshift ネットワーク機能を設定するための前提条件ではありません。また、次を有効にするうえでオンにする必要もありません。
>
> - クロスアカウントまたはクロスリージョン VPC から Redshift への接続 – このセクションで説明されるとおり、ある AWS 仮想プライベートクラウド (VPC) から Redshift データベースが配置された別の仮想プライベートクラウド (VPC) に接続できます。
> - カスタムドメイン名の設定 – Amazon Redshift クラスターまたは Amazon Redshift Serverless ワークグループのカスタムドメイン名 (別名: カスタム URL) を作成して、エンドポイント名をより覚えやすくシンプルにできます。詳細については、「クライアント接続にカスタムドメイン名を使用する」を参照してください。

その他のリソース

ネットワークトラフィックの設定手順については、「Public accessibility with default or custom security group configuration」を参照してください。このセクションでは、クラスターへのパブリックアクセス可能な場合のユースケースについても説明しています。

ネットワークトラフィックの設定手順については、「Private accessibility with default or custom security group configuration」を参照してください。このセクションでは、クラスターがインターネットで利用できない場合のユースケースについても説明します。

アクセス許可の付与、追加サービスへのアクセスの許可、IAM ロールの作成など、Amazon Redshift Serverless への安全な接続の詳細については、「Amazon Redshift Serverless での Identity and Access Management」を参照してください。

Amazon Redshift Serverless のフェデレーションユーザーに付与するデータベースロールの定義

組織に所属しているときは、関連するロール群があります。例えば、プログラマーやマネージャーなど、職務に応じたロールがあります。ロールによって、どのアプリケーションやデータにアクセスできるかが決まります。ほとんどの組織では、Microsoft Active Directory などの ID プロバイダーを使用して、ユーザーやグループにロールを割り当てています。組織は個々のユーザーに対してそれほど多くの管理を行う必要がないため、リソースアクセスを制御するためのロールの使用は増えてきました。

最近、Amazon Redshift Serverless にロールベースのアクセスコントロールが導入されました。データベースロールを使用すると、スキーマまたはテーブルなどのデータおよびオブジェクトへのアクセスを保護することができます。または、ロールを使用して、システムモニタリングやデータベース管理者などの、一連の上位権限を定義することもできます。ただし、データベースロールにリソース権限を付与したあとは、組織内のユーザーのロールをデータベースロールに結び付ける、別の手順があります。SQL ステートメントを実行することにより、初回サインイン時に各ユーザーをデータベースロールに割り当てることができますが、手間がかかります。より簡単な方法は、付与するデータベースロールを定義して Amazon Redshift Serverless に渡すことです。これには最初のサインインプロセスを簡略化できるという利点があります。

ロールは `GetCredentials` を使用して Amazon Redshift Serverless に渡すことができます。ユーザーが Amazon Redshift Serverless データベースに初めてサインインすると、関連するデータベースユーザーが作成され、一致するデータベースロールにマッピングされます。このトピックでは、Amazon Redshift Serverless ロールを渡すメカニズムについて詳しく説明します。

データベースロールを渡すには、次のようないくつかの主なユースケースがあります。

- ユーザーがサードパーティーの ID プロバイダー (通常はフェデレーションを設定済み) を介してサインインし、セッションタグを使用してロールを渡します。
- ユーザーが IAM サインイン認証情報を使用してサインインし、タグキーおよび値を使用してそれらのロールを渡します。

ロールベースのアクセスコントロールの詳細については、「ロールベースのアクセスコントロール (RBAC)」を参照してください。

データベースロールの定義

Amazon Redshift Serverless にロールを渡すには、データベースにデータベースロールを設定し、データベースリソースに適切な権限を付与する必要があります。例えば、簡単なシナリオでは、sales という名前のデータベースロールを作成し、そのロールに売上データを含むクエリテーブルへのアクセス権を付与できます。データベースロールを作成して権限を付与する方法の詳細については、「CREATE ROLE」と「GRANT」を参照してください。

フェデレーションユーザーに付与するデータベースロールを定義するユースケース

これらのセクションでは、データベースロールを Amazon Redshift Serverless に渡すことでデータベースリソースへのアクセスを簡素化する、いくつかのユースケースについて概説します。

ID プロバイダーを使用したサインイン

最初のユースケースでは、組織が ID およびアクセス管理サービスのユーザー ID を持っていると仮定しています。このサービスは、JumpCloud や Okta などのクラウドベースでも、Microsoft Active Directory などのオンプレミスでもかまいません。目的は、ユーザーがクエリエディタ V2 などのクライアントにサインインするとき、または JDBC クライアントでサインインするときに、ID プロバイダーからデータベースロールにユーザーのロールを自動的にマッピングすることです。これを設定するには、いくつかの設定タスクを完了する必要があります。これには以下が含まれます。

1. 信頼関係を使用して ID プロバイダー (IdP) とのフェデレーション統合を設定します。これは必須条件です。これを設定すると、ID プロバイダーが SAML アサーションによるユーザー認証とサインイン認証情報の提供を行います。詳細については、「サードパーティーの SAML ソリューションプロバイダーと AWS の統合」を参照してください。また、「Federate access to Amazon Redshift query editor V2 with Active Directory Federation Services (AD FS)(Active Directory フェデレーションサービス (AD FS) を使用して Amazon Redshift クエリエディタ v2 へのアクセスをフェデレーションする)」または「Federate single sign-on access to Amazon Redshift query editor v2 with Okta (Okta を使用して Amazon Redshift クエリエディタ v2 へのシングルサインオンアクセスをフェデレーションする)」を参照してください。
2. ユーザーは、次のポリシー許可を持っている必要があります。
   - `GetCredentials` – Amazon Redshift Serverless へのログインを一時的に認可するための認証情報を提供します。

- sts:AssumeRoleWithSAML — エンタープライズ ID ストアまたはディレクトリをロールベースの AWS アクセスに結び付けるメカニズムを提供します。
- sts:TagSession — ID プロバイダープリンシパルでの、タグセッションアクションへの権限。

この場合、AssumeRoleWithSAML は SAML 認証レスポンスによって認証されたユーザーのセキュリティ認証情報のセットを返します。この操作により、ユーザー固有の認証情報なしで、ID ストアまたはディレクトリをロールベースの AWS アクセスに結び付けるメカニズムが提供されます。AssumeRoleWithSAML への権限を持つユーザーの場合、ロール情報の受け渡しに使用される SAML アサーションの管理は ID プロバイダーが行います。

ベストプラクティスとして、アクセス許可ポリシーを IAM ロールにアタッチし、それを必要に応じてユーザーやグループに割り当てることをお勧めします。詳細については、「Amazon Redshift での Identity and Access Management」を参照してください。

3. タグ RedshiftDbRoles は、role1:role2 の形式で、コロンで区切られたロール値を使用して設定します。例えば、manager:engineer。これらは、ご使用の ID プロバイダーで設定されたセッションタグ実装から取得できます。SAML 認証リクエストはロールをプログラムで渡します。セッションタグの受け渡しについては、「AWS STS でのセッションタグの引き渡し」を参照してください。

   データベースに存在しないロール名を渡した場合、そのロール名は無視されます。

このユースケースでは、ユーザーがフェデレーション ID を使用してサインインすると、そのロールはセッションタグのキーと値を介した承認リクエストで渡されます。そして、承認後、GetCredentials はロールをデータベースに渡します。接続に成功すると、データベースロールがマッピングされ、ユーザーは自分のロールに対応するデータベースタスクを実行することができます。操作の重要な部分は、最初の承認リクエストで RedshiftDbRoles セッションタグにロールが割り当てられることです。詳細については、「AssumeRoleWithSAML を使用したセッションタグの受け渡し」を参照してください。

IAM 認証情報を使ったサインイン

2 つ目のユースケースでは、ロールをユーザーに渡すことにより、ユーザーは IAM 認証情報を使用してデータベースクライアントアプリケーションにアクセスできます。

1. この場合、サインインするユーザーには、以下のアクションを行うためのポリシー権限が割り当てられている必要があります。

- `tag:GetResources` - 指定されたタグに関連付けられているタグ付きリソースを返します。
- `tag:GetTagKeys`— 現在使用中のタグキーを返します。

  ベストプラクティスとして、アクセス許可ポリシーを IAM ロールにアタッチし、それを必要に応じてユーザーやグループに割り当てることをお勧めします。詳細については、「Amazon Redshift での Identity and Access Management」を参照してください。

2. Amazon Redshift Serverless などのデータベースサービスにアクセスするには、アクセス許可も必要です。
3. このユースケースでは、AWS Identity and Access Management でロールのタグ値を設定します。[タグの編集] を選択して、ロールを含むタグ値文字列を持つ RedShiftDBroles という名前のタグキーを作成できます。例えば、manager:engineer です。

ユーザーがログインすると、そのロールが承認リクエストに追加され、データベースに渡されます。ロールは既存のデータベースロールにマッピングされます。

その他のリソース

ユースケースで述べたように、IdP と AWS との信頼関係を設定できます。詳細については、「証明書利用者の信頼およびクレームの追加によって SAML 2.0 IdP を設定する」を参照してください。

Amazon Redshift Serverless での Identity and Access Management

Amazon Redshift へのアクセスには、AWS が要求の認証に使用する認証情報が必要です。これらの認証情報には、Amazon Redshift Serverless などの AWS リソースへのアクセス許可が必要です。

次のセクションでは、AWS Identity and Access Management(IAM) と Amazon Redshift を使用してリソースにアクセスできるユーザーを制御することで、リソースを保護する方法について詳しく説明します。詳細については、「Amazon Redshift での Identity and Access Management」を参照してください。

Amazon Redshift Serverless へのアクセス許可の付与

Amazon Redshift Serverless が、他の AWS サービスにアクセスする際には、そのためのアクセス許可が必要です。一部の Amazon Redshift 機能では、ユーザーに代わって Amazon Redshift が他の AWS のサービスにアクセスする必要があります。Amazon Redshift Serverless インスタンスを、ユーザーに代わって動作させるためには、それに対してセキュリティ認証情報を指定します。セキュリティ認証情報を指定する望ましい方法は、AWS Identity and Access Management(IAM) ロールを

指定することです。あるいは、Amazon Redshift コンソールから IAM ロールを作成し、デフォルトとして設定することもできます。詳細については、「Amazon Redshift 用にデフォルトの IAM ロールを作成する」を参照してください。

他の AWS サービスにアクセスするためには、適切なアクセス許可を持つ IAM ロールを作成します。また、この作成したロールを、Amazon Redshift Serverless と関連付ける必要もあります。さらに、Amazon Redshift コマンドの実行時にロールの Amazon リソースネーム (ARN) を指定するか、`default`キーワードを指定します。

IAM ロールの信頼関係を https://console.aws.amazon.com/iam/ で変更する場合は、`redshift-serverless.amazonaws.com` と `redshift.amazonaws.com` がプリンシパルサービス名として含まれていることを確認します。ユーザーに代わって他の AWS サービスにアクセスするための IAM ロールの管理については、「ユーザーに代わって Amazon Redshift が他の AWS のサービスにアクセスすることを許可する」を参照してください。

Amazon Redshift 用にデフォルトの IAM ロールを作成する

Amazon Redshift コンソールから IAM ロールを作成する場合、Amazon Redshift は AWS アカウント内でロールをプログラム的に作成します。さらに Amazon Redshift では、作成したロールに対し、既存の AWS 管理ポリシーを自動的にアタッチします。このアプローチにより、ロール作成のために IAM コンソールに切り替える必要はなくなり、作業を Amazon Redshift コンソール内で完了できます。

クラスター用としてコンソールから作成する IAM ロールには、`AmazonRedshiftAllCommandsFullAccess` マネージドのポリシーが自動的にアタッチされます。この IAM ロールにより Amazon Redshift は、IAM アカウントにある AWS リソースにおいて、データのコピー、解放、クエリ、および分析を行えるようになります。これに関連するコマンドとしては、COPY、UNLOAD、CREATE EXTERNAL FUNCTION、CREATE EXTERNAL TABLE、CREATE EXTERNAL SCHEMA、CREATE MODEL、CREATE LIBRARY などがあります。Amazon Redshift でのデフォルト使用のために IAM ロールを作成する方法については、「Amazon Redshift 用にデフォルトの IAM ロールを作成する」を参照してください。

Amazon Redshift のデフォルトとして IAM ロールを作成するには、AWS マネジメントコンソールを開いて [Amazon Redshift コンソール] を選択します。次に、メニューで [Redshift Serverless] を選択します。Serverless ダッシュボードから、新しいワークグループを作成できます。作成手順では、IAM ロールの選択、または新しい IAM ロールの設定を行います。

既に Amazon Redshift Serverless ワークグループがあり、それに対して IAM ロールを設定する場合は、AWS マネジメントコンソール を開きます。[Amazon Redshift コンソール] を選択して、次に

[Redshift Serverless] を選択します。Amazon Redshift Serverless コンソールで、[名前空間の設定] を選択します。[セキュリティと暗号化] でアクセス許可を編集できます。

名前空間への IAM ロールの割り当て

各 IAM ロールは各ロールが AWS で実行できるアクションを決定するアクセス許可ポリシーを持つ AWS ID です。ロールは、それを必要とするすべてのユーザーが引き受け可能です。また、各名前空間は、テーブルやスキーマなどのオブジェクトとユーザーのコレクションです。Amazon Redshift Serverless を使用する場合、複数の IAM ロールを名前空間に関連付けることができます。これにより、データベースオブジェクトのコレクションに対してアクセス許可を適切に構成することが容易になり、ロールは内部データと外部データの両方に対してアクションを実行できます。例えば、Amazon Redshift データベースで COPY コマンドを実行し、Amazon S3 からデータを取得して Redshift テーブルに入力するようにできます。

このセクションで前に説明したとおり、コンソールを使用して、複数のロールを名前空間に関連付けることができます。API コマンド `CreateNamespace`、または CLI コマンド `create-namespace` を使用することもできます。API または CLI コマンドを使用すると、1 つ以上のロールで `IAMRoles` を入力することにより、名前空間に IAM ロールを割り当てることができます。具体的には、コレクションに特定のロールの ARN を追加します。

名前空間に関連付けられた IAM ロールの管理

AWS マネジメントコンソール で、AWS Identity and Access Management のロールに対するアクセス権限ポリシーを管理できます。[Namespace configuration] (名前空間の設定) の中の設定を使って、名前空間の IAM ロールを管理できます。Amazon Redshift Serverless での名前空間とその使用の詳細については、「ワークグループと名前空間」を参照してください。

Amazon Redshift 用の IAM 認証情報の使用開始

Amazon Redshift コンソールに初めてサインインし、Amazon Redshift Serverless を初めて試す場合は、必要なポリシーがアタッチされた IAM ロールを持つユーザーとしてサインインすることをお勧めします。Amazon Redshift Serverless インスタンスの作成を開始すると、サインイン時に使用した IAM ロール名が Amazon Redshift に記録されます。Amazon Redshift コンソールと Amazon Redshift Serverless コンソールには、同じ認証情報を使用してサインインできます。

Amazon Redshift Serverless インスタンスの作成中に、データベースを作成できます。一時認証情報オプションを使用しながらデータベースに接続するには、クエリエディタ v2 を使用します。

データベース用として保持される新しい管理者ユーザー名とパスワードを追加するには、[Customize admin user credentials] (管理者ユーザー認証情報のカスタマイズ) をクリックし、新しい管理者ユーザ名とそのパスワードを入力します。

Amazon Redshift Serverless の使用を開始し、ワークグループと名前空間をコンソールで初めて作成する場合は、アクセス許可ポリシーがアタッチされた IAM ロールを使用します。このロールでは、管理者のアクセス許可 (`arn:aws:iam::aws:policy/AdministratorAccess`) または Amazon Redshift への完全なアクセス許可 (`arn:aws:iam::aws:policy/AmazonRedshiftFullAccess`) を IAM ポリシーにアタッチする必要があります。

Amazon Redshift Serverless コンソールの使用を開始する際、Amazon Redshift Serverless が IAM 認証情報を使用する方法の概要について、次のシナリオで説明します。

- [Use default setting] (デフォルト設定の使用) を選択した場合 - Amazon Redshift Serverlessが、現在の IAM ID をデータベースのスーパーユーザーに変換します。この同じ IAM アイデンティティを使用して、Amazon Redshift Serverless コンソールから、Amazon Redshift Serverless 内のデータベースに対しスーパーユーザーアクションを実行できます。
- [Admin user name] (管理者ユーザー名) とパスワード を指定せずに、[Customize settings] (設定のカスタマイズ) をクリックすると、Amazon Redshift Serverless は、現在の IAM 認証情報をデフォルトの管理者ユーザー認証情報として使用します。
- [Customize settings] (設定のカスタマイズ) を選択し [Admin user name] (管理者ユーザー名) とパスワードを指定した場合 - Amazon Redshift Serverless は、現在の IAM ID をデータベースのスーパーユーザーに変換します。また、Amazon Redshift Serverless は、別の長期ログインユーザー名とパスワードのペアをスーパーユーザーとして作成します。ここで作成されたユーザーネームとパスワードのペア、または現在の IAM アイデンティティを使用して、スーパーユーザーとしてデータベースにログインできます。

データベースロールのアクセス許可を使用した Amazon Redshift Serverless データベースオブジェクトへのアクセス

この手順では、Amazon Redshift データベースロールを通じてテーブルをクエリするアクセス許可を付与する方法を示します。ロールを割り当てるには、IAM でユーザーにタグをアタッチします。ユーザーがサインインすると、ロールが Amazon Redshift に渡されます。これは、「Amazon Redshift Serverless のフェデレーションユーザーに付与するデータベースロールの定義」の概念の例に基づく説明です。これらのステップを実行する利点は、ユーザーをデータベースロールに関連付けることで、データベースオブジェクトごとにアクセス許可を設定する必要がなくなることです。これ

により、ユーザーがテーブルに対して実行するクエリ、変更、データの追加や、その他のアクションを管理しやすくなります。

この手順では、ユーザーが Amazon Redshift Serverless データベースをセットアップ済みであり、データベースにアクセス許可を付与できることを前提としています。また、AWS コンソールでの IAM ユーザーの作成、IAM ロールの作成、ポリシー許可の割り当てに対するアクセス許可を持っていることを前提としています。

1. IAM コンソールを使用して IAM ロールを作成します。後で、このユーザーを使用してデータベースに接続します。
2. クエリエディタ v2 または別の SQL クライアントを使用して、Redshift データベースロールを作成します。データベースロールの作成の詳細については、「CREATE ROLE」を参照してください。

   ```
   CREATE ROLE urban_planning;
   ```

   SVV_ROLES システムビューにクエリを実行し、ロールが作成済みであることを確認します。このクエリでは、システムロールも返されます。

   ```
   SELECT * from SVV_ROLES;
   ```

3. 作成したデータベースロールに、テーブルから選択するためのアクセス許可を付与します (作成した IAM ユーザーは、このデータベースロールにより、最終的にサインインしてテーブルからレコードを選択します)。次のコード例のロール名とテーブル名はサンプルです。ここでは、cities という名前のテーブルから選択するためのアクセス許可を付与します。

   ```
   GRANT SELECT on TABLE cities to ROLE urban_planning;
   ```

4. AWS Identity and Access Management コンソールを使用して IAM ロールを作成します。このロールは、クエリエディタ v2 を使用するためのアクセス許可を付与します。新しい IAM ロールを作成し、信頼できるエンティティタイプとして、AWS アカウントを選択します。次に [このアカウント] を選択します。以下のポリシーのアクセス許可をロールに付与します。

   - AmazonRedshiftReadOnlyAccess
   - tag:GetResources
   - tag:GetTagKeys
   - sqlworkbench:ListDatabases と sqlworkbench:UpdateConnection を含む、sqlworkbench のすべてのアクション。

5. IAM コンソールで、以前に作成した IAM ユーザーにキー `RedshiftDbRoles` を使用してタグを追加します。タグの値は、最初のステップで作成したデータベースロールと一致する必要があります。それは `urban_planning` サンプルにあります。

これらのステップを完了したら、IAM コンソールで作成したユーザーに IAM ロールを割り当てます。ユーザーがクエリエディタ v2 を使用してデータベースにサインインすると、タグ内のデータベースロール名が Amazon Redshift に渡され、ユーザーに関連付けられます。このデータベースロールにより、ユーザーは適切なテーブルをクエリできます。例えば、このサンプルのユーザーは `urban_planning` データベースロールを通じて `cities` テーブルをクエリできます。

プロビジョニングされたクラスターを Amazon Redshift Serverless へ移行する

既存のプロビジョニングされたクラスターを Amazon Redshift Serverless に移行して、コンピューティングリソースのオンデマンドおよび自動スケーリングを有効にできます。プロビジョニングされたクラスターを Amazon Redshift Serverless に移行すると、使用するリソースに対してのみ料金を支払うことでコストを最適化し、ワークロードの需要に基づいて容量を自動的にスケーリングできます。移行の一般的なユースケースには、アドホッククエリの実行、定期的なデータ処理ジョブ、リソースの過剰プロビジョニングのない予測不可能なワークロードの処理などがあります。プロビジョニングされた Amazon Redshift クラスターをサーバーレスデプロイオプションに移行するには、次のタスクセットを実行します。

プロビジョニングされたクラスターのスナップショットを作成する

> **Note**
>
> Amazon Redshift は、プロビジョニングされたクラススナップショットをサーバーレス名前空間に復元するときに、インターリーブキーを自動的に複合キーに変換します。

プロビジョニングされたクラスターから Amazon Redshift Serverless にデータを転送するには、プロビジョニングされたクラスターのスナップショットを作成し、Amazon Redshift Serverless でスナップショットを復元します。

> **Note**
>
> データをサーバーレスワークグループに移行する前に、プロビジョニングしたクラスターのニーズが Amazon Redshift Serverless で選択した RPU の量と互換性があることを確認してください。

プロビジョニングされたクラスターのスナップショットを作成するには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Clusters] (クラスター)、[Snapshots] (スナップショット)、次に [Create snapshot] (スナップショットの作成) を選択します。
3. スケジュール定義のプロパティを入力してから、[スナップショットの作成] を選択します。スナップショットが使用できるようになるまではしばらくかかります。

プロビジョニングされたクラスタースナップショットをサーバーレス名前空間に復元するには以下の内容を実行します。

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. Amazon Redshift でプロビジョニングされたクラスターのコンソールを開き、[Clusters] (クラスター) ページ、[Snapshots] (スナップショット) ページの順に移動します。
3. 使用するスナップショットを選択します。
4. [Restore snapshot] (スナップショットから復元)、[Restore to serverless namespace] (サーバーレス名前空間に復元) の順にクリックします。
5. スナップショットの復元先の名前空間を選択します。
6. スナップショットから復元することを確認します。この操作により、サーバーレスエンドポイント内のすべてのデータベースが、プロビジョニングされたクラスターのデータで置き換えられます。[復元] を選択します。

プロビジョニングされたクラスタースナップショットの詳細については、「Amazon Redshift スナップショット」を参照してください。

ドライバーでの Amazon Redshift Serverless への接続

好みの SQL クライアントを使用して Amazon Redshift Serverless に接続するには、Amazon Redshift が提供する JDBC ドライバーバージョン 2.x を使用することができます。Amazon Redshift JDBC ドライバーバージョン 2.x の最新バージョンを使用して Amazon Redshift に接続することをお勧めします。ポート番号はオプションです。そのドライバーが含まれていない場合、Amazon Redshift Serverless のデフォルトのポート番号は 5439 です。5431-5455 または 8191-8215 のポート範囲から別のポートに変更できます。サーバーレスエンドポイントのデフォルトのポートを変更するには、AWS CLI と Amazon Redshift API を使用します。

JDBC、ODBC、または Python ドライバーに使用する正確なエンドポイントを見つけるには、Amazon Redshift Serverless の「Workgroup configuration (ワークグループの設定)」を参照してください。Amazon Redshift Serverless API オペレーションの `GetWorkgroup` または AWS CLI オペレーション `get-workgroups` を使用して、ワークグループに関する情報を戻してから接続することもできます。

パスワードベースの認証を使用した接続

パスワードベースの認証で Amazon Redshift JDBC ドライバーバージョン 2.x を使用して接続を確立するには、次の構文を使用します。

```
jdbc:redshift://<workgroup-name>.<account-number>.<aws-region>.redshift-
serverless.amazonaws.com:5439/?username=username&password=password
```

パスワードベースの認証で Amazon Redshift Python コネクタを使用して接続を確立するには、次の構文を使用します。

```
import redshift_connector
with redshift_connector.connect(
    host='<workgroup-name>.<account-number>.<aws-region>.redshift-
serverless.amazonaws.com',
    database='<database-name>',
    user='username',
    password='password'
    # port value of 5439 is specified by default
) as conn:
    pass
```

パスワードベースの認証で Amazon Redshift ODBC ドライバーバージョン 2.x を使用して接続を確立するには、次の構文を使用します。

```
Driver={Amazon Redshift ODBC Driver (x64)}; Server=<workgroup-name>.<account-
number>.<aws-region>.redshift-serverless.amazonaws.com; Database=database-name;
 User=username; Password=password
```

IAM を使用した接続

IAM でログインする場合は、Amazon Redshift Serverless GetCredentials API オペレーションを使用します。

IAM 認証を使用するには、次の例に示すように、jdbc:redshift: に続いて Amazon Redshift JDBC URL に iam: を追加します。

```
jdbc:redshift:iam://<workgroup-name>.<account-number>.<aws-region>.redshift-
serverless.amazonaws.com:5439/<database-name>
```

この Amazon Redshift Serverless エンドポイントは、dbUser、dbGroup、または自動作成のカスタマイズをサポートしていません。デフォルトでは、ドライバーはログイン時にデータベースユーザーを自動的に作成します。次に、IAM で指定されたタグ、または ID プロバイダー (IdP) で定義されたグループに基づいて、Amazon Redshift データベースロールにユーザーを割り当てます。

AWS ID が redshift-serverless:GetCredentials アクションに対する正しい IAM ポリシーを保持していることを確認します。次に、AWS ID に Amazon Redshift Serverless に接続するための正しいアクセス権限を付与する IAM ポリシーの例を示します。IAM アクセス許可の詳細については、「IAM ユーザーガイド」の「IAM ID のアクセス許可の追加および削除」を参照してください。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Action": "redshift-serverless:GetCredentials",
            "Resource": "*"
        }
    ]
}
```

IAM ベースの認証で Amazon Redshift Python コネクタを使用して接続を確立するには、次の構文に示すように、コード内で iam=true を使用します。

```
import redshift_connector
with redshift_connector.connect(
    iam=True,
    host='<workgroup-name>.<account-number>.<aws-region>.redshift-
serverless.amazonaws.com',
    database='<database-name>'
    <IAM credentials>
) as conn:
    pass
```

IAM credentials では、以下を含む任意の認証情報を使用できます。

- AWS プロファイル設定。
- IAM 認証情報 (アクセスキー ID、シークレットアクセスキー、およびオプションでセッショントークン)。
- アイデンティティプロバイダーフェデレーション。

Amazon Redshift ODBC ドライバーバージョン 2.x を使用して IAM ベースの認証とプロファイルで接続を確立するには、次の構文を使用します。

```
Driver={Amazon Redshift ODBC Driver (x64)}; IAM=true; Server=<workgroup-name>.<account-
number>.<aws-region>.redshift-serverless.amazonaws.com; Database=database-name;
 Profile=aws-profile-name;
```

GetClusterCredentials API で IAM を使用して接続する

> **Note**
>
> Amazon Redshift Serverless に接続するときは、GetCredentials API を使用することをお勧めします。この API は、包括的なロールベースのアクセスコントロール (RBAC) 機能と、GetClusterCredentials では利用できないその他の新機能を提供します。プロビジョニングされたクラスターからサーバーレスワークグループへの移行を簡素化するために GetClusterCredentials API をサポートしていますが、互換性を最適化するために、できるだけ早く GetCredentials の使用に移行することを強くお勧めします。

GetClusterCredentials API を使用して Amazon Redshift Serverless への接続を確立できます。この認証方法を実装するには、次のパラメータを組み込んでクライアントまたはアプリケーションを変更します。

- iam=true
- clusterid/cluster_identifier=redshift-serverless-<*workgroup-name*>
- region=<*aws-region*>

次の例は、3 つのドライバーすべてにわたる BrowserSAML プラグインを示しています。これは、利用可能な認証アプローチの 1 つを表します。特定の要件に応じて、別の認証方法またはプラグインを使用するように例を変更できます。

GetClusterCredentials の IAM ポリシーアクセス許可

以下は、Amazon Redshift Serverless で GetClusterCredentials を使用するために必要なアクセス許可を持つ IAM ポリシーの例です。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "redshift-serverless:GetWorkgroup",
                "redshift-serverless:GetNamespace",
                "redshift:GetClusterCredentials",
                "redshift:CreateClusterUser",
                "redshift:JoinGroup",
                "redshift:ExecuteQuery",
                "redshift:FetchResults",
                "redshift:DescribeClusters",
                "redshift:DescribeTable"
            ],
            "Resource": [
                "arn:aws:redshift-serverless:us-
east-1:111122223333:workgroup/workgroup-name",
```

```
                "arn:aws:redshift-serverless:us-
east-1:111122223333:namespace/namespace-name",
                "arn:aws:redshift:us-east-1:111122223333:cluster:cluster-name",
                "arn:aws:redshift:us-east-1:111122223333:dbuser:database-name/
${redshift:DbUser}"
            ]
        }
    ]
}
```

Amazon Redshift JDBC ドライバーバージョン 2.x を使用して、GetClusterCredentials で接続を確立するには、次の構文を使用します。

```
jdbc:redshift:iam://redshift-serverless-<workgroup-name>:<aws-region>/<database-name>?
plugin_name=com.amazon.redshift.plugin.BrowserSamlCredentialsProvider&login_url=<single
 sign-on URL from IdP>"
```

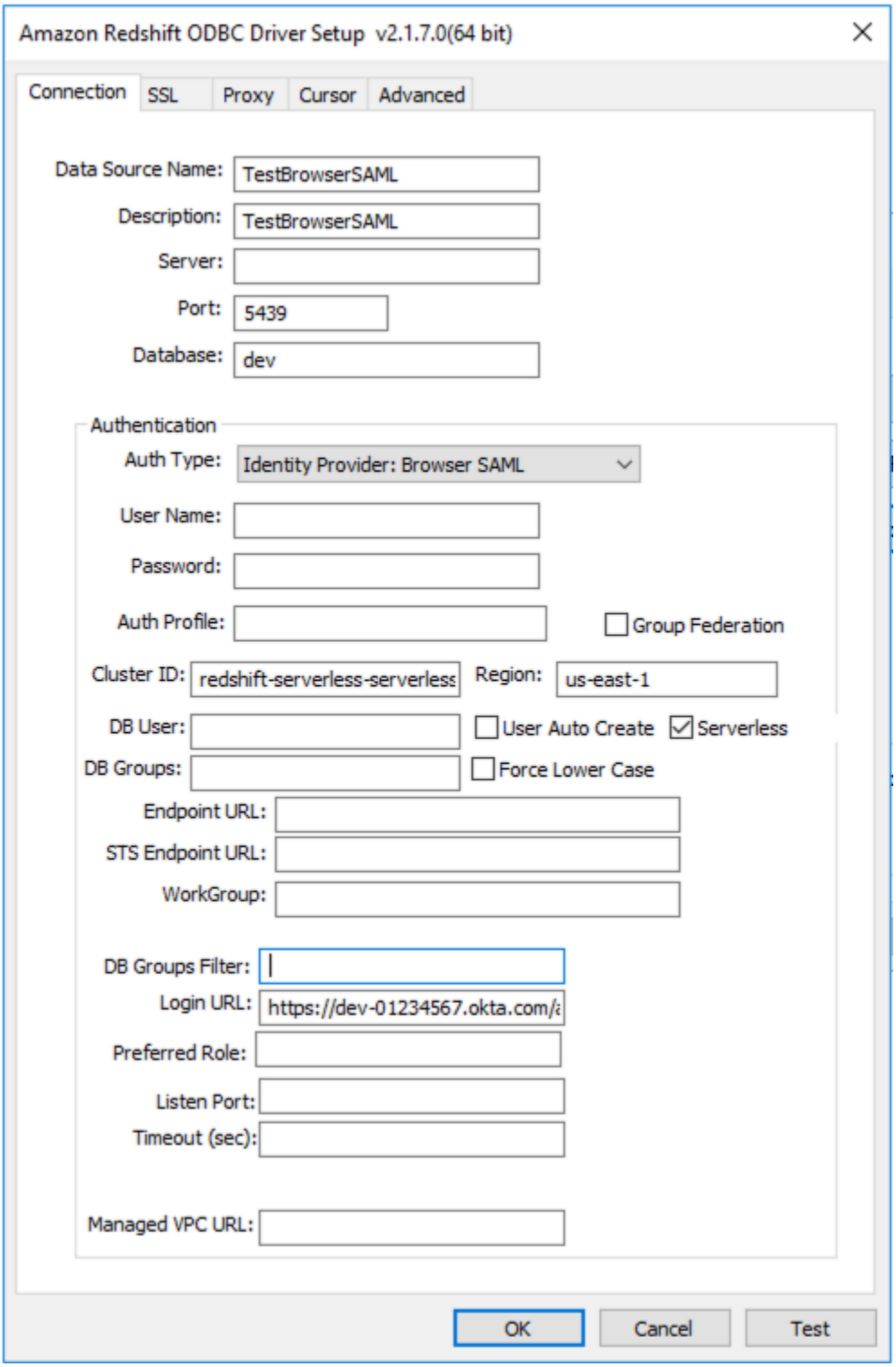
Amazon Redshift Python コネクタを使用して、GetClusterCredentials で接続を確立するには、次の構文を使用します。

```
import redshift_connector
with redshift_connector.connect(
    iam=True,
    cluster_identifier='redshift-serverless-<workgroup-name>',
    region='<aws-region>',
    database='<database-name>',
    credentials_provider='BrowserSamlCredentialsProvider'
    login_url='<single sign-on URL from IdP>'
    # port value of 5439 is specified by default
) as conn:
    pass
```

Amazon Redshift ODBC ドライバーバージョン 2.x を使用して、GetClusterCredentials で接続を確立するには、次の構文を使用します。

```
Driver= {Amazon Redshift ODBC Driver (x64)}; IAM=true; isServerless=true;
 ClusterId=redshift-serverless-<workgroup-name>; region=<aws-region>;
 plugin_name=BrowserSAML;login_url=<single sign-on URL from IdP>
```

Windows での ODBC DSN 設定の例を次に示します。



Amazon Redshift Serverless SDK を使用する

Amazon Redshift SDK を使用して管理スクリプトを作成した場合は、新しい Amazon Redshift Serverless SDK を使用して Amazon Redshift Serverless と関連リソースを管理する必要がありま

す。利用可能なAPI オペレーションの詳細については、「Amazon Redshift Serverless API のリファレンスガイド」を参照してください。

ワークグループと名前空間

Amazon Redshift Serverless でワークロードを分離し、さまざまなリソースを管理するには、名前空間とワークグループを作成してストレージリソースとコンピューティングリソースを別々に管理できます。

名前空間は、データベースオブジェクトとユーザーのコレクションです。ストレージ関連の名前空間は、スキーマ、テーブル、ユーザー、またはデータを暗号化するための AWS Key Management Service キーをグループ化します。ストレージプロパティには、管理者ユーザーのデータベース名とパスワード、権限、暗号化およびセキュリティが含まれます。名前空間の下にグループ化されたリソースには、データ共有、復旧ポイント、使用制限などもあります。これらのストレージプロパティは、Amazon Redshift Serverless コンソール、AWS Command Line Interface、または特定のリソースの Amazon Redshift Serverless API を使用して設定できます。

ワークグループは、コンピューティングリソースの集合です。コンピューティング関連のワークグループは、RPU、VPC サブネットグループ、セキュリティグループなどのコンピューティングリソースをグループ化します。ワークグループのプロパティには、ネットワークとセキュリティ設定が含まれます。ワークグループの下にグループ化されたリソースには、アクセス制限と使用制限もあります。これらのコンピューティングプロパティは、Amazon Redshift Serverless コンソール、AWS Command Line Interface、または Amazon Redshift Serverless API を使用して設定できます。

1 つ以上の名前空間とワークグループを作成できます。各名前空間に関連付けることができるワークグループは 1 つのみです。逆に、各ワークグループに関連付けることができる名前空間も 1 つのみです。

コンソールを使用したワークグループと名前空間

Amazon Redshift Serverless のセットアップには、いくつかの設定手順を実行する必要があります。手順に従って Amazon Redshift Serverless をセットアップする場合、名前空間とワークグループを作成し、それらを相互に関連付けます。Amazon Redshift Serverless コンソールを使用して Amazon Redshift Serverless 構成の設定を開始するには、[Get started with Amazon Redshift Serverless] (Amazon Redshift Serverless の使用開始) を選択して Amazon Redshift Serverless をセットアップし、その操作を開始します。デフォルト設定の環境を選択すると、セットアップが迅速になります。また、組織の要件に従って設定を明示的に構成することもできます。このプロセスでは、ワークグループと名前空間の設定を指定します。

環境の設定後に、ワークグループのプロパティ および名前空間のプロパティ は設定を理解するのに役立ちます。

AWS Command Line Interface と Amazon Redshift Serverless API を使用したワークグループと名前空間

AWS コンソールを使用する以外にも、AWS CLI、または Amazon Redshift Serverless API を使用して、ワークグループと名前空間を使用できます。ワークグループと名前空間の管理に使用できる API と CLI のオペレーションは、次の表のとおりです。

API オペレーション	CLI コマンド	説明
CreateNamespace	create-namespace	名前空間を作成します。デフォルトでは、Amazon Redshift Serverless はデフォルトの AWS Key Management Service キーを使用して名前空間を作成します。ただし、別のキーを指定してデータを暗号化することもできます。スナップショットを復元して名前空間を作成することもできます。詳細については、「スナップショットと復旧ポイントの使用」を参照してください。
UpdateNamespace	update-namespace	名前空間を更新します。
GetNamespace	get-namespace	名前空間に関する情報を取得します。
ListNamepaces	list-namespaces	名前空間のリストに関する情報を取得します。
DeleteNamespace	delete-namespace	名前空間を削除します。

API オペレーション	CLI コマンド	説明
CreateWorkGroup	create-workgroup	ワークグループを作成します。ワークグループを作成する際は、ワークグループに関連付けることができる既存の名前空間があることを確認します。ワークグループを作成する際、サブネット、セキュリティグループ、RPU などのコンピューティングリソースを指定できます。
UpdateWorkGroup	update-workgroup	ワークグループを更新します。
GetWorkGroup	get-workgroup	ワークグループに関する情報を取得します。
ListWorkGroups	list-workgroups	ワークグループのリストに関する情報を取得します。
DeleteWorkGroup	delete-workgroup	ワークグループを削除します。

ワークグループ

Amazon Redshift Serverless を使用すると、ワークロードやユーザーごとにコンピューティングリソースを分離および制御するワークグループを作成および管理できます。ワークグループを使用すると、メモリや同時実行スケーリングの制限などの設定オプションを設定し、ワークロード全体でクエリの実行を優先できます。コンピューティング関連のワークグループは、RPU、VPC サブネットグループなどのコンピューティングリソースをグループ化します。

名前空間を伴うワークグループの作成

次のステップを実行して、ワークグループを作成します。ワークグループ設定の詳細については、「ワークグループのプロパティ」を参照してください。

1. [Serverless dashboard] (サーバーレスダッシュボード) を選択します。[Create workgroup] (ワークグループの作成) を選択します。
2. ワークグループ名を入力します。
3. ワークグループの [IP アドレスタイプ] を選択します。選択肢は次のとおりです。

   - [IPv4] – このオプションでは、AWS リソースは IPv4 アドレス指定プロトコルを介してのみ通信します。
   - [デュアルスタックモード] – このオプションを使用すると、AWS リソースは IPv4, IPv6、またはその両方のアドレス指定プロトコルを介して通信できます。また、IPv6 CIDR ブロックを Amazon VPC のワークグループに使用される VPC とサブネットに関連付ける必要があります。Amazon VPC コンソールを使用して Amazon VPC を作成するか、既存の Amazon VPC を更新して IPv6 アドレス指定を使用できます。詳細については、「Amazon VPC ユーザーガイド」の「VPC の IPv6 サポート」を参照してください。

4. Amazon Redshift Serverless の [Virtual Private Cloud (VPC)] (仮想プライベートクラウド (VPC)) を選択します。これにより、AWS 環境で特定の仮想ネットワークにワークグループが割り当てられます。[デュアルスタックモード] を使用する場合、選択する Amazon VPC は IPV6 アドレス指定をサポートしている必要があります。Amazon VPC の詳細については、「VPC とサブネットの概要」を参照してください。
5. 1 つ以上の VPC セキュリティグループを選択します。詳細については、「Control traffic to resources using security groups」(セキュリティグループを使用してリソースへのトラフィックを制御する) を参照してください。
6. 自動最適化用に追加のコンピューティングリソースを有効にするかどうかを選択します。詳細については、「Amazon Redshift データベースデベロッパーガイド」の「データベースの自動最適化のための追加のコンピューティングリソースの割り当て」を参照してください。
7. [Subnet] (サブネット) で、データベースに関連付けるサブネットを 1 つ以上指定します。これらのサブネットは、以前に選択した Amazon VPC に含まれており、3 つの個別のアベイラビリティゾーンに属している必要があります。詳細については、「Amazon Redshift サーバーレスを使用する場合の考慮事項」を参照してください。
8. 要件に合ったベース RPU 容量を選択します。

名前空間の選択

1. [Create a new namespace] (新しい名前空間の作成) を選択して名前空間名を入力するか、[Add to an existing namespace] (既存の名前空間に追加) を選択し、ドロップダウンリストから名前空間を選択します。

2. [Database name and password] (データベース名とパスワード) で、最初のデータベース名を指定します。[Admin user credentials] (管理者ユーザーの認証情報) を編集してデフォルトのコンソール管理者以外の管理者を指定することもできます。
3. [Permissions] (アクセス許可) をでは、[Associate IAM role] (IAM ロールを関連付ける) を選択して特定の IAM ロールを名前空間とワークグループに関連付けます。IAM ロールと Amazon Redshift の関連付けの詳細については、「Amazon Redshift での Identity and Access Management」を参照してください。
4. 新しいキーを作成するか、デフォルト以外のキーを選択して、暗号化設定をカスタマイズできます。監査ログを使用するために、エクスポートするログを選択します。ログタイプごとに、異なるメタデータが指定されます。[Continue] (続行) を選択して選択内容を確認します。

ワークグループの選択を確認する

1. [Review and create] (確認と作成) で設定を確認します。前の手順で選択した設定が表示されます。
2. [保存] を選択します。

ワークグループを作成すると、ワークグループが [Workgroups] (ワークグループ) リストに追加されます。

ワークグループのプロパティを表示する

Amazon Redshift Serverless でのワークグループとは、使用可能なコンピューティングリソースのコレクションです。Amazon Redshift Serverless を選択すると、AWS コンソールで、ナビゲーションメニューから [Workgroup configuration] (ワークグループの設定) を選択し、リストを表示できます。[Search] (検索) ボックスを使用して、検索条件を満たすワークグループを検索できます。各ワークグループエントリには、いくつかのプロパティが表示されます。

- [Workgroup] (ワークグループ) -ワークグループの名前。これを選択すると、ワークグループのプロパティを表示および編集できます。
- [Status] (ステータス) -ワークグループが使用可能かどうかを示します。
- [Namespace] (名前空間) - ワークグループに関連付けられた名前空間。各ワークグループは、1 つの名前空間に関連付けられます。
- [作成日] - ワークグループが作成された日付 (UTC)。
- タグ - ワークグループに関連付けられたタグ

さらに、[ワークグループの設定] には、他にもマネージドワークグループ (AWS Glue によって管理されている Amazon Redshift Serverless ワークグループ) のリストが表示されます。マネージドワークグループの詳細については、「Amazon Redshift データベースデベロッパーガイド」の「マネージドワークグループ」を参照してください。

ワークグループのプロパティ

ワークグループを一覧表示するには、左のメニューで [Workgroup configuration] (ワークグループの設定) を選択します。次に、リストからワークグループを選択できます。いくつかのパネルに、ワークグループのプロパティが表示されます。アクションを実行することもできます。[General information] (一般情報) セクションには、以下の項目が表示されます。

- [Workgroup] (ワークグループ) -ワークグループの名前。
- [Namespace] (名前空間) - ワークグループに関連付けられた名前空間。これを選択して、そのプロパティを表示できます。ワークグループは、1 つの名前空間に関連付けられます。
- [Date created] (作成日) -ワークグループが作成された日時。
- [Status] (ステータス) -ワークグループリソースが使用可能かどうかを示します。使用可能な場合は、クライアントと Amazon Redshift Serverless インスタンスに接続して、データをクエリしたり、データベースリソースを作成したり、クエリエディタ v2 で接続できます。
- [Endpoint] (エンドポイント) - URL。
- [JDBC URL] - JDBC クライアント接続を確立するための URL。この URL を使用して、Amazon Redshift の JDBC ドライバーに接続できます。詳しくは、「Amazon Redshift での JDBC ドライバーバージョン 2.x による接続の構成」を参照してください。
- [ODBC URL] - ODBC クライアント接続を確立するための URL。これには、データベースやユーザー ID などのプロパティ、およびそれらの値が含まれます。
- ワークグループバージョンとパッチバージョン - Amazon Redshift Serverless は、定期的に新しいバージョンやパッチをリリースしています。ワークグループバージョン番号とパッチバージョン番号を使用して、Amazon Redshift Serverless ワークグループのソフトウェア更新を追跡できます。特定のパッチの変更点と機能の詳細については、「Amazon Redshift のクラスターバージョン」を参照してください。
- ワークグループ ARN - ワークグループの Amazon リソースネーム。
- 自動最適化用の追加のコンピューティングリソース - 使用量が多い期間だとしても、Amazon Redshift が追加のコンピューティングリソースを割り当てて自動最適化を実行するかどうか。詳細については、「Amazon Redshift データベースデベロッパーガイド」の「データベースの自動最適化のための追加のコンピューティングリソースの割り当て」を参照してください。

[Data access] (データアクセス) タブには、いくつかのパネルがあります。

- [ネットワークとセキュリティ] - [仮想プライベートクラウド (VPC)] 識別子、[VPC セキュリティグループ] リスト、[拡張された VPC ルーティング]、[IP アドレスタイプ]、[パブリックアクセス可能] 設定など、ネットワークプロパティを表示できます。[Edit] (編集) を選択すると、これらの設定を変更できます。また、[Turn on enhanced VPC routing] (拡張された VPC ルーティングを有効にする) を選択して、サーバーレスデータベースとデータリポジトリ間のネットワークトラフィックを VPC 経由でルーティングし、プライバシーとセキュリティを強化できます。[Turn on Public Accessible] (パブリックアクセス可能をオンにする) を選択すると、VPC の外部からデータベースがパブリックにアクセス可能になり、インスタンスとデバイスの接続が可能になります。

  [IP アドレスタイプ] を [デュアルスタックモード] に設定すると、IPv4 と IPv6 の両方のワークグループへのアクセスを同時にサポートできます。ネットワークレイヤー通信インターネットプロトコル (IP) の詳細については、Wikipedia の「インターネットプロトコル」を参照してください。
- [Redshift managed VPC endpoints] (Redshift マネージド VPC エンドポイント) - マネージド VPC エンドポイントを作成して、別の VPC から Amazon Redshift Serverless にアクセスできます。

[Limits] (制限) タブには、Amazon Redshift Serverless の容量と使用制限を制御するための設定があります。これには以下のパネルが含まれています。

- [Base capacity in Redshift processing units (RPUs)] (Redshift プロセッシング単位 (RPU) の基本キャパシティ) – ワークロードの処理に使用するコンピューティングリソースの基本容量を設定できます。詳細については、「Amazon Redshift Serverless 容量を計算する」を参照してください。
- 使用制限 - Amazon Redshift Serverless インスタンスが一定期間に使用できる最大コンピューティングリソースの制限を最大 4 つに設定して、この制限に達した場合に Amazon Redshift Serverless が実行すべきアクションを選択できます。例えば、ワークグループに 2 つの制限、1 つは 500 RPU 時間、もう 1 つは 900 RPU 時間などを設定できます。最初の制限の 500 RPU 時間に達すると、Amazon Redshift Serverless にアラートを送信させて、2 番目の制限の 900 時間に達した場合には、ユーザーのクエリをオフにすることができます。このような制限は、コスト管理や予測の向上につながります。
- [Query limits] (クエリの制限) -タイムアウト設定など、クエリに制限を設定できます。これらの制限は、コストとパフォーマンスの最適化に役立ちます。

[タブ] タブには、[タグ] パネルがあり、ワークグループのためにユーザーが作成したタグがすべて表示されます。 リソースのタグ付けの詳細については、「」を参照してくださいAmazon Redshift Serverless のリソースへのタグ付け

マネージドワークグループのプロパティ

[マネージドワークグループ] リストで、AWS Glue Data Catalog によって管理されているワークグループを選択することもできます。

マネージドワークグループには、通常のワークグループとは異なるプロパティがあります。マネージドワークグループの詳細については、「Amazon Redshift データベースデベロッパーガイド」の「マネージドワークグループ」を参照してください。

[General information] (一般情報) セクションには、以下の項目が表示されます。

- [ワークグループ] - マネージドワークグループの名前。
- [作成日] - マネージドワークグループが作成された日付 (UTC)。
- [カタログ ARN] - AWS Glue Data Catalog のマネージドワークグループの Amazon リソースネーム (ARN)。
- [ステータス] - マネージドワークグループのコンピューティングリソースが使用可能かどうかを示します。リソースが利用可能な場合は、マネージドワークグループを使用するカタログに Apache Iceberg 互換の SQL クライアントで接続して、データのクエリやデータベースリソースの作成を行うことができます。Amazon Redshift クエリエディタ v2 を使用してカタログに接続することもできます。

[クエリとデータベースのモニタリング] には、[マネージドワークグループのパフォーマンス] グラフがあり、ワークグループからのすべてのクエリの平均経過時間が経時的に表示されます。

[クエリ履歴] タブは、マネージドワークグループからのすべてのクエリのリストです。その詳細には、クエリを実行したユーザー、クエリの送信元のクライアントエンジン、クエリの ID とステータスなどの情報が含まれます。[ユーザー] タブは、ワークグループ内のすべてのユーザーのリストです。[パフォーマンスメトリクス] タブには、平均クエリ時間、完了したクエリの数、ストレージ容量の消費率など、さまざまなメトリクスが表示されます。

ワークグループの削除

コンソールを使用してワークグループを削除できます。これを行う前に、データをバックアップとスナップショットが適切であることを確認します。ワークグループの一部として削除されたリソースは、多くの場合、取得できません。

以下のステップを実行します。

1. [Amazon Redshift Serverles]、[Workgroup configuration] (ワークグループの設定)、[Delete Amazon Redshift Serverless instance] (Amazon Redshift Serverless インスタンスの削除) を選択します。

2. ダイアログが開きます。ワークグループの削除を選択すると、すべての使用制限、すべての VPC エンドポイント、VPC エンドポイントへのアクセスが削除されます。

   確認のため、delete と入力し、[Delete] (削除) を選択します。

ステップが完了すると、ワークグループのステータスが [Deleting] (削除中) になり、ワークグループを削除中であることがバナーに表示されます。削除プロセスの進行中に、[Serverless dashboard] (サーバーレスダッシュボード) の一部の機能は無効になります。ただし、[Provisioned clusters dashboard] (プロビジョニングされたクラスターダッシュボード) で、プロビジョニングされたクラスターを設定できます。

ワークグループを削除すると、そのワークグループ名前空間と一緒に表示されません。[Create workgroup] (ワークグループの作成) ボタンを選択して新しいものを作成することもできます。

既存のワークグループを削除し、別の設定の新しいワークグループを同じ名前空間に関連付けることができます。新しいワークグループを作成する場合は、名前空間に関連付けられたデータのサイズで動作する基本容量を選択します。

カスタマー管理キー (CMK) で作成された名前空間に、ワークグループを関連付けることができます。AWS KMS の詳細については、「AWS KMS の概念」を参照してください。

名前空間

Amazon Redshift Serverless では、名前空間はデータベースオブジェクトの論理コンテナを定義します。テーブル、ワークグループ、およびその他のデータベースリソースを保持できます。ワークグループと名前空間を作成しておらず、Amazon Redshift Serverless を使い始める方法を示す手順を探している場合は、「Amazon Redshift Serverless を初めてセットアップする」を参照してください。

名前空間のプロパティ

Amazon Redshift Serverless では、名前空間はデータベースオブジェクトのコンテナを定義します。ナビゲーションリストで [Namespace configuration] (名前空間の設定) を選択し、リストから名前空間を選択して、その設定を編集します。

名前空間の一般情報には、次の情報が含まれています。

- [Namespace] (名前空間) -名前。
- [Namespace ID] (名前空間 ID) - 一意の識別子。
- [ARN] - リソースを指定するために使用される一意の識別子AWS。これには、リージョンやサービスなどのプロパティが含まれています。
- [Status] (ステータス) - [Available] (使用可能) などのステータス。
- [作成日] - 名前空間が作成された日付 (UTC)。
- [Storage used] (使用するストレージ) -名前空間とそのすべてのオブジェクトによって使用されるストレージ領域。
- [Admin user name] (管理者ユーザー名) -管理者アカウント。これは通常、名前空間の作成に使用されるアカウントです。
- [Database name] (データベース名) -名前空間に含まれるデータベースの名前。
- [Total table count] (合計テーブル数) -すべてのスキーマのテーブルの数。

名前空間の追加の設定とプロパティは、複数のタブにあります。これには以下が含まれます。

- [Workgroup] (ワークグループ) - 名前空間に関連付けられているワークグループを表示します。
- [Data back up] (データのバックアップ) -スナップショットの構成と作成、および復旧ポイントの構成が可能です。
- [Security and encryption] (セキュリティと暗号化) - IAM ロールのアクセス許可を管理し、セキュリティと暗号化の設定を表示または編集できます。これには、暗号化キーのステータスおよび監査ログをオンにする設定が含まれます。Amazon Redshift Serverless の監査ログの詳細については、「Amazon Redshift Serverless の監査ログ記録」を参照してください。
- [Datashares] (データ共有) -データ共有を表示します。データ共有を使用すると、データをコピーしたり移動することなく、データへのアクセスを提供できます。データ共有の詳細については、「Amazon Redshift Serverless でのデータ共有」を参照してください。

名前空間の検索

Amazon Redshift メニューで、[Namespaces] (名前空間) リストから選択してプロパティを表示または編集できます。コンソールの情報には、名前空間名、管理者名、およびその他のプロパティが含まれます。

名前空間の設定とプロパティは複数のタブにあります。これには以下が含まれます。

- [Workgroup] (ワークグループ) - 名前空間に関連付けられているワークグループを表示します。

- [Data back up] (データのバックアップ) -スナップショットの構成と作成、および復旧ポイントの構成が可能です。
- [Security and encryption] (セキュリティと暗号化) - IAM ロールのアクセス許可を管理し、セキュリティと暗号化の設定を表示または編集できます。これには、暗号化キーのステータスおよび監査ログ設定が含まれます。
- [Datashares] (データ共有) -データ共有を表示します。

セキュリティと暗号化を編集する

Amazon Redshift Serverless は KMS 暗号化によって保護されています。暗号化設定はコンソールから更新できます。

1. コンソールのメインメニューで [Namespace configuration] (名前空間の設定) を選択し、編集する名前空間を選択して、[Security and encryption] (セキュリティと暗号化) タブで [Edit] (編集) を選択します。ダイアログが表示されます。
2. [暗号化設定のカスタマイズ] を選択し、次に [AWS カスタマーマネージドキーを選択] を選択して、リソースの暗号化に使用するキーを変更します。
3. 監査ログを使用するために、エクスポートするログを選択します。ログタイプごとに、異なるメタデータが指定されます。
4. 変更が完了したら、[Save changes] (変更の保存) を選択します。

名前空間の AWS KMS キーの変更

Amazon Redshift では暗号化により、保管中のデータを保護します。Amazon Redshift Serverless は AWS KMS キー暗号化を使用して、Amazon Redshift Serverless リソースとスナップショットの両方を自動的に暗号化します。ベストプラクティスとして、ほとんどの組織では、保存するデータの種類を確認し、暗号化キーを定期的に更新する計画があります。キーの更新頻度は、データセキュリティのポリシーによって異なります。Amazon Redshift Serverless は、名前空間に対する AWS KMS キーの変更に対応しているため、組織のセキュリティポリシーを順守できます。

AWS KMS キーを変更しても、データは変更されません。

コンソールを使用した AWS KMS キーの変更

Amazon Redshift では暗号化により、保管中のデータを保護します。Amazon Redshift Serverless は AWS KMS キー暗号化を使用して、Amazon Redshift Serverless とスナップショットの両方を自動的に暗号化します。ベストプラクティスとして、ほとんどの組織では、保存するデータの種類を確認

し、暗号化キーを定期的に更新する計画があります。キーの更新頻度は、データセキュリティのポリシーによって異なります。Amazon Redshift Serverless は、名前空間に対する AWS KMS キーの変更に対応しているため、組織のセキュリティポリシーを順守できます。

AWS KMS キーを変更しても、データは変更されません。

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Namespace configuration] (名前空間の設定) を選択します。リストから名前空間を選択します。
3. [Security and encryption] (セキュリティと暗号化) タブで、[Edit] (編集) を選択します。
4. [Customize encryption settings] (暗号化設定のカスタマイズ) を選択し、次に名前空間のキーを選択します。オプションで、新しい VPC を作成することもできます。

AWS CLI を使用した AWS KMS 暗号化キーの変更

`update-namespace` を使用して名前空間の AWS KMS キーを変更します。コマンドの構文を次に示します。

```
aws redshift-serverless update-namespace
--namespace-name
[--kms-key-id <id-of-kms-key>]
// other parameters omitted here
```

名前空間を作成しておく必要があります。そうしないと、CLI コマンドでエラーが発生します。

キーの変更にかかる時間は、Amazon Redshift Serverless のデータ量によって異なります。通常、8 TB の保存データにつき 15 分かかります。

制限事項

カスタマーマネージド KMS キーから AWS KMS キーには変更できません。この場合、新しい名前空間を作成する必要があります。

キーの変更中は、他のアクションを実行することはできません。

名前空間の削除

関連付けられたワークグループを持つ名前空間を削除する場合は、まずワークグループを削除する必要があります。

Amazon Redshift Serverless コンソールで、以下の手順を完了します。

1. 左側のメニューから [Namespace configuration] (名前空間の設定) を選択して削除する名前空間を選択します。
2. [Actions] (アクション) を選択して、[Delete namespace] (名前空間の削除) を選択します。
3. ダイアログボックスが開きます。削除操作を完了する前に、手動のスナップショットを作成することで、データを保持できます。

   確認のため、delete と入力し、[Delete] (削除) を選択します。

Amazon Redshift Serverless でのクエリとワークロードのモニタリング

提供されたシステムビューを使用して、Amazon Redshift Serverless のクエリとワークロードをモニタリングできます。

モニタリングビューは、クエリとワークロードの使用状況を監視するために使用する、Amazon Redshift Serverless のシステムビューです。これらのビューは、`pg_catalog`スキーマ内にあります。用意されたこのシステムビューは、Amazon Redshift Serverless のモニタリングに必要な情報を提供するように設計されており、プロビジョニングされたクラスタ用のものよりもはるかにシンプルです。SYS システムビューは、Amazon Redshift Serverless のために動作するように設計されています。これらのビューで提供される情報を表示するには、SQL SELECT ステートメントを実行します。

システムビューは、以下のモニタリング対象をサポートするように定義されています。

ワークロードモニタリング

以下の目的で、クエリアクティビティを時間の経過とともにモニタリングします。

- ワークロードのパターンを調べて、通常 (ベースライン) とビジネスサービスレベルアグリーメント (SLA) に含まれるものを識別する。
- 一時的な問題によるか、さらなるアクションの理由ともなり得る、平常状態からの逸脱を迅速に特定する。

データのロードとアンロードのモニタリング

Amazon Redshift Serverless においてのデータの出し入れは重要な機能です。データのロードおよびアンロードには COPY および UNLOAD を使用します。また、ビジネス SLA を遵守するた

め、転送されたバイト/行および完了したファイルの進捗状況を注意深くモニタリングする必要があります。通常これは、システムテーブルへのクエリを頻繁に (つまり、毎分) 実行して進行状況を追跡し、顕著な逸脱が検出された場合には、調査や是正アクションを促すためのアラートを発生することによって実現します。

障害と問題の診断

クエリまたはランタイムの障害に対応するために、アクションの実行が必要になる場合があります。デベロッパーは、問題を自己診断し、適切な解決策を決定するために、システムテーブルを利用します。

パフォーマンスチューニング

SLA 要件を満たしていないクエリに対しては、使用開始当初から、あるいは時間の経過とともに性能が低下した際に調整を行う必要があります。調整のためには、実行計画、統計、有効期間、リソース消費量など、ランタイムに関する詳細事項を決定する必要があります。逸脱を起こした原因を特定し、パフォーマンスの改善の方法を探るには、問題のあるクエリにおける基準のデータが必要となります。

ユーザーオブジェクトでのイベントのモニタリング

ユーザーオブジェクトでは、マテリアライズドビューのリフレッシュ、バキュームおよび分析など、アクションやアクティビティをモニタリングする必要があります。これには、マテリアライズドビューの自動更新など、システムが管理するイベントも含まれます。ユーザーが開始したイベントについてはその終了タイミングを、また、システムが開始した場合には最後に正常に実行されたイベントを、それぞれモニタリングします。

課金のための使用状況の追跡

以下の目的のために、時間の経過に伴う使用状況の傾向変化をモニタリングします。

- 予算計画ならびに事業の拡大見通しの報告。
- コールドデータの削除など、潜在的なコスト削減機会の特定。

Amazon Redshift Serverless をモニタリングするには、SYS システムビューを使用します。SYS モニタリングビューの詳細については、「Amazon Redshift データベースデベロッパーガイド」の「SYS モニタリングビュー」を参照してください。

クエリモニタリングポリシーを追加する

スーパーユーザーは、スーパーユーザーではないユーザーにアクセス権を提供して、これらのユーザーがすべてのユーザーに対するクエリモニタリングを実行できるようにすることが可能です。ま

ず、クエリモニタリングのアクセス権を提供するためのポリシーをユーザーまたはロールに追加します。次に、ユーザーまたはロールにクエリモニタリング許可を付与します。

クエリモニタリングポリシーを追加する

1. https://console.aws.amazon.com/iam/ を選択します。
2. [Access management] (アクセス管理) で、[Policies] (ポリシー) を選択します。
3. [Create Policy] (ポリシーの作成) を選択します。
4. [JSON] を選択して、以下のポリシー定義を貼り付けます。

   JSON

```
{
"Version":"2012-10-17",
"Statement": [
{
"Effect": "Allow",
"Action": [
    "redshift-data:ExecuteStatement",
    "redshift-data:DescribeStatement",
    "redshift-data:GetStatementResult",
    "redshift-data:ListDatabases"
],
"Resource": "*"
},
{
"Effect": "Allow",
"Action": "redshift-serverless:GetCredentials",
"Resource": "*"
}
]
}
```

5. [Review policy] (ポリシーの確認) を選択します。
6. [Name] (名前) にポリシーの名前を入力します (例: `query-monitoring`)。
7. [ポリシーの作成] を選択します。

ポリシーを作成したら、適切なアクセス許可を付与できます。

アクセス許可を付与するにはユーザー、グループ、またはロールにアクセス許可を追加します。

- AWS IAM アイデンティティセンター のユーザーとグループ:

  アクセス許可セットを作成します。「AWS IAM アイデンティティセンター ユーザーガイド」の「アクセス許可セットを作成する」の手順に従ってください。

- IAM 内で、ID プロバイダーによって管理されているユーザー:

  ID フェデレーションのロールを作成します。詳細については IAM ユーザーガイド の サードパーティー ID プロバイダー (フェデレーション) 用のロールを作成する を参照してください。

- IAM ユーザー:
  - ユーザーが担当できるロールを作成します。手順については IAM ユーザーガイド の IAM ユーザーのロールの作成 を参照してください。
  - (お奨めできない方法) ポリシーをユーザーに直接アタッチするか、ユーザーをユーザーグループに追加します。「IAM ユーザーガイド」の「ユーザー (コンソール) へのアクセス許可の追加」の指示に従います。

ユーザーにクエリモニタリング許可を付与する

`sys:monitor` 許可を持つユーザーは、すべてのクエリを表示できます。さらに、`sys:operator`許可を持つユーザーは、クエリのキャンセル、クエリ履歴の分析、およびバキューム操作を実行することができます。

ユーザーにクエリモニタリング許可を付与する

1. 以下のコマンドを入力して、システムモニタリングのアクセス権を提供します。user-name は、アクセス権を提供するユーザーの名前です。

   ```
   grant role sys:monitor to "IAM:user-name";
   ```

2. (オプション) 以下のコマンドを入力して、システムオペレーターのアクセス権を提供します。user-name は、アクセス権を提供するユーザーの名前です。

   ```
   grant role sys:operator to "IAM:user-name";
   ```

ロールにクエリモニタリング許可を付与する

sys:monitor 許可を持つロールが設定されたユーザーは、すべてのクエリを表示できます。さらに、sys:operator許可を持つロールが設定されたユーザーは、クエリのキャンセル、クエリ履歴の分析、およびバキューム操作を実行することができます。

ロールにクエリモニタリング許可を付与する

1. 以下のコマンドを入力して、システムモニタリングのアクセス権を提供します。role-name は、アクセス権を提供するロールの名前です。

```
grant role sys:monitor to "IAMR:role-name";
```

2. (オプション) 以下のコマンドを入力して、システムオペレーターのアクセス権を提供します。role-name は、アクセス権を提供するロールの名前です。

```
grant role sys:operator to "IAMR:role-name";
```

RPU 制限の設定を含む使用制限の設定

ワークグループの [Limits] (制限) タブでは、1 つ以上の使用制限を追加して、特定の期間に使用する最大 RPU を管理したり、データ共有の使用制限を設定できます。

1. [Manage usage limits] (使用制限の管理) を選択します。[制限] セクションは、[Compute usage by period] パネルの下部に表示されます。
2. 使用制限の期間を設定します。
3. [頻度] を選択します。頻度は、[毎日]、[毎週]、または [毎月] を選択します。これにより、使用制限の期間が設定されます。このインスタンスで [Daily] (日別) を選択すると、より詳細な制御が可能になります。
4. 使用制限を時間数で設定します。
5. アクションを設定します。これには次のものが含まれます。

   - システムテーブルへのログ記録 - システムビュー SYS_QUERY_HISTORY にレコードを追加します。このビューの usage_limit 列をクエリして、クエリが制限を超えたかどうかを判断できます。

- [Alert] (アラート) -Amazon SNS を使用して通知サブスクリプションを設定し、制限に違反した場合に通知を送信します。新しい Amazon SNS トピックを作成することも、既存のトピックを使用することもできます。
- [Turn off user queries] (ユーザークエリを無効にする) - クエリを無効にして Amazon Redshift Serverless の使用を停止します。また、通知を送信します。

最初の 2 つのアクションは情報提供ですが、最後のアクションはクエリ処理をオフにします。

6. オプションで、[Cross-Region data sharing usage limit] (クロスリージョンでのデータ共有の使用制限) を設定すると、プロデューサーリージョンからコンシューマーリージョンのコンシューマーに転送されるデータ量が制限されます。これを行うには、[Add limit] (制限の追加) を選択して手順を実行します。
7. ページの下部にある [変更の保存] をクリックして制限を保存します。
8. 必要に応じてさらに最大 3 つまで制限を設定します。

RPU と請求に関する概念の詳細については、「Amazon Redshift Serverless の請求」を参照してください。

クエリ制限の設定

ワークグループの [Limits] (制限) タブでは、パフォーマンスと制限をモニタリングするための制限を追加できます。クエリモニタリング制限の詳細については、「WLM クエリモニタリングルール」を参照してください。

1. [Manage query limits] (クエリ制限の管理) を選択します。[Manage query limits] (クエリ制限の管理) ダイアログで [Add new limit] (新しい制限の追加) を選択します。
2. 設定する制限タイプを選択し、対応する制限の値を入力します。
3. [Save changes] (変更の保存) を選択して設定を保存します。

クエリ制限と構成パラメータを変更すると、データベースが再起動します。

クエリキューの設定

Amazon Redshift Serverless は、キューベースのクエリリソース管理をサポートしています。ワークロードごとにカスタマイズされたモニタリングルールを使用して、専用のクエリキューを作成できます。この機能により、リソースの使用状況をきめ細かく制御できます。

クエリのモニタリングルール (QMR) は Redshift Serverless ワークグループレベルでのみ適用され、このワークグループで実行されるすべてのクエリに均一に影響します。キューベースのアプローチでは、個別のモニタリングルールを使用してキューを作成できます。これらのキューは、特定のユーザーロールとクエリグループに割り当てることができます。各キューは独立して動作し、ルールはそのキュー内のクエリにのみ影響します。

キューを使用して、メトリクスベースの述語と自動応答を設定できます。例えば、時間制限を超えた、またはリソースを大量に消費するクエリを自動的に中止するようにルールを設定できます。

考慮事項

サーバーレスキューを使用する場合、次の点を考慮してください。

- Amazon Redshift でプロビジョニングされたクラスターで使用されるワークロード管理 (WLM) 設定キーの、max_execution_time、short_query_queue、auto_wlm、concurrency_scaling、priorit は、Redshift Serverless キューではサポートされていません。

  さらにホップすると、change_query_priority アクションは Serverless ではサポートされていません。
- ホップアクション (キュー間のクエリの移動) は、Amazon Redshift Serverless ではサポートされていません。
- キューの優先順位は、Amazon Redshift でプロビジョニングされたクラスターでのみサポートされます。
- Amazon Redshift Serverless は、パフォーマンスが最適になるようにスケーリングとリソース割り当てを自動的に管理するため、キューの優先順位を手動で設定する必要はありません。

クエリキューの設定

AWS マネジメントコンソール、AWS CLI、または Redshift Serverless API を使用して、サーバーレスワークグループの [制限] タブでキューを作成できます。

Console

サーバーレスワークグループのキューを作成するには、次の手順に従います。

1. Redshift Serverless ワークグループに移動します。
2. [制限] タブを選択します。

3. [クエリキュー] で、[キューを有効にする] を選択します。

> **Important**
>
> クエリキューの有効化は永続的な変更です。一度有効にすると、キューレスモニタリングに戻すことはできません。

4. 次のパラメータを使用してキューを設定します。

   キューレベルのパラメータ

   - `name` - キュー識別子 (必須、一意、空でない)
   - `user_role` - ユーザーロールの配列 (オプション)
   - `query_group` - クエリグループの配列 (オプション)
   - `query_group_wild_card` - 0 またはワイルドカードマッチングを有効にするには 1 (オプション)
   - `user_group_wild_card` - 0 またはワイルドカードマッチングを有効にするには 1 (オプション)
   - `rules` - モニタリングルールの配列 (オプション)

   ルールレベルのパラメータ

   - `rule_name` - 一意の識別子、最大 32 文字 (必須)
   - `predicate` - 条件の配列、1 ~ 3 述語 (必須)
   - `action` - 「中止」または「ログ」 (必須)

   述語レベルのパラメータ

   - `metric_name` - モニタリングするメトリクス (必須)
   - `operator` - 「=」、「<」、または「>」(必須)
   - `value` - 数値のしきい値 (必須)

   制限

   - 最大 8 キュー
   - すべてのキューで最大 25 個のルール

- ルールあたり最大 3 つの述語
- ルール名はグローバルで一意である必要があります。

構成例

3 つのキューの例: 1 つは短いタイムアウトのダッシュボードクエリ用、もう 1 つはタイムアウトが長い ETL クエリ用、1 つは管理者キュー用です。

```
[
  {
    "name": "dashboard",
    "user_role": ["analyst", "viewer"],
    "query_group": ["reporting"],
    "query_group_wild_card": 1,
    "rules": [
      {
        "rule_name": "short_timeout",
        "predicate": [
          {
            "metric_name": "query_execution_time",
            "operator": ">",
            "value": 60
          }
        ],
        "action": "abort"
      }
    ]
  },
  {
    "name": "ETL",
    "user_role": ["data_scientist"],
    "query_group": ["analytics", "ml"],
    "rules": [
      {
        "rule_name": "long_timeout",
        "predicate": [
          {
            "metric_name": "query_execution_time",
            "operator": ">",
            "value": 3600
          }
        ],
```

```
        "action": "log"
      },
      {
        "rule_name": "memory_limit",
        "predicate": [
          {
            "metric_name": "query_temp_blocks_to_disk",
            "operator": ">",
            "value": 100000
          }
        ],
        "action": "abort"
      }
    ]
  },
  {
    "name": "admin_queue",
    "user_role": ["admin"],
    "query_group": ["admin"]
  }
]
```

この例では、以下のようになっています：

- ダッシュボードクエリの実行は、60 秒を超えると中止されます
- ETL クエリは 1 時間以上実行されると口グに記録される
- 管理者キューにリソース制限がない

CLI

CreateWorkgroup API または UpdateWorkgroup API と `wlm_json_configuration` 設定パラメータを使用してキューを管理し、JSON 形式でキューを指定できます。

```
aws redshift-serverless create-workgroup \
  --workgroup-name test-workgroup \
  --namespace-name test-namespace \
  --config-parameters '[{"parameterKey": "wlm_json_configuration", "parameterValue":
 "[{\"name\":\"dashboard\",\"user_role\":[\"analyst\",\"viewer\"],\"query_group
\":[\"reporting\"],\"query_group_wild_card\":1,\"rules\":[{\"rule_name\":
\"short_timeout\",\"predicate\":[{\"metric_name\":\"query_execution_time\",
\"operator\":\">\",\"value\":60}],\"action\":\"abort\"}]},{\"name\":\"ETL
\",\"user_role\":[\"data_scientist\"],\"query_group\":[\"analytics\",\"ml
```

```
\"],\"rules\":[{\"rule_name\":\"long_timeout\",\"predicate\":[{\"metric_name
\":\"query_execution_time\",\"operator\":\">\",\"value\":3600}],\"action\":
\"log\"},{\"rule_name\":\"memory_limit\",\"predicate\":[{\"metric_name\":
\"query_temp_blocks_to_disk\",\"operator\":\">\",\"value\":100000}],\"action\":
\"abort\"}]},{\"name\":\"admin_queue\",\"user_role\":[\"admin\"],\"query_group\":
[\"admin\"]}]"}]'
```

ベストプラクティス

サーバーレスキューを使用する場合は、次のベストプラクティスに留意してください。

- 個別の制限要件 (ETL、レポート、アドホック分析など) を持つワークロードには、個別のキューを使用します。
- 単純なしきい値から始めて、クエリの動作と使用パターンに基づいて調整します。クエリモニタリングルールのシステムテーブルとビューに記載されているテーブルとビューを使用して、クエリの使用パターンをモニタリングできます。

ダッシュボードを使用して Amazon Redshift Serverless 概要データをチェックする

Amazon Redshift Serverless ダッシュボードには、ワークグループと名前空間に関するメトリクスやその他の情報が一目でわかるようにパネルが集められています。これには次のようなものがあります。

- [Resources summary] (リソースの概要) - 使用されているストレージ、その他のメトリックなど、Amazon Redshift Serverless に関する情報の概要を表示します。
- クエリの概要 - 完了したクエリや実行中のクエリなど、クエリに関する情報を表示します。[View details] (詳細を表示) をクリックして、追加のフィルターがある画面に移動します。
- RPU 容量 - 例えば、過去 10 時間など、特定の期間に使用された全体の容量を表示します。
- データ共有 - データ共有数を表示します。データ共有は、例えば AWS アカウントのデータの共有に使用されます。メトリクスは、認可が必要なデータ共有やその他の情報を示します。
- Total compute usage - 選択した期間にわたって、選択したワークグループで利用された合計 RPU 時間を最大過去 7 日間まで表示します。

これらの利用できるメトリクスの詳細をダッシュボードからすばやく調べることで、Amazon Redshift Serverless に関する詳細の確認、クエリの調査、作業項目の追跡などが可能となります。

Amazon Redshift Serverless の監査ログ記録

Amazon Redshift Serverless を設定することで、接続、ユーザー、およびユーザーアクティビティに関するログデータを、Amazon CloudWatch Logs のロググループにエクスポートできます。Amazon CloudWatch Logs を使用すると、ログデータのリアルタイム分析や、CloudWatch を使用したアラームの作成、メトリクスの表示を行うことができます。CloudWatch Logs を使用して、非常に耐久性が高いストレージにログレコードを格納できます。

Amazon Redshift コンソールで CloudWatch アラームを作成して、メトリクスを追跡できます。アラームの作成の詳細については、「アラームの管理」を参照してください。

生成されたログデータを Amazon CloudWatch Logs にエクスポートするには、コンソールの Amazon Redshift Serverless 設定で、エクスポート対象のログを選択する必要があります。これを行うには、[セキュリティと暗号化] の [名前空間の設定] を選択します。

CloudWatch でのログイベント

エクスポートする Redshift ログを選択すると、Amazon CloudWatch Logs でイベントをモニタリングできるようになります。Amazon Redshift Serverless のための新しいロググループが自動的に作成されます。ここで `log_type` はログタイプを表します。

```
/aws/redshift/<namespace>/<log_type>
```

最初のワークグループと名前空間を作成するときのデフォルトは、名前空間名です。ロググループ名は、名前空間の名前によって異なります。

例えば、接続ログをエクスポートする場合、ログデータは次のロググループに保存されます。

```
/aws/redshift/default/connectionlog
```

ログイベントをロググループに対しエクスポートする際には、サーバーレスログストリームが使用されます。この場合の動作は、次の条件のうちのどれが true であるかによって異なります。

- 指定された名前のロググループが存在する。Redshift は、既存のロググループを使用してログデータをエクスポートします。事前定義されたログ保持期間、メトリクスフィルター、カスタムアクセスを持つロググループを作成するために、AWSCloudFormation で提供されているものをはじめとした自動設定機能を使用できます。
- 指定された名前のロググループが存在しない。対称となるログエントリがインスタンスのログで検出されると、Amazon Redshift Serverless は自動的に、Amazon CloudWatch Logs 内で新しいログ

グループを作成します。このロググループは、デフォルトの失効なしのログ保持期間を使用します。ログの保持期間を変更するには、Amazon CloudWatch Logs コンソール、AWS CLI、または Amazon CloudWatch Logs API を使用します。CloudWatch Logs でログの保持期間を変更する方法については、「ロググループとログストリームの操作」の「CloudWatch Logs でログデータ保管期間を変更する」を参照してください。

ログイベント内で情報を検索するには、Amazon CloudWatch Logs コンソール、AWS CLI、または Amazon CloudWatch Logs API を使用します。検索およびログデータのフィルタ処理の詳細については、「ログデータの検索およびフィルタ処理」を参照してください。

CloudWatch メトリクス

Amazon Redshift Serverless メトリクスは、コンピューティングメトリクス、データ、ストレージメトリクスに分割され、それぞれワークグループディメンションセットと名前空間ディメンションセットに分類されます。ワークグループおよび名前空間の詳細については、「ワークグループと名前空」を参照してください。

CloudWatch のコンピューティングメトリクスは以下のとおりです。

メトリクス名	単位	説明	ディメンションのセット
QueriesCompletedPerSecond	クエリ件数	1 秒ごとに完了したクエリの数。	{Database, LatencyRange, Workgroup}, {LatencyRange, Workgroup}
QueryDuration	マイクロ秒	クエリを完了するまでの平均時間。	{Database, LatencyRange, Workgroup}, {LatencyRange, Workgroup}
QueriesRunning	クエリ件数	ある時点における実行中のクエリの数。	{Database , QueryType , Workgroup

メトリクス名	単位	説明	ディメンションのセット
			}, {QueryType, Workgroup}
QueriesQueued	クエリ件数	ある時点におけるキュー内のクエリの数。	{Database, QueryType, Workgroup}, {QueryType, Workgroup}
DatabaseConnections	接続の数	ある時点におけるデータベースへの接続の数。	{Database, Workgroup}, {Workgroup}
QueryRuntimeBreakdown	ミリ秒	クエリステージごとの、クエリが実行された合計時間。	{Database, Stage, Workgroup}, {Stage, Workgroup}
ComputeCapacity	RPU	過去 30 分間に割り当てられたコンピューティングユニットの、最も近い整数に切り上げられた平均数。	{Workgroup}
ComputeSeconds	RPU 秒	過去 30 分間に使用されたコンピューティングユニットの累積秒数。	{Workgroup}

メトリクス名	単位	説明	ディメンションのセット
QueriesSucceeded	クエリ件数	過去 5 分間に成功したクエリの数。	{Database, QueryType, Workgroup}, {QueryType, Workgroup}
QueriesFailed	クエリ件数	過去 5 分間に失敗したクエリの数。	{Database, QueryType, Workgroup}, {QueryType, Workgroup}

メトリクス名	単位	説明	ディメンションのセット
UsageLimitAvailable	RPU 時間または TB	UsageType に応じて、UsageLimitAvailable は次の値を返します。 • UsageType が SERVERLESS_COMPUTE の場合、UsageLimitAvailable は、ワークグループが指定した制限内でクエリできる RPU 時間の残り数を返します。 • UsageType が CROSS_REGION_DATASHARING の場合、UsageLimitAvailable は、指定された制限内でお客様がスキャンできる残りの TB 数を返します。	{UsageLimitId, UsageType, Workgroup}

メトリクス名	単位	説明	ディメンションのセット
UsageLimitConsumed	RPU 時間または TB	UsageType に応じて、UsageLimitConsumed は次の値を返します。 • UsageType が SERVERLESS_COMPUTE の場合、UsageLimitConsumed は、指定された制限内でワークグループが既にクエリした RPU 時間数を返します。 • UsageType が CROSS_REGION_DATASHARING の場合、UsageLimitConsumed は、指定された制限内でお客様がスキャンに既に使用した	{UsageLimitId, UsageType, Workgroup}

メトリクス名	単位	説明	ディメンションのセット
		TB 数を返します。	
ExtraComputeForAutomaticOptimizationChargedSeconds	RPU 秒	過去 30 分間の自動最適化オペレーションに対して課金されたコンピューティング単位の秒数。	{Workgroup}

CloudWatch のデータおよびストレージメトリクスは以下のとおりです。

メトリクス名	単位	説明	ディメンションのセット
TotalTableCount	テーブルの数	特定の時点でのテーブルの総数。この合計には、永続テーブル、一時テーブル、データ共有テーブル、外部テーブル、およびマテリアライズドビューが含まれます。ビューおよびシステムテーブルはこの数には含まれません。	{Database, Namespace}
DataStorage	Megabytes	Redshift データにディスクまた	{Namespace}

メトリクス名	単位	説明	ディメンションのセット
		はストレージ領域で使用されるメガバイト数。	

ServerlessUniqueSnapshotStorage メトリクスは名前空間およびワークグループに依存しません。CloudWatch の ServerlessUniqueSnapshotStorage メトリクスを示します。

メトリクス名	単位	説明	ディメンションのセット
ServerlessUniqueSnapshotStorage	Megabytes	すべてのアクティブな手動スナップショットに保存された一意のデータブロックのメガバイト数。	{}

ディメンションのセットは、メトリクスに適用されるグループ化ディメンションです。これらのディメンショングループを使用して、統計の取得方法を指定できます。

以下は、特定のメトリクスに対するディメンションとディメンション値を詳しく説明した表です。

ディメンション	説明と値
DatabaseName	データベースの名前。カスタム値。
Latency	可能な値は以下のとおりです。 • Short – 10 秒未満 • medium – 10 秒 ~ 10 分 • Long – 10 分超過

ディメンション	説明と値
QueryType	指定できる値は INSERT、DELETE、UPDATE、UNLOAD、LOAD、SELEC および OTHER です。
stage	クエリの実行ステージ。可能な値は以下のとおりです。 • QueryPlanning: SQL ステートメントの解析と最適化にかかった時間。 • QueryWaiting: WLM キューの待機にかかった時間。 • QueryExecutingRead: 読み込みクエリの実行にかかった時間。 • QueryExecutingInsert: 挿入クエリの実行にかかった時間。 • QueryExecutingDelete: 削除クエリの実行にかかった時間。 • QueryExecutingUpdate: 更新クエリの実行にかかった時間。 • QueryExecutingCtas: テーブル作成クエリの実行にかかった時間。 • QueryExecutingUnload: アップロードクエリの実行にかかった時間。 • QueryExecutingCopy: コピークエリの実行にかかった時間。 • QueryCommit: コミットにかかった時間。
Namespace	名前空間の名前 カスタム値。
Workgroup	ワークグループの名前。カスタム値。
UsageLimitId	使用制限の識別子。

ディメンション	説明と値
UsageType	制限される Amazon Redshift Serverless 機能。可能な値は以下のとおりです。 • SERVERLESS_COMPUTE • CROSS_REGION_DATASHARING

スナップショットと復旧ポイント

Amazon Redshift Serverless のバックアップは、名前空間内のオブジェクトとデータをポイントインタイムの表現です。バックアップには、手動で作成されるスナップショットと、Amazon Redshift Serverless が自動的に作成する復旧ポイントの 2 種類があります。

Amazon Redshift Serverless は、30 分ごと、またはノードあたり 5 GB のデータ変更があった時点のいずれか早い方で復旧ポイントを自動的に作成します。大規模なデータセット (5 GB × ノード数を超える) の場合、復旧ポイント間の最小間隔は 15 分です。すべての復旧ポイントは 24 時間保持されます。

> **Note**
>
> 独自のスナップショットスケジュールを作成して復旧ポイントの作成タイミングを制御することはできません。

Amazon Redshift Serverless は、Redshift マネージドストレージ (RMS) にスナップショットを作成します。詳細については、「Amazon Redshift Serverless 容量を計算する」を参照してください。

> **Note**
>
> RG または RA3 でプロビジョニングされたクラスターと Amazon Redshift Serverless ワークグループでは、バックアップ用ではないテーブルはサポートされていません。RG または RA3 クラスターおよびサーバーレスワークグループでバックアップしないとマークされたテーブルは、スナップショットの作成中に常にバックアップされ、スナップショットから復元するときに復元される永続テーブルとして扱われます。バックアップ用ではないテーブルのスナップショットコストを回避するには、スナップショットを作成する前にそれらを切り捨てます。

スナップショットまたは復旧ポイントのデータを取得する場合は、サーバーレス名前空間またはプロビジョン済みのクラスターにスナップショットを復元できます。スナップショットを復元するには、次の 3 つのシナリオがあります。

- サーバーレススナップショットをサーバーレス名前空間に復元します。
- サーバーレススナップショットをプロビジョニングされたクラスターに復元します。
- プロビジョニングされたクラススナップショットをサーバーレス名前空間に復元します。

プロビジョン済みのクラスターにサーバーレス スナップショットを復元する場合は、使用するノードタイプ (RG または RA3 など) とノードの数を選択する必要があります。これにより、クラスターまたはノードレベルで設定を制御できます。

プロビジョニングされたクラスターのスナップショットをサーバーレス名前空間に復元するには、Redshift プロビジョニングコンソールから開始し、復元するスナップショットを選択して、[Restore from snapshot] (スナップショットからの復元)、[Restore to serverless namespace] (サーバーレス名前空間に復元する) を選択します。プロビジョニングされたクラススナップショットをサーバーレス名前空間に復元すると、Amazon Redshift はインターリーブキーを含むテーブルを複合ソートキーに変換します。ソートキーの詳細については、「ソートキーの使用」を参照してください。

さらにコンテキストを追加したい場合は、スナップショットとリカバリポイントにメタデータと情報を提供するキーと値のペアでスナップショットとリカバリポイントにタグを付けることができます。リソースへのタグ付けの詳細については、「リソースのタグ付けの概要」を参照してください。

最後に、スナップショットを他の AWS アカウントと共有することもできます。これにより、スナップショット内のデータにアクセスし、クエリを実行できるようになります。

AWS Backup の統合

AWS Backup を使用してスナップショットを作成および復元することもできます。Backup は、AWS のサービス、クラウド、オンプレミスにわたってデータ保護の一元化と自動化に役立つフルマネージドサービスです。詳細については、「AWS Backup と Amazon Redshift の統合」を参照してください。AWS Backup の詳細については、「AWS Backup Developer Guide」の「What is AWS Backup?」を参照してください。

スナップショットの作成

> **Note**
>
> RG または RA3 でプロビジョニングされたクラスターと Amazon Redshift Serverless ワークグループでは、バックアップ用ではないテーブルはサポートされていません。RG または RA3 クラスターおよびサーバーレスワークグループでバックアップしないとマークされたテーブルは、スナップショットの作成中に常にバックアップされ、スナップショットから復元するときに復元される永続テーブルとして扱われます。バックアップ用ではないテーブルのスナップショットコストを回避するには、スナップショットを作成する前にそれらを切り捨てます。

スナップショットを作成するには、以下の手順を実行します。

スナップショットを作成する方法

1. Amazon Redshift Serverless コンソールで、[Data backup] (データのバックアップ) を選択します。
2. [Create snapshot] (スナップショットを作成) を選択します。
3. スナップショットを作成する名前空間を選択します。
4. スナップショット ID を入力します。
5. (オプション) 保持期間を選択します。[カスタム値] を選択した場合は、日数を選択します。選択する値は、1 ~ 3,653 日の間でなければなりません。デフォルトは [retain indefinitely] (無期限に保持) です。
6. [作成] を選択します。

名前空間の設定からスナップショットを作成するには

1. Amazon Redshift Serverless コンソールで、[Namespace configuration] (名前空間の設定) を選択します。
2. スナップショットを作成する名前空間を選択します。作成できるのは、ワークグループに関連付けられ、ステータスが [Available] (使用可能) の名前空間のスナップショットのみです。
3. [Data backup] (データバックアップ) タブを選択します。
4. [スナップショットを作成] を選択します。

5. スナップショット ID を入力します。
6. (オプション) 保持期間を選択します。[カスタム値] を選択した場合は、日数を選択します。選択する値は、1 ~ 3,653 日の間でなければなりません。
7. [作成] を選択します。

最終スナップショットの作成

名前空間を削除する前に、名前空間内のすべてのデータの最終スナップショットを作成するには、以下の手順に従います。

最終スナップショットを作成するには

1. Amazon Redshift Serverless コンソールで、[Namespace configuration] (名前空間の設定) を選択します。
2. 削除する名前空間を選択します。
3. [アクション]、[削除] の順に選択します。
4. [Create final snapshot] (最終スナップショットの作成) を選択します。
5. スナップショットの名前を入力します。
6. delete (削除) と入力します。
7. [削除] を選択します。

スナップショットを共有またはスナップショットのアクセス許可を削除

スナップショットを別の AWS アカウントと共有したり、スナップショットへのアカウントのアクセス権を削除したりするには、以下の手順に従います。

スナップショットへのアクセスを共有または削除するには

1. Amazon Redshift Serverless コンソールで、[データのバックアップ] を選択します。
2. 共有するスナップショットを選択します。
3. [Action] (アクション)、[Manage access] (アクセスの管理) の順にクリックします。
4. 別のアカウントとスナップショットを共有するには、[AWS アカウント ID] を入力します。アカウントからアクセス権を削除するには、[削除] を選択します。
5. [Save changes] (変更の保存) をクリックします。

スナップショットをスケジュールする

スナップショットを作成するタイミングを正確に制御するために、特定の名前空間のスナップショットスケジュールを作成できます。スナップショットの作成をスケジュールする場合、1 回限りのイベントを作成することも、Unix の cron 式を使用して定期的なスケジュールを作成することもできます。Cron 式では、空白文字区切りの 3 つのフィールドをサポートします。

```
cron(Minutes Hours Day-of-month Month Day-of-week Year)
```

フィールド	値	ワイルドカード
分	0 ~ 59	, - * /
時間	0 ~ 23	, - * /
日	1 ~ 31	, - * ? / L W
月	1 ~ 12 または JAN ~ DEC	, - * /
曜日	1 ~ 7 または SUN ~ SAT	, - * ? L #
年	1970 ~ 2199	, - * /

ワイルドカード

- ワイルドカード [,] (カンマ) には追加の値が含まれます。Day-of-week フィールドの、MON,WED,FRI は、月曜日、水曜日、金曜日を含みます。合計値はフィールドあたり 24 に制限されています。
- ワイルドカード [-] (ダッシュ) は範囲を指定します。Hour フィールドの、「1 ~ 15」は、指定した日の 1 時間から 15 時間を含みます。
- ワイルドカード [*] (アスタリスク) にはフィールドのすべての値が含まれます。Hours フィールドの、* にはすべての時間が含まれています。
- ワイルドカード [/] (スラッシュ) で増分を指定します。Hours フィールドで、**1/10** と入力して、その日の最初の時間から始めて、10 時間毎を指定できます (01:00、11:00、21:00 など)。
- [?] (疑問符) のワイルドカードは、任意を意味します。Day-of-month フィールドで 7 と入力し、7 日が何曜日であってもかまわない場合、Day-of-week フィールドに ? を入力できます。

- Day-of-month フィールドまたは Day-of-week フィールドにある [L] のワイルドカードは、月または週の最終日を指定します。
- Day-of-month フィールドの、ワイルドカード W は、平日を指定します。Day-of-month フィールドで、3W は月の 3 番目の平日に最も近い日を指定します。
- Day-of-week フィールドの # ワイルドカードは、月の指定された曜日の特定のインスタンスを指定します。例えば、3#2 は、月の第 2 火曜日を示します。3 は週の 3 番目の日 (火曜日) を示し、2 は月のそのタイプの 2 番目の日を示します。

> **Note**
>
> 「#」文字を使用する場合、曜日フィールドには 1 つの式しか定義できません。例えば、「3#1,6#3」は 2 つの式として解釈されるため、無効です。

制限

- Cron 式の Day-of-month フィールドと Day-of-week フィールドを同時に指定することはできません。一方のフィールドに値を指定する場合、もう一方のフィールドで [?] (疑問符) を使用する必要があります。
- スナップショットスケジュールは以下の頻度をサポートしていません。
  - 1 時間に 1 回を超える頻度でスケジュールされるスナップショット。
  - 1 日 (24 時間) に 1 回未満の頻度でスケジュールされるスナップショット。

  1 時間以内にスナップショットをスケジュールする結果になる重複したスケジュールがある場合、検証エラーが発生します。

cron 文字列の例は、次の表のとおりです。

分	時間	曜日	意味			
0	14-20/1	火	毎週火曜日の午後 2 時から午後 8 時の間。			
0	21	MON-FRI	毎晩、月曜日 ~ 金曜日の午後9時。			

分	時間	曜日	意味			
30	0/6	土 - 日	土曜日と日曜日は、その日の深夜 30 分過ぎ (00:30) から、6 時間ごとに増分されます。これにより、各日とも [00:30、06:30、12:30、および 18:30] にスナップショットが作成されます。			
30	12/4	*	毎日 12:30 から 4 時間ごとに増分します。これにより [12:30、16:30、20:30] となります。			

毎日 15:15 から 2 時間ずつ実行するスケジュールを作成する方法は、次の例のとおりです。

```
cron(15 15/2 *)
```

スナップショットスケジュールの作成には、Amazon Redshift Serverless コンソール、API、または AWS CLI を使用できます。

スナップショットをスケジュールするには

1. Amazon Redshift Serverless コンソールで、[データのバックアップ] を選択します。
2. [スナップショットスケジュール] を選択します。
3. [スケジュールを作成] を選択します。
4. スナップショットスケジュールの名前を入力します。
5. スナップショットを作成する名前空間を選択します。
6. スケジュールの cron 式を入力するか、スケジュールビルダーを使用して作成します。
7. (オプション) 保持期間を選択します。[カスタム値] を選択した場合は、日数を指定します。

8. [スケジュールを作成] を選択します。

スナップショットの保持期間を更新する

スナップショットの保持期間を更新するには、次の手順を実行します。

スナップショットの保持期間を更新するには

1. Amazon Redshift Serverless コンソールで、[データのバックアップ] を選択します。
2. 更新するスナップショットを選択します。
3. [Actions] (アクション)、[Set manual snapshot settings] (手動スナップショット設定) の順に選択します。
4. 保持期間を選択します。[カスタム値] を選択した場合は、日数を選択します。
5. [Save changes] (変更の保存) をクリックします。

スナップショットの削除

スナップショットを削除するには、次の手順を実行します。

スナップショットを削除するには

> **Note**
>
> 別のアカウントと共有しているスナップショットは削除できません。スナップショットを削除する前に、まず、そのアカウントのスナップショットへのアクセス権を削除する必要があります。

1. Amazon Redshift Serverless コンソールで、[データのバックアップ] を選択します。
2. 削除するスナップショットを選択します。
3. [Actions] (アクション) で、[Delete] (削除) を選択します。
4. [削除] を選択します。

スナップショットの復元

> **Note**
>
> RG または RA3 でプロビジョニングされたクラスターと Amazon Redshift Serverless ワークグループでは、バックアップ用ではないテーブルはサポートされていません。RG または RA3 クラスターおよびサーバーレスワークグループでバックアップしないとマークされたテーブルは、スナップショットの作成中に常にバックアップされ、スナップショットから復元するときに復元される永続テーブルとして扱われます。バックアップ用ではないテーブルのスナップショットコストを回避するには、スナップショットを作成する前にそれらを切り捨てます。

サーバーレス名前空間にスナップショットを復元すると、現在のデータベースがスナップショット内のデータベースに置き換えられます。

サーバーレス名前空間へのスナップショットの復元は、2 つの段階で行われます。最初の段階は数分で完了します。この段階では、データを名前空間に復元し、クエリで使用できるようにします。2 番目の段階では、データベースのチューニングが行われ、小さなパフォーマンスの問題が発生する可能性があります。この 2 番目の段階は、数時間から数日、場合によっては数週間必要となる場合があります。この処理時間の長さはデータのサイズによって異なりますが、データベースのチューニングが進むと、パフォーマンスは徐々に向上します。この段階の終了時には、サーバーレス名前空間が完全にチューニングされ、クエリの送信時にパフォーマンスが発生しません。

スナップショットをサーバーレス名前空間に復元するには

1. Amazon Redshift Serverless コンソールで、[データのバックアップ] を選択します。
2. 復元するスナップショットを選択します。一度に復元できるのは 1 つのスナップショットのみです。
3. [Actions] (アクション)、[Restore to serverless namespace] (サーバーレス名前空間に復元する) の順に選択します。
4. 復元先に使用可能な名前空間を選択します。ステータスが [Available] (使用可能) の名前空間にのみ復元できます。
5. [復元] を選択します。

スナップショットをプロビジョニングされたクラスターに復元するには

1. Amazon Redshift Serverless コンソールで、[データのバックアップ] を選択します。
2. 復元するスナップショットを選択します。
3. [Actions] (アクション)、[Restore to provisioned cluster] (プロビジョニングされたクラスターに復元する) の順に選択します。
4. クラスター識別子を入力します。
5. [Node type] (ノードタイプ) を選択します。ノード数は、ノードタイプによって異なります。
6. コンソールページの指示に従って [Cluster configuration] (クラスター設定)のプロパティを入力します。詳細については、「クラスターの作成」を参照してください。

プロビジョン済みクラスター上のスナップショットの詳細については、「Amazon Redshift のスナップショットとバックアップ」を参照してください。

復旧ポイントの変換

Amazon Redshift Serverless の復旧ポイントは約 30 分ごとに作成され、24 時間保存されます。復旧ポイントをスナップショットに変換するには、以下の手順を実行します。

復旧ポイントをスナップショットに変換するには

1. Amazon Redshift Serverless コンソールで、[Data backup] (データのバックアップ) を選択します。
2. 復旧ポイントで、スナップショットに変化する復旧ポイントの [Creation time] (作成時刻) を選択します。
3. [Create snapshot from recovery point] (復旧ポイントからスナップショットを作成する) を選択します。
4. スナップショット ID を入力します。
5. [作成] を選択します。

復旧ポイントの復元

Amazon Redshift Serverless の復旧ポイントは約 30 分ごとに作成され、24 時間保存されます。復旧ポイントをサーバーレス名前空間に復元するには、次の手順の手順を実行します。

サーバーレス名前空間を復旧ポイントに復元するには

1. Amazon Redshift Serverless コンソールで、[Data backup] (データのバックアップ) を選択します。
2. 復旧ポイントで、復元する復旧ポイントの [Creation time] (作成時刻) を選択します。
3. [復元] を選択します。ステータスが [Available] (利用可能) の名前空間にのみ復元できます。
4. テキスト入力フィールドに restore (復元) と入力し、[Restore] (復元) を選択します。

別の AWS リージョン へのバックアップのコピー

スナップショットと復旧ポイントを別の AWS リージョン に自動的にコピーするように Amazon Redshift Serverless を設定できます。ソース AWS リージョン でスナップショットを作成すると、このスナップショットは、送信先リージョンにコピーされます。スナップショットと復旧ポイントを一度に単一の送信先 AWS リージョン にのみコピーするように名前空間を設定できます。Amazon Redshift Serverless が利用できる AWS リージョン の一覧については、「Amazon Web Services 全般のリファレンス」の「Redshift Serverless API」に記載されているエンドポイントを参照してください。

バックアップのコピーを設定する際は、コピーされたスナップショットを Amazon Redshift Serverless が保持する期間を指定することもできます。復旧ポイントの保持期間は変更できず、1 日である必要があります。送信先リージョンのスナップショットの保持期間は、ソースリージョンのスナップショットの保持期間とは異なります。この保持期間はデフォルトでは、スナップショットの無期限の保持です。[カスタム値] を選択した場合は日数を選択します。この値は、1～3653 日の範囲で選択する必要があります。

スナップショットのコピーの送信先リージョンを変更するには、まずバックアップのコピーを無効にしてから、コピーを再度有効にする際に新しい送信先リージョンを指定します。

スナップショットまたは復旧ポイントを送信先リージョンにコピーしたら、これを使用してデータを送信先リージョンで復元できます。

データはデフォルトで AWS マネージドキーを使用して暗号化されます。別のキーを使用するには、ソース AWS リージョン でバックアップコピーを設定する際に使用するキーを選択すると、Amazon Redshift Serverless が権限を自動的に作成し、送信先 AWS リージョン でスナップショット暗号化が有効になります。

バックアップを別のリージョンにコピーするには、次の IAM アクセス許可があることを確認します。

```
redshift-serverless:CreateSnapshotCopyConfiguration
redshift-serverless:UpdateSnapshotCopyConfiguration
redshift-serverless:ListSnapshotCopyConfigurations
redshift-serverless:DeleteSnapshotCopyConfiguration
```

独自の KMS キーを使用してバックアップを暗号化する場合は、次のアクセス許可も必要です。

```
kms:CreateGrant
kms:DescribeKey
```

スナップショットまたは復旧ポイントを別の AWS リージョン にコピーするように設定するには

1. Amazon Redshift Serverless コンソールで、スナップショットまたは復旧ポイントのコピーを設定する名前空間を選択します。
2. [アクション]、[クロスリージョンバックアップの設定] を選択します。
3. スナップショットをコピーする送信先 AWS リージョン を選択します。
4. (オプション) スナップショットの保持期間を選択します。[カスタム値]を選択する場合は日数を選択します。この値は、1～3653 日の間で選択する必要があります。デフォルトでは無期限に保持されます。
5. (オプション) 送信先リージョンでの暗号化に使用する別の AWS KMS キーを選択します。
6. [設定の保存] を選択します。

テーブルの復元

スナップショットまたは復旧ポイントから特定のテーブルを復元することもできます。この場合は、ソース スナップショットまたは復旧ポイント、データベース、スキーマ、テーブル、ターゲット データベース、スキーマ、新しいテーブル名を指定します。この新しいテーブルは、既存のテーブルと同じ名前は使用できません。テーブルを復元して既存のテーブルを置き換える場合は、テーブルを復元する前に、まずテーブル名を変更するか削除する必要があります。

> **Note**
>
> RG または RA3 でプロビジョニングされたクラスターと Amazon Redshift Serverless ワークグループでは、バックアップ用ではないテーブルはサポートされていません。RG または RA3 クラスターおよびサーバーレスワークグループでバックアップしないとマークされたテーブルは、スナップショットの作成中に常にバックアップされ、スナップショットから復

元するときに復元される永続テーブルとして扱われます。ただし、バックアップ用ではないテーブルの選択的な復元はサポートされていません。

ターゲットテーブルは、ソーステーブルの列の定義、テーブル属性、および外部キーを除く列の属性を使って作成されます。依存関係による競合を回避するため、ターゲットテーブルはソーステーブルから外部キーを継承しません。ソーステーブルで付与されたビューや許可などの依存関係は、ターゲットテーブルに適用されません。

ソーステーブルの所有者が存在するなら、そのユーザーは指定されたデータベースおよびスキーマの関係において所有者となるのに十分なアクセス許可がある場合にのみ、復元されたテーブルの所有者となります。それ以外の場合には、クラスターの起動時に作成した管理者ユーザーが、復元されたテーブルを所有します。

復元されたテーブルは、バックアップが作成された時の状態に戻されます。これには、Amazon Redshift の直列化分離への準拠により定義されるトランザクションの可視性のルールが含まれます。つまり、バックアップ後に開始した実行中のトランザクションにデータがすぐに見えるようになるということです。

Amazon Redshift Serverless コンソールを使用して、スナップショットからテーブルを復元できます。

データバックアップからのテーブルの復元には、次のとおりの制限があります。

- 一度に復元できるのは 1 つのテーブルのみです。
- ソーステーブルで付与されたビューや許可などの依存関係は、ターゲットテーブルに適用されません。
- 復元中のテーブルに対して行レベルのセキュリティが有効になっている場合、Amazon Redshift Serverless は、行レベルのセキュリティがオンになっているテーブルを復元します。

Amazon Redshift Serverless コンソールを使用してテーブルを復元するには

1. Amazon Redshift Serverless コンソールで、[データのバックアップ] を選択します。
2. 復元するテーブルを含むスナップショットまたは復旧ポイントを選択します。
3. [アクション]、[スナップショットからテーブルを復元する] または [Restore table from recovery point] を選択します。
4. ソースのスナップショットまたは復旧ポイントとターゲットテーブルに関する情報を入力して、[テーブルの復元] をクリックします。

Amazon Redshift Serverless でのデータ共有

データ共有を使用すると、データにライブアクセスできるため、Amazon Redshift Serverless で更新された一貫性のある最新情報をユーザーが確認できます。

同じもしくは異なる AWS アカウント に対し、異なる Amazon Redshift Serverless インスタンスを通じて、データを読み取り目的で共有します。

データ共有を開始するには、SQL インターフェイスまたは Amazon Redshift コンソールのいずれかを使用します。詳細については、「Amazon Redshift データベースデベロッパーガイド」の「Amazon Redshift でのデータの共有」を参照してください。

データ共有により、Amazon Redshift Serverless 名前空間とプロビジョニングされたクラスターは、1 つの AWS アカウント 内でも、複数の AWS アカウント や AWS リージョン にまたがっていても、ライブデータを互いに共有できます。詳細について、「データ共有が利用可能なリージョン」を参照してください。

AWS アカウント 内でのデータ共有を開始するには、AWS マネジメントコンソールを開き、[Amazon Redshift console] (Amazon Redshift コンソール) をクリックします。[Namespace configuration] (名前空間の設定)、[Datashares] (データ共有) の順にクリックします。

データ共有内のデータのクエリを開始するには、ワークグループが関連付けられた名前空間内にデータベースを作成します。指定したデータ共有から、ワークグループが関連付けられている名前空間を選択し、データをクエリするデータベースを作成します。

考慮事項

Amazon Redshift Serverless のデータ共有を使用する際には以下について考慮します。

- Amazon Redshift は、プロビジョニングされた RG クラスターと RA3 クラスター、およびサーバーレスエンドポイントワークグループで、プロデューサーまたはコンシューマーとしてのデータ共有をサポートしています。
- Amazon Redshift Serverless は、デフォルトで暗号化されます。

サポートされているデータベースオブジェクト、暗号化要件、ソートキー要件など、データ共有の制限事項のリストについては、「Amazon Redshift データベースデベロッパーガイド」の「Amazon Redshift でのデータ共有に関する考慮事項」を参照してください。

データ共有表示のアクセス権を付与する

スーパーユーザーは、スーパーユーザーではないユーザーにアクセス権を提供して、すべてのユーザーが作成したデータ共有を表示できるようにすることができます。

データ共有のアクセス権をユーザーに付与するには、以下のコマンドを使用して、データ共有のアクセス権を提供します。datashare_name はデータ共有の名前、user-name は アクセス権を提供するユーザーの名前です。

```
grant share on datashare datashare_name to "IAM:test_user";
```

データ共有のアクセス権をユーザーグループに付与するには、まずユーザーを含むユーザーグループを作成します。ユーザーグループの作成方法については、「CREATE GROUP」を参照してください。次に、以下のコマンドを使用してユーザーにデータ共有のアクセス権を付与します。ここで、datashare_name はデータ共有の名前、user-group はアクセス権を付与するユーザーグループの名前です。

```
grant share on datashare datashare_name to group user_group;
```

GRANT ステートメントの使用方法については、「GRANT」を参照してください。

AWS Glue Data Catalog への名前空間の登録

名前空間全体を AWS Glue Data Catalog に登録し、AWS Glue によって管理されるカタログを作成できます。これらのカタログには、Apache Iceberg REST API に対応した任意の SQL エンジンでアクセスできます。Amazon Redshift から Apache Iceberg 互換のカタログを作成する方法については、「Amazon Redshift データベースデベロッパーガイド」の「Apache Iceberg の Amazon Redshift との互換性」を参照してください。

サーバーレス名前空間を AWS Glue Data Catalog に登録するには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Redshift Serverless] を選択します。Serverless ダッシュボードが表示されます。[名前空間/ワークグループ] セクションには、現在の AWS リージョン にある、ご利用アカウントの名前空間とワークグループのリストが表示されます。名前空間がない場合は、[ワークグループを作成] を選択して、ワークグループとそれに対応する名前空間を作成します。

3. 登録する名前空間の名前を選択します。
4. [アクション] から [AWS Glue Data Catalog に登録] を選択します。[AWS Glue Data Catalog に登録] ポップアップボックスが表示されます。
5. 名前空間の登録先の AWS アカウント ID を [登録先アカウント ID] に入力します。このアカウント ID が、AWS Glue Data Catalog でカタログを保持します。
6. [名前空間の登録名] に名前を入力します。これは、Data Catalog で名前空間の名前になります。
7. [登録] を選択します。AWS Lake Formation コンソールに自動的に移動します。
8. AWS Lake Formation でカタログ作成のプロセスに従います。カタログ作成の詳細については、「AWS Lake Formation デベロッパーガイド」の「Amazon Redshift データ AWS Glue Data Catalog への取り込み」を参照してください。

Amazon Redshift Serverless のリソースへのタグ付け

AWS では、タグはキーと値のペアで構成されるユーザー定義のラベルです。Amazon Redshift Serverless では、リソースに関するメタデータを一目で確認できるタグ付けをサポートしています。

タグはリソースに必須ではありませんが、コンテキストの提供に役立ちます。リソースには、リソースに関連する情報を含むメタデータでタグ付けしたいと思うかもしれません。たとえば、テスト環境に属するリソースと本稼働環境に属するリソースを追跡するとします。環境という名前のキーを作成し、テスト用または本番稼働用の値を指定して、各環境で使用されるリソースを特定できます。他の AWS サービスでタグ付けを使用する場合や、ビジネスの標準カテゴリがある場合、一貫性を保つためにリソースに同じキーと値のペアを作成することをお勧めします。

リソースを削除した場合、関連付けられたタグが削除されます。AWS CLI と Amazon Redshift サーバーレスコンソールの両方を使用して、サーバーレスリソースにタグ付けできます。使用可能な API オペレーションは `TagResource`、`UntagResource`、`ListTagsForResource` です。

各リソースには、リソースに割り当てられた 1 つ以上のタグの集合であるタグセットが 1 つあります。リソースごとに、タグセットあたり最大 50 個のタグを含めることができます。タグは、リソースの作成時とリソースの作成後に追加できます。以下のサーバーレスリソースタイプにタグを追加できます。

- ワークグループ
- 名前空間
- スナップショット
- 復旧ポイント

タグには、次の要件があります。

- キーにプレフィックス aws: を付けることはできません。
- キーはタグセットごとに一意であることが必要です。
- キーに使用できる文字数は 1 ~ 128 文字です。
- 値に使用できる文字数は 0 ~ 256 文字です。
- 値は、タグセットごとに一意にする必要はありません。
- キーと値に使用できる文字は、Unicode 文字、数字、空白、および _ . : / = + - @ の記号です。
- キーと値は大文字と小文字が区別されます。

Amazon Redshift サーバーレスリソースのタグを管理するには

1. Amazon Redshift サーバーレスコンソールで、[Manage Tags] (タグの管理) を選択します。
2. 検索するリソースタイプを入力し、[Search resources] (リソースを検索) を選択します。タグを管理するリソースを選択し、[Manage tags] (タグの管理) を選択します。
3. リソースに追加するキーとオプションの値を指定します。タグを変更する場合、タグの値を変更することはできますが、キーは変更できません。
4. タグの追加、削除、変更が完了したら、[Save changes] (変更の保存) を選択し、[Apply] (適用) を選択して変更を保存します。

Amazon Redshift でプロビジョニングされたクラスター

Amazon Redshift データウェアハウスは、ノードと呼ばれるコンピューティングリソースの集合で、クラスターと呼ばれるグループに編成されています。各クラスターは Amazon Redshift エンジンを実行し、1 つ以上のデータベースを含みます。

> **Note**
>
> 現時点では、Amazon Redshift バージョン 1.0 エンジンを利用できます。ただし、エンジンの更新に伴い、複数の Amazon Redshift エンジンバージョンを選択できるようになる可能性があります。

Amazon Redshift のクラスターとノード

Amazon Redshift クラスターは、ノードで構成されています。クラスターごとに、リーダーノードと 1 つまたは複数のコンピューティングノードがあります。リーダーノードは、クライアントアプリケーションからクエリを受け取ってクエリを解析し、クエリ実行プランを作成します。次に、これらのプランの並列実行をコンピューティングノードと調整し、コンピューティングノードからの中間結果を集計します。最終的にクライアントアプリケーションに結果を返します。

コンピューティングノードは、クエリ実行プランを実行し、これらのクエリを処理するためにコンピューティングノード間でのデータ送信を行います。中間結果は、リーダーノードに送られて集計されてから、クライアントアプリケーションに送り返されます。リーダーノードとコンピューティングノードの詳細については、Amazon Redshift データベースデベロッパーガイドのデータウェアハウスシステムのアーキテクチャを参照してください。

> **Note**
>
> Amazon Redshift コンソールでクラスターを作成する場合 (https://console.aws.amazon.com/redshiftv2/) は、データのサイズとクエリの特性に基づいてクラスター設定に関する推奨事項を取得できます。このサイズ計算ツールを使用するには、RG または RA3 ノードタイプがサポートされている AWS リージョンのコンソールで [ヘルプ選択] を見つけてください。詳細については、「クラスターの作成」を参照してください。

クラスターを起動するときに指定するオプションの 1 つに、ノードタイプがあります。ノードタイプによって、各ノードの CPU、RAM、ストレージ容量、およびストレージデバイスでのタイプが決まります。

Amazon Redshift ではお客様のワークロードに対処するため、様々なノードタイプをご用意しています。必要なパフォーマンス、データサイズ、その増大によって、RG または RA3 をお選びいただくよう推奨します。

マネージドストレージが付属する RG または RA3 ノードでは、コンピューティング性能とマネージドストレージのスケーリングと支払いを独立させることで、データウェアハウスを最適化できます。RG または RA3 では、パフォーマンス要件に基づいてノードの数を選択し、使用したマネージドストレージに対してのみ料金が発生します。日々処理するデータ量を基に、RG または RA3 クラスターのサイズを選択してください。RG または RA3 ノードタイプを使用するクラスターは、仮想プライベートクラウド (VPC) で起動します。詳細については、「VPC での Redshift でプロビジョンされたクラスターまたは Amazon Redshift Serverless ワークグループの作成」を参照してください。

Amazon Redshift のマネージドストレージでは、各 RG または RA3 ノードに大容量の高性能 SSD を使用し、ローカルストレージの高速化が行われています。また、Amazon S3 によって長期間の耐久性があるストレージが提供されています。1 つのノード内のデータが増加して大容量ローカル SSD のサイズを超えた場合、そのデータは Amazon Redshift のマネージドストレージにより自動的に Amazon S3 にオフロードされます。Amazon Redshift のマネージドストレージに対して支払うのは、データが高性能 SSD 内にあるか Amazon S3 内にあるかにかかわらず、同じ低額な料金です。ますます増大するストレージを必要とするワークロードの場合、マネージドストレージを使用することで、コンピューティングノードとは別に、データウェアハウスのストレージ容量を自動的にスケールできます。

RG ノードは、RA3 ノードタイプに対して次の機能を提供します。

- Graviton ベースのインスタンスタイプ – 分析ワークロードに優れた料金パフォーマンスを提供します。
- 統合データレイククエリエンジン – RG クラスターには、クラスター独自のコンピューティングリソースで直接実行される統合データレイククエリエンジンが含まれています。RA3 クラスターは、データレイククエリに Amazon Redshift Spectrum を使用します。

DC2 ノードでは、ローカル SSD ストレージを使用してコンピューティング負荷の高いデータウェアハウスを持つことができます。必要なノード数はデータサイズとパフォーマンス要件に基づいて選択します。DC2 ノードは高いパフォーマンスを引き出すためローカルにデータを保存し、データのサ

イズが増えるに従って、さらに多くのコンピューティングノードを追加して、クラスターのストレージ容量を増強できます。圧縮で 1 TB 未満のデータセットでは、最も低い価格で最良のパフォーマンスを得るため、DC2 ノードタイプの利用を推奨します。データ量の増大が予想される場合は、RA3 ノードのご利用をお勧めします。このタイプのノードを使用することで、コンピューティング性能とストレージを別々にサイジングし、最高量の料金とパフォーマンスを活用できます。DC2 ノードタイプを使用するクラスターは、Virtual Private Cloud (VPC) で起動します。詳細については、「VPC での Redshift でプロビジョンされたクラスターまたは Amazon Redshift Serverless ワークグループの作成」を参照してください。

ノードタイプは、さまざまなサイズで使用できます。ノードサイズとノードの数によって、クラスターのストレージ総容量が決まります。詳細については、「ノードタイプの詳細」を参照してください。

ノードタイプに応じて、1 つのノード (単一ノード) または複数のノード (複数ノード) を使用できます。一部のノードタイプのクラスターのノードの 最小数は 2 ノードです。単一ノードクラスターでは、ノードは機能上リーダーとコンピューティングで共有されます。単一ノードクラスターは、本番稼働ワークロードの実行には推奨されません。マルチノードクラスターでは、リーダーノードとコンピューティングノードは分かれています。リーダーノードは、コンピューティングノードと同じノードタイプです。料金はコンピューティングノードに対してのみ課金されます。

Amazon Redshift では、各 AWS リージョンにおけるそれぞれの AWS アカウントのリソースにクォータが適用されます。クォータは、アカウントが AWS リージョン内の特定のリソースタイプ (ノードやスナップショットなど) に作成できるリソースの数を制限します。Amazon Redshift リソースに適用されるデフォルトクォータの詳細については、「Amazon Redshift でのクォータと制限」を参照してください。

クラスターのコストは、AWS リージョン、ノードタイプ、ノード数、ノードが事前に予約されているかどうかによって異なります。ノードのコストについては、Amazon Redshift の料金表ページを参照してください。

ノードタイプの詳細

以下のテーブルは、各ノードタイプとサイズのノード仕様をまとめたものです。表の見出しには、次の意味があります。

- vCPU は各ノードの仮想 CPU の数です。
- RAM は、各ノードのギビバイト (GiB) 単位のメモリ量です。
- ノードあたりのスライスの数は、従来のサイズ変更でクラスターを作成またはサイズ変更するときに、コンピューティングノードがパーティション分割されるデフォルトのスライス数です。

伸縮自在なリサイズを使用してクラスターのサイズを変更すると、ノードあたりのスライス数が変わる可能性があります。 ただし、クラスターのすべてのコンピューティングノードのスライスの総数は、伸縮自在なサイズ変更後も変わりません。

スナップショットからの復元操作を使用してクラスターを作成する場合、ノードタイプを変更すると、元のクラスターから生成されるクラスターのスライスの数が変わることがあります。

- ストレージは、各ノードのストレージの容量とタイプです。
- ノード範囲は、Amazon Redshift によりサポートされている、ノードタイプとサイズに対するノードの最小数および最大数です。

> **Note**
>
> 選択した AWS リージョン内の AWS アカウントに適用されたクォータによっては、さらに少ないノードに制限されることがあります。Amazon Redshift リソースに適用されるデフォルトクォータの詳細については、「Amazon Redshift でのクォータと制限」を参照してください。

- 総容量とは、ノード範囲で指定されているノードの最大数をデプロイした場合のクラスターのストレージ合計容量です。

次の表は、RG ノードの仕様を示しています。

ノードの種類	vCPU	RAM (GiB)	ノードごとのデフォルトスライス	ノードあたりのマネージドストレージの制限 [1]	クラスター作成によるノード範囲	マネージドストレージの総容量 [2]
rg.large (単一ノード)	2	16	2	1 TB	1	1 TB
rg.large (マルチノード)	2	16	2	8 TB	2 ~ 16	128 TB

ノードの種類	vCPU	RAM (GiB)	ノードごとのデフォルトスライス	ノードあたりのマネージドストレージの制限 [1]	クラスター作成によるノード範囲	マネージドストレージの総容量 [2]
rg.xlarge (マルチノード)	4	32	2	32 TB	2 ~ 163[3]	1024 TB[3]
rg.4xlarge	16	128	8	128 TB	2 ~ 324[4]	8192 TB[4]
rg.12xlarge	48	384	16	128 TB	2 ~ 128	16,384 TB

[1] Amazon Redshift マネージドストレージ用のストレージ制限を示します。これはハード制限です。

[2] マネージドストレージ制限の合計は、ノードの最大数にノードあたりのマネージドストレージの制限を掛けたものです。

[3] rg.xlarge (マルチノード) ノードタイプで、最大 16 個のノードを持つクラスターを作成できます。マルチノードクラスターでは、伸縮自在なサイズ変更により最大 32 個のノードに変更できます。

[4] rg.4xlarge (複数ノード) ノードタイプで、最大 32 個のノードを持つクラスターを作成できます。伸縮自在なサイズ変更で、最大 64 ノードまでノード変更できます。

次の表は、RA3 ノードの仕様を示しています。

ノードの種類	vCPU	RAM (GiB)	ノードごとのデフォルトスライス	ノードあたりのマネージドストレージの制限 [1]	クラスター作成によるノード範囲	マネージドストレージの総容量 [2]
ra3.large (単一ノード)	2	16	2	1 TB	1	1 TB[3]
ra3.large (マルチノード)	2	16	2	8 TB	2〜16	128 TB
ra3.xlplus (シングルノード)	4	32	2	4 TB	1	4 TB[3]
ra3.xlplus (マルチノード)	4	32	2	32 TB	2 ~ 164[4]	1024 TB[4]
ra3.4xlarge	12	96	4	128 TB	2 ~ 325[5]	8192 TB[5]
ra3.16xlarge	48	384	16	128 TB	2 ~ 128	16,384 TB

[1] Amazon Redshift マネージドストレージ用のストレージ制限を示します。これはハード制限です。

[2] マネージドストレージ制限の合計は、ノードの最大数にノードあたりのマネージドストレージの制限を掛けたものです。

[3] シングルノードクラスターのサイズをマルチノードに変更するには、従来のサイズ変更のみがサポートされます。

[4] ra3.xlplus (マルチノード) ノードタイプで、最大 16 個のノードを持つクラスターを作成できます。マルチノードクラスターでは、伸縮自在なサイズ変更により最大 32 個のノードに変更できます。

[5] ra3.4xlarge ノードタイプで、最大 32 個のノードを持つクラスターを作成できます。伸縮自在なサイズ変更で、最大 64 ノードまでノード変更できます。

次の表は、高密度コンピューティングノードの仕様を示しています。

ノードの種類	vCPU	RAM (GiB)	ノードごとのデフォルトスライス	1 ノードあたりのストレージ	ノード範囲	合計容量
dc2.large	2	15	2	160 GB NVMe-SSD	1 ~ 32	5.12 TB
dc2.8xlarge	32	244	16	2.56 TB NVMe-SSD	2 ~ 128	326 TB

> **Note**
>
> 高密度ストレージ (DS2) ノードタイプは利用できなくなりました。

ノードタイプの以前の名前

Amazon Redshift の以前のリリースでは、特定のノードタイプは異なる名前でした。Amazon Redshift API および AWS CLI では、以前の名前を使用できます。ただし、古い名前を参照するスクリプトは更新し、現在の名前を使用することをお勧めします。現在の名前と以前の名前は、次のとおりです。

現在の名前	以前の名前
ds2.xlarge	ds1.xlarge、dw.hs1.xlarge、dw1.xlarge
ds2.8xlarge	ds1.8xlarge、dw.hs1.8xlarge、dw1.8xlarge
dc1.large	dw2.large
dc1.8xlarge	dw2.8xlarge

ノードの数の決定

Amazon Redshift はクラスターのすべてのコンピューティングノードにクエリを分散して並列的に実行するので、クラスターにノードを追加することでクエリパフォーマンスを向上させることができます。2 つ以上のコンピューティングノードを持つクラスターを実行すると、データ損失が発生するリスクを低減するため、各ノードのデータが別のノードのディスクにミラーリングされます。

Amazon Redshift コンソールと Amazon CloudWatch メトリクスを使用して、クエリのパフォーマンスをモニタリングできます。クラスターの価格とパフォーマンスのバランスを維持するために、必要に応じてノードを追加または削除することもできます。追加のノードをリクエストしたときは、Amazon Redshift でデプロイメント、ロードバランシング、およびデータメンテナンスの詳細がすべて管理されます。クラスターパフォーマンスの詳細については、「Amazon Redshift クラスターパフォーマンスのモニタリング」を参照してください。

リザーブドノードは、一定量の本番稼働番ワークロードに最適で、オンデマンド料金と比べて大幅な割引を受けることができます。本番稼働設定を検証するために、実験と概念実証を実行した後で、リザーブドノードを購入できます。詳細については、「リザーブドノード」を参照してください。

クラスターを一時停止すると、クラスターが一時停止している間にオンデマンド課金が中断されます。この一時停止期間中は、バックアップストレージに対してのみお支払いいただきます。これにより、データウェアハウスの容量を事前にプランニングしたり購入したりする手間が省けます。また、開発環境またはテスト環境をコスト効率のよい方法で管理することができます。

オンデマンドノードとリザーブドノードの料金については、「Amazon Redshift 料金表」を参照してください。

EC2 を使用してクラスターを作成する

Amazon Redshift クラスターは、選択した Amazon Redshift ノードタイプとサイズに合わせて構成された Amazon EC2 インスタンスで実行されます。これらのネットワークプラットフォームの詳細については、「Amazon EC2 ユーザーガイド」の「サポートされているプラットフォーム」を参照してください。

> **Note**
>
> SQL クライアントツールと Amazon Redshift データベースとの接続問題を防止するには、以下の 2 つの方法のいずれかを実行することをお勧めします。パケットサイズの交渉をホストに許可する着信ルールを設定できます。または、Amazon EC2 インスタンスのネットワークインターフェイス (NIC) で最大転送単位 (MTU) を 1500 に設定して、TCP/IP ジャンボフレームを無効化できます。上記の方法の詳細については、「クエリがハングして、クラスターに達しない場合がある」を参照してください。

Amazon Virtual Private Cloud (Amazon VPC)

Amazon VPC を使用する場合、クラスターは AWS アカウントから論理的に分離された仮想プライベートクラウド (VPC) 内で稼働します。Amazon VPC でクラスターをプロビジョニングする場合は、クラスターに 1 つ以上の VPC のセキュリティグループを関連付けて、クラスターへのアクセスを制御します。詳細については、Amazon VPC ユーザーガイドの VPC のセキュリティグループを参照してください。

VPC でクラスターを作成するには、最初に VPC のサブネット情報を指定して Amazon Redshift クラスターサブネットグループを作成してから、クラスターの起動時にサブネットグループを指定する必要があります。詳細については、「Redshift リソースのサブネット」を参照してください。

Amazon Virtual Private Cloud (Amazon VPC) の詳細については、「Amazon VPC 製品の詳細ページ」を参照してください。

デフォルトのディスク容量アラーム

Amazon Redshift クラスターを作成するとき、クラスターのすべてのノードで使用されているディスク容量の平均比率を監視するように Amazon CloudWatch アラームを任意で設定できます。このアラームをデフォルトのディスク容量アラームと言います。

デフォルトのディスク容量アラームの目的は、クラスターのストレージ容量を監視することです。このアラームは、データウェアハウスのニーズを基に設定できます。たとえば、クラスターのサイズを変更する必要性の指標として警告を使用できます。別のノードタイプにするかノードを追加するため、または今後の拡張に備えてリザーブドノードを購入するためにサイズを変更できます。

ディスクの使用量が、指定した期間、特定の回数の指定した割合に達したとき、または超えたとき、デフォルトのディスク容量アラームがトリガーされます。デフォルトでは、指定した割合に達し、かつ 5 分以上その割合かそれ以上の状態が続いたとき、このアラームがトリガーされます。デフォルトの値は、クラスターを起動した後編集できます

CloudWatch アラームがトリガーされると、Amazon Simple Notification Service (Amazon SNS) により、指定された受取人に、割合がしきい値に達したことを警告する通知が送信されます。Amazon SNS は、トピックを使用して通知を送信する受取人およびメッセージを指定します。既存の Amazon SNS トピックを使用できますが、クラスターの起動時に指定した設定に基づいてトピックを作成することもできます。このアラームのトピックは、クラスターを起動した後編集できます。Amazon SNS トピック作成の詳細については、Amazon Simple Notification Service 入門ガイドを参照してください。

クラスターを起動した後、CloudWatch Alarms の下にあるクラスターの [Status] (ステータス) ウィンドウからアラームを表示および編集できます。名前は percentage-disk-space-used-default-<*string*> です。アラームを開いて、関連付けられている Amazon SNS トピックを表示し、アラームの設定を編集することができます。既存の Amazon SNS トピックを使用するように選択していない場合、作成されるトピックの名前は、<*clustername*>-default-alarms (<*recipient*>) です。たとえば、examplecluster-default-alarms (notify@example.com) のようになります。

デフォルトのディスク容量アラームを設定して編集する方法については、「クラスターの作成」および「ディスク容量アラームの作成」を参照してください。

> **Note**
>
> クラスターを削除すると、クラスターに関連付けられているアラームは削除されませんが、トリガーされることもありません。必要がなくなったら、CloudWatch コンソールからアラームを削除できます。

クラスターステータス

クラスター状態は、クラスターの現在のステータスを表示します。次の表では、各クラスターステータスについて説明します。

ステータス	説明
available	クラスターは実行されていて、利用可能です。
available, prep-for-resize	クラスターは伸縮自在なサイズ変更の準備をしています。クラスターは稼働しており、読み取りクエリや書き込みクエリに使用できますが、スナップショットの作成などのクラスターオペレーションは使用できません。
available, resize-cleanup	伸縮自在なサイズ変更オペレーションは、新しいクラスターのノードへのデータ転送を完了します。クラスターは稼働しており、読み取りクエリや書き込みクエリに使用できますが、スナップショットの作成などのクラスターオペレーションは使用できません。
cancelling-resize	サイズ変更オペレーションはキャンセルされています。
creating	Amazon Redshift がクラスターを作成しています。詳細については、「クラスターの作成」を参照してください。
deleting	Amazon Redshift がクラスターを削除しています。詳細については、「クラスターのシャットダウンと削除」を参照してください。
final-snapshot	Amazon Redshift は、クラスターを削除する前に最後のスナップショットを取得しています。詳細については、「クラスターのシャットダウンと削除」を参照してください。
hardware-failure	クラスターにハードウェア障害が生じました。 単一ノードクラスターがある場合、そのノードを置き換えることはできません。クラスターを復元するには、スナップショットを使います。詳細については、「Amazon Redshift スナップショットとバックアップ」を参照してください。
incompatible-hsm	Amazon Redshift は、ハードウェアセキュリティモジュール (HSM) に接続できません。クラスターと HSM 間の HSM 設定を確認してください。詳細については、「ハードウェアセキュリティモジュールを使用した暗号化」を参照してください。

ステータス	説明
incompatible-network	基本的なネットワーク設定に問題があります。クラスターを起動する VPC が存在すること、そして設定が正しいことを確認してください。詳細については、「VPC での Redshift リソース」を参照してください。
incompatible-parameters	関連付けられたパラメータグループの 1 つ以上のパラメータ値に問題があり、パラメータ値の適用ができません。パラメータグループを変更し、無効な値を更新してください。詳細については、「Amazon Redshift パラメータグループを作成します。」を参照してください。
incompatible-restore	スナップショットからクラスターを復元時に問題が発生しました。別のスナップショットでクラスターの復元を再度お試しください。詳細については、「Amazon Redshift スナップショットとバックアップ」を参照してください。
modifying	Amazon Redshift は、クラスターへの変更を加えています。詳細については、「クラスターの変更」を参照してください。
paused	クラスターが一時停止しています。詳細については、「クラスターの一時停止と再開」を参照してください。
rebooting	Amazon Redshift がクラスターを再起動しています。詳細については、「クラスターの再起動」を参照してください。
renaming	Amazon Redshift は、クラスターに新しい名前を適用しています。詳細については、「クラスターの名前変更」を参照してください。
resizing	Amazon Redshift がクラスターのサイズを変更しています。詳細については、「クラスターのサイズ変更」を参照してください。
rotating-keys	Amazon Redshift は、クラスターの暗号化キーを更新しています。詳細については、「暗号化キーのローテーション」を参照してください。
storage-full	クラスターがストレージ容量限に達しました。ノードを追加するためにクラスターのサイズ変更をするか、または別のノードサイズを選択してください。詳細については、「クラスターのサイズ変更」を参照してください。

ステータス	説明
updating-hsm	Amazon Redshift は HSM 設定を更新しています。

Amazon Redshift でプロビジョニングされたクラスターを使用する際の考慮事項

クラスターの作成後に、機能を使用できるリージョン、メンテナンスタスク、ノードタイプ、使用制限についての情報をこのセクションで確認できます。

リージョンとアベイラビリティーゾーンの考慮事項

Amazon Redshift は、複数の AWS リージョンで利用できます。デフォルトでは、Amazon Redshift は、選択した AWS リージョン内のアベイラビリティーゾーン (AZ) にクラスターをプロビジョニングします。アベイラビリティーゾーン (AZ) はランダムに選択されます。すべてのクラスターノードが同じアベイラビリティーゾーンにプロビジョニングされます。

特定のアベイラビリティーゾーンで Amazon Redshift が使用可能な場合は、オプションでそのゾーンをリクエストできます。たとえば、1 つの AZ で Amazon EC2 インスタンスが既に実行されている場合、同じアベイラビリティーゾーン内に Amazon Redshift クラスターを作成して、レイテンシーを低減させることができます。一方、可用性を高めるために別のアベイラビリティーゾーンを選択することもできます。Amazon Redshift は、AWS リージョン内のすべてのアベイラビリティーゾーンで利用できるとは限りません。

Amazon Redshift クラスターをプロビジョンできる、サポート対象の AWS リージョンの一覧については、Amazon Web Services 全般のリファレンス の「Amazon Redshift エンドポイント」を参照してください。

クラスターのメンテナンス

Amazon Redshift は定期的にメンテナンスを実行して、クラスターにアップグレードを適用します。更新中は Amazon Redshift クラスターで通常のオペレーションを実行することはできません。クラスターのメンテナンス方法を制御する方法がいくつかあります。たとえば、クラスターにアップデートをいつ展開するかを制御できます。また、クラスターが常に最新のリリースバージョンを実行するのか、以前にリリースされたバージョンから最新のリリースバージョンを実行するのかを選択することもできます。最後に、必須ではないメンテナンスアップデートをしばらく延期することもできます。

メンテナンスウィンドウ

AWS リージョンごとに決められた 8 時間のうちのランダムな 30 分間、Amazon Redshift によりメンテナンスウィンドウが割り当てられます。これは、1 週間 (月曜日から日曜日まで) のうちのランダムな日に起こります。

デフォルトの メンテナンスウィンドウ

次のリストでは、デフォルトでメンテナンスウィンドウが割り当てられる各 AWS リージョンの時間を示します。

- 米国東部 (バージニア北部) リージョン: 03:00 ~ 11:00 UTC
- 米国東部 (オハイオ) リージョン: 03:00 ~ 11:00 UTC
- 米国西部 (北カリフォルニア) リージョン: 06:00 ~ 14:00 UTC
- 米国西部 (オレゴン) リージョン: 06:00 ~ 14:00 UTC
- アフリカ (ケープタウン) リージョン: 20:00 ~ 04:00 UTC
- アジアパシフィック (香港) リージョン: 13:00 ~ 21:00 UTC
- アジアパシフィック (ハイデラバード) リージョン: 16:30 ~ 00:30 UTC
- アジアパシフィック (ジャカルタ) リージョン: 15:00 ~ 23:00 UTC
- アジアパシフィック (マレーシア) リージョン: 14:00 ~ 22:00 UTC
- アジアパシフィック (メルボルン) リージョン: 12:00 ~ 20:00 UTC
- アジアパシフィック (ムンバイ) リージョン: 16:30 ~ 00:30 UTC
- アジアパシフィック (ニュージーランド) リージョン: 10:00 ~ 18:00 UTC
- アジアパシフィック (大阪) リージョン: 13:00 ~ 21:00 UTC
- アジアパシフィック (ソウル) リージョン: 13:00 ~ 21:00 UTC
- アジアパシフィック (シンガポール) リージョン: 14:00 ~ 22:00 UTC
- アジアパシフィック (シドニー) リージョン: 12:00 ~ 20:00 UTC
- アジアパシフィック (台北) リージョン: 14:00 ~ 22:00 UTC
- アジアパシフィック (タイ) リージョン: 15:00 ~ 23:00 UTC
- アジアパシフィック (東京) リージョン: 13:00 ~ 21:00 UTC
- カナダ (中部) リージョン: 03:00 ~ 11:00 UTC
- カナダ西部 (カルガリー) リージョン: 04:00 ~ 12:00 UTC

- 中国 (北京) リージョン: 13:00 ~ 21:00 UTC
- 中国 (寧夏) リージョン: 13:00 ~ 21:00 UTC
- 欧州 (フランクフルト) リージョン: 06:00 ~ 14:00 UTC
- 欧州 (アイルランド) リージョン: 22:00 ~ 06:00 UTC
- 欧州 (ロンドン) リージョン: 22:00 ~ 06:00 UTC
- 欧州 (ミラノ) リージョン: 21:00 ~ 05:00 UTC
- 欧州 (パリ) リージョン: 23:00 ~ 07:00 UTC
- 欧州 (ストックホルム) リージョン: 23:00 ~ 07:00 UTC
- 欧州 (チューリッヒ) リージョン: 20:00 ~ 04:00 UTC
- イスラエル (テルアビブ) リージョン:20:00 ~ 04:00 UTC
- メキシコ (中部) リージョン: 04:00 ~ 12:00 UTC
- 欧州 (スペイン) リージョン: 21:00 ~ 05:00 UTC
- 中東 (バーレーン) リージョン: 13:00 ~ 21:00 UTC
- 中東 (アラブ首長国連邦) リージョン: 18:00–02:00 UTC
- 南米 (サンパウロ) リージョン: 19:00 ~ 03:00 UTC

メンテナンスイベントが特定の週に予定されている場合、割り当てられた 30 分のメンテナンスウィンドウ中に開始されます。メンテナンスの実行中、Amazon Redshift で実行中のクエリまたはその他のオペレーションは終了します。計画的なメンテナンス中に発生したダウンタイムは、Amazon Redshift サービスレベルアグリーメントでは、毎月のダウンタイムや利用不能とは見なされません。ほとんどのメンテナンスは 30 分のメンテナンスウィンドウ中に完了しますが、メンテナンスタスクの一部はウィンドウが終了した後も実行を続ける場合があります。スケジュールされたメンテナンスウィンドウの間に実行されるメンテナンスタスクがない場合、クラスターは次にスケジュールされたメンテナンスウィンドウまで通常どおり稼働します。

クラスターをプログラムで、または Amazon Redshift コンソールを使用して変更すると、スケジュールされたメンテナンスウィンドウを変更できます。[メンテナンス] タブではメンテナンスウィンドウが表示され、メンテナンス期間を確認したり、クラスターのメンテナンス実施日時を設定したりできます。

クラスターはメンテナンスウィンドウ外で再起動することがあります。これが発生する理由はいくつかあります。もう 1 つの一般的な理由は、クラスターに問題が検出され、正常な状態に戻すためのメンテナンスオペレーションが行われていることです。詳細については、「Amazon Redshift クラス

ターがメンテナンスウィンドウ外で再起動したのはなぜですか?」という記事を参照してください。この問題が発生する理由に関する詳細情報が記載されています。

メンテナンスの遅延

クラスターのメンテナンスウィンドウを変更する場合は、メンテナンスを最長 60 日まで延期できます。例えば、クラスターのメンテナンスウィンドウが水曜日の 8:30 ~ 9:00 (UTC) に設定されていて、その時間にクラスターにアクセスする必要がある場合、メンテナンスを延期できます。

メンテナンスを延期しても、Amazon Redshift は引き続きハードウェアの更新やその他の必須のセキュリティ更新をクラスターに適用します。更新中は、クラスターを使用できません。

次回のメンテナンスウィンドウ中にハードウェアの更新またはその他の必須のセキュリティ更新が予定されている場合、Amazon Redshift は [保留中] カテゴリで事前に通知を送信します。保留中のイベント通知の詳細については、「Amazon Redshift プロビジョニングされたクラスターのイベント通知」を参照してください。

Amazon Simple Notification Service (Amazon SNS) からイベント通知を受け取ることもできます。Amazon SNS イベント通知のサブスクライブの詳細については、「Amazon Redshift クラスターイベント通知のサブスクリプション」を参照してください。

クラスターのメンテナンスを延期すると、延期した後の次回のメンテナンスウィンドウは延期できません。

> **Note**
>
> メンテナンスの開始後に延期することはできません。

クラスターのメンテナンスの詳細については、以下のドキュメントを参照してください。

- メンテナンスウィンドウ
- クラスターオペレーション
- クラスターの変更

クラスターメンテナンストラックの選択

Amazon Redshift が新しいクラスターバージョンをリリースすると、メンテナンスウィンドウ中にクラスターが更新されます。クラスターを最新のリリースに更新するか、前のリリースに更新するか制御できます。

トラックは、メンテナンスウィンドウ中にどのクラスターバージョンを適用するかを制御します。Amazon Redshift が新しいクラスターバージョンをリリースすると、そのバージョンは最新のトラックに割り当てられ、以前のバージョンは前のトラックに割り当てられます。

クラスターのトラックの詳細については、「Amazon Redshift でプロビジョニングされたクラスターとサーバーレスワークグループのトラック」を参照してください。

RG および RA3 ノードがコンピューティングとストレージを分離する方法を理解する

これらのセクションでは、RG および RA3 ノードタイプで使用できるタスクについて詳しく説明し、一連のユースケースへの適用可能性を示し、以前に利用可能だったノードタイプと比較した場合の利点について詳しく説明します。

RG および RA3 ノードの利点と可用性

RG および RA3 ノードには、次のような利点があります。

- ストレージコストを増加させることなく、コンピューティング能力を柔軟に拡張できます。また、コンピューティング能力を過剰にプロビジョニングすることなく、ストレージを拡張できます。
- ホットデータには高性能の SSD を使用し、コールドデータには Amazon S3 を使用します。したがって、使いやすさ、コスト効率の高いストレージ、高いクエリパフォーマンスを提供します。
- また、AWS Nitro System 上に構築された高帯域幅ネットワークを使用して、Amazon S3 へのデータのオフロードと取得にかかる時間をさらに短縮できます。

次の場合は、RG および RA3 ノードタイプを選択することを検討してください。

- コンピューティングをストレージから別にスケーリングできる柔軟性が必要な場合。
- 合計データの一部を照会します。
- データ量は急速に増加しているか、急速に増加すると予想されます。
- パフォーマンスのニーズのみに基づいてクラスターをサイズ変更できる柔軟性が必要です。

データレイククエリの計算に関する考慮事項 – DC2 および RA3 でプロビジョニングされたクラスターでは、データレイククエリは Redshift Spectrum で実行されます。Redshift Spectrum は、クラスターから独立した専用の Amazon Redshift サーバーを使用します。RG でプロビジョニングされたクラスターでは、データレイククエリはクラスター独自のコンピューティングリソースで実行され、それらのリソースを他のワークロードと共有します。大量のデータレイククエリを使用するワークロードの場合は、RG クラスターのサイズを設定するときにこの点を考慮してください。

RG ノードタイプを使用するには、AWS リージョンが RG をサポートしている必要があります。詳細については、「AWS リージョンでの RG ノードタイプの可用性」を参照してください。

RA3 ノードタイプを使用するには、AWS リージョンが RA3 をサポートしている必要があります。詳細については、「AWS リージョンでの RA3 ノードタイプの可用性」を参照してください。

> **Important**
>
> ra3.xlplus ノードタイプは、クラスターバージョン 1.0.21262 以降でのみ使用できます。Amazon Redshift コンソールを使用して、既存のクラスターのバージョンを表示できます。詳細については、「ワークグループバージョンまたはクラスターのバージョンの確認」を参照してください。
>
> RG または RA3 ノードタイプを使用するときは、必ず新しい Amazon Redshift コンソールを使用してください。
>
> また、トラックを使用する Amazon Redshift オペレーションで RG または RA3 ノードタイプを使用するには、メンテナンストラック値を、RG または RA3 をサポートするクラスターバージョンに設定する必要があります。トラックの詳細については、「クラスターメンテナンストラックの選択」を参照してください。

シングルノードの RA3 ノードタイプを使用する場合は、以下の点を考慮してください。

- データ共有のプロデューサーとコンシューマーがサポートされます。
- ノードタイプを変更する際には、従来のサイズ変更のみがサポートされます。伸縮自在なサイズ変更、またはスナップショットによる復元では、ノードタイプの変更はサポートされません。以下のシナリオがサポートされています。
  - 従来のサイズ変更による、1 ノードの dc2.xlarge から 1 ノードの ra3.xlplus への、およびその逆方向での変更。
  - 従来のサイズ変更による、1 ノードの dc2.xlarge からマルチノードの ra3.xlplus への、およびその逆方向での変更。
  - 従来のサイズ変更による、マルチノードの dc2.xlarge から 1 ノードの ra3.xlplus への、およびその逆方向での変更。

Amazon Redshift マネージドストレージの操作

Amazon Redshift マネージドストレージを使用すると、すべてのデータを Amazon Redshift に保存して処理できると同時に、柔軟性が高まってコンピューティング容量とストレージ容量を個別にス

ケーリングできます。データの取り込みは、引き続き COPY コマンドまたは INSERT コマンドを使用して行います。Amazon Redshift パフォーマンスを最適化し、ストレージ階層間で自動データ配置を管理するために、Amazon Redshift は、データブロックの温度、データブロックの有効期間、ワークロードパターンなどの最適化を利用します。手動操作を行わなくても、Amazon Redshift は必要に応じてストレージを自動的に Amazon S3 に拡張します。

ストレージコストの詳細については、「Amazon Redshift 料金表」を参照してください。

RG または RA3 ノードタイプの管理

コンピューティングとストレージを分離するには、RG または RA3 ノードタイプでクラスターを作成またはアップグレードします。RG または RA3 ノードタイプを使用するには、仮想プライベートクラウド (EC2-VPC) にクラスターを作成します。

RG または RA3 ノードタイプの Amazon Redshift クラスターのノード数を変更するには、次のいずれかの操作を行います。

- 伸縮自在なサイズ変更オペレーションでノードを追加または削除します。状況によっては、RG または RA3 クラスターからノードを削除することは、伸縮自在なサイズ変更では許可されません。たとえば、2:1 ノード数のアップグレードで、ノードあたりのスライス数が 32 になるとします。詳細については、「クラスターのサイズ変更」を参照してください。 伸縮自在なサイズ変更が利用できない場合は、従来のサイズ変更を使用してください。
- 従来のサイズ変更オペレーションでノードを追加または削除します。伸縮自在のサイズ変更では利用できない設定にサイズ変更する場合は、このオプションを選択します。伸縮性のあるサイズ変更は、従来のサイズ変更よりも高速です。詳細については、「クラスターのサイズ変更」を参照してください。

AWS リージョンでの RG ノードタイプの可用性

RG ノードタイプは、次の AWS リージョンでのみ使用できます。

- 米国東部 (バージニア北部) リージョン (us-east-1)
- 米国東部 (オハイオ) リージョン (us-east-2)
- 米国西部 (北カリフォルニア) リージョン (us-west-1)
- 米国西部 (オレゴン) リージョン (us-west-2)
- アフリカ (ケープタウン) リージョン (af-south-1)
- アジアパシフィック (香港) リージョン (ap-east-1)

- アジアパシフィック (台北) リージョン (ap-east-2)
- アジアパシフィック (東京) リージョン (ap-northeast-1)
- アジアパシフィック (ソウル) リージョン (ap-northeast-2)
- アジアパシフィック (大阪) リージョン (ap-northeast-3)
- アジアパシフィック (ムンバイ) リージョン (ap-south-1)
- アジアパシフィック (ハイデラバード) リージョン (ap-south-2)
- アジアパシフィック (シンガポール) リージョン (ap-southeast-1)
- アジアパシフィック (シドニー) リージョン (ap-southeast-2)
- アジアパシフィック (ジャカルタ) リージョン (ap-southeast-3)
- アジアパシフィック (メルボルン) リージョン (ap-southeast-4)
- アジアパシフィック (マレーシア) リージョン (ap-southeast-5)
- アジアパシフィック (タイ) リージョン (ap-southeast-7)
- カナダ (中部) リージョン (ca-central-1)
- 欧州 (フランクフルト) リージョン (eu-central-1)
- 欧州 (チューリッヒ) リージョン (eu-central-2)
- 欧州 (ストックホルム) リージョン (eu-north-1)
- 欧州 (ミラノ) リージョン (eu-south-1) (rg.xlarge および rg.4xlarge のみ)
- 欧州 (スペイン) リージョン (eu-south-2)
- 欧州 (アイルランド) リージョン (eu-west-1)
- 欧州 (ロンドン) リージョン (eu-west-2)
- 欧州 (パリ) リージョン (eu-west-3)
- メキシコ (中部) リージョン (mx-central-1)
- 南米 (サンパウロ) リージョン (sa-east-1)
- AWS GovCloud (米国東部) (us-gov-east-1)
- AWS GovCloud (米国西部) (us-gov-west-1)

AWS リージョンでの RA3 ノードタイプの可用性

RA3 ノードタイプは、次の AWS リージョンでのみ使用できます。

- 米国東部 (バージニア北部) リージョン (us-east-1)

- 米国東部 (オハイオ) リージョン (us-east-2)
- 米国西部 (北カリフォルニア) リージョン (us-west-1)
- 米国西部 (オレゴン) リージョン (us-west-2)
- アフリカ (ケープタウン) リージョン (af-south-1)
- アジアパシフィック (香港) リージョン (ap-east-1)
- アジアパシフィック (ハイデラバード) リージョン (ap-south-2)
- アジアパシフィック (ジャカルタ) リージョン (ap-southeast-3)
- アジアパシフィック (マレーシア) リージョン (ap-southeast-5)
- アジアパシフィック (メルボルン) リージョン (ap-southeast-4)
- アジアパシフィック (ムンバイ) リージョン (ap-south-1)
- アジアパシフィック (ニュージーランド) リージョン (ap-southeast-6)
- アジアパシフィック (大阪) リージョン (ap-northeast-3)
- アジアパシフィック (ソウル) リージョン (ap-northeast-2)
- アジアパシフィック (シンガポール) リージョン (ap-southeast-1)
- アジアパシフィック (シドニー) リージョン (ap-southeast-2)
- アジアパシフィック (台北) リージョン (ap-east-2)
- アジアパシフィック (タイ) リージョン (ap-southeast-7)
- アジアパシフィック (東京) リージョン (ap-northeast-1)
- カナダ (中部) リージョン (ca-central-1)
- カナダ西部 (カルガリー) — リージョン (ca-west-1)
- 中国 (北京) リージョン (cn-north-1)
- 中国 (寧夏) リージョン (cn-northwest-1)
- 欧州 (フランクフルト) リージョン (eu-central-1)
- 欧州 (チューリッヒ) リージョン (eu-central-2)
- 欧州 (アイルランド) リージョン (eu-west-1)
- 欧州 (ロンドン) リージョン (eu-west-2)
- 欧州 (ミラノ) リージョン (eu-south-1)
- 欧州 (スペイン) リージョン (eu-south-2)
- 欧州 (パリ) リージョン (eu-west-3)

- 欧州 (ストックホルム) リージョン (eu-north-1)
- イスラエル (テルアビブ) リージョン (il-central-1)
- メキシコ (中部) リージョン (mx-central-1)
- 中東 (バーレーン) リージョン (me-south-1)
- 中東 (UAE) リージョン (me-central-1)
- 南米 (サンパウロ) リージョン (sa-east-1)
- AWS GovCloud (米国東部) (us-gov-east-1)
- AWS GovCloud (米国西部) (us-gov-west-1)

RG または RA3 ノードタイプへのアップグレード

既存のノードタイプを RG または RA3 にアップグレードするには、ノードタイプを変更するための次のオプションがあります。

- スナップショットから復元する — Amazon Redshift では、クラスターの最新のスナップショットを使用し、これを復元して新しい RG または RA3 クラスターを作成します。クラスターの作成が完了するとすぐに (通常は数分以内に)、RG または RA3 ノードは本番稼働用ワークロード全体を実行する準備が整います。コンピューティングはストレージから分離されているため、ネットワーク帯域幅が広いので、ホットデータは高速でローカルキャッシュに取り込まれます。最新の DC2 スナップショットから復元する場合、RG または RA3 は DC2 ワークロードのホットブロック情報を保持し、最もホットなブロックをローカルキャッシュに格納します。詳細については、「スナップショットからのクラスターの復元」を参照してください。

  アプリケーションとユーザーに対して同じエンドポイントを維持する場合は、新しい RG または RA3 クラスターの名前を元の DC2 クラスターと同じ名前に変更します。クラスターの名前を変更するには、Amazon Redshift コンソールまたは `ModifyCluster` API オペレーションでクラスターを変更します。詳細については、Amazon Redshift API リファレンスから「クラスターの名前変更」または「`ModifyCluster` API オペレーション」を参照してください。
- 伸縮自在なサイズ変更 – サイズ変更は、伸縮自在なサイズ変更を使用してクラスターのサイズを変更します。伸縮自在なサイズ変更を使用してノードタイプを変更すると、Amazon Redshift はスナップショットを自動的に作成し、新しいクラスターを作成し、古いクラスターを削除し、新しいクラスターの名前を変更します。伸縮自在なサイズ変更オペレーションはオンデマンドで実行できるほか、任意のタイミングに実行することをスケジューリングすることもできます。既存の DC2 ノードタイプクラスターは、伸縮自在なサイズ変更を使用して RG または RA3 にすばやくアップグレードできます。詳細については、「伸縮自在なサイズ変更」を参照してください。

> **Note**
>
> 既存の RA3 クラスターを rg.xlarge または rg.4xlarge に移行するには、ソースクラスターでパッチバージョン P201 以降を実行する必要があります。rg.large または rg.12xlarge に移行するには、ソースクラスターで P202 以降を実行する必要があります。単一ノードの rg.large クラスターを作成したり、これに移行したりするには、クラスターで P204 以降を実行している必要があります。DC2 クラスターは、パッチバージョンに関係なく RG に移行できます。

次の表に、RG ノードタイプにアップグレードする際の推奨事項を示します。

この表では、開始時のクラスターノードの種類とサイズを基に推奨事項を示しますが、推奨事項はワークロードのコンピューティング要件によって異なります。RG のパフォーマンス特性が異なるため、推奨されるマッピングは出発点です。重要なワークロードをベンチマークし、観測されたパフォーマンスに基づいてノードの数を増減します。RG インスタンスは RA3 と比較して vCPU あたりの料金が低いため、ノードを追加してパフォーマンスを最適化しても、全体的なコストが低下する可能性があります。ノードタイプの料金の詳細については、「Amazon Redshift 料金表」を参照してください。

要件をより正確に見積もるには、Test Drive を使用して潜在的な構成を実行する概念実証 (POC) の実施を検討してください。Redshift Serverless の代わりに POC データウェアハウス用のクラスターをプロビジョニングします。概念実証の実施に関する詳細については、「Amazon Redshift データベース開発者ガイド」の「Amazon Redshift の概念実証 (POC) を実施する」を参照してください。

既存のノードタイプ	既存のノード数	推奨される新しいノードタイプ	アップグレードアクション
ra3.16xlarge	2 ~ 128	rg.12xlarge	1 つの ra3.16xlarge[1] ノードごとに 1 つの rg.12xlarge ノードで始めます。
ra3.4xlarge	2 ~ 64	rg.4xlarge	4 つの ra3.4xlarge[1] ノードごとに 3 つの rg.4xlarge ノードで始めます。

既存のノードタイプ	既存のノード数	推奨される新しいノードタイプ	アップグレードアクション
ra3.xlplus	1	rg.large	単一ノードの ra3.xlplus クラスター[1、2] の場合、2 つの rg.large ノードで始めます1,2。
ra3.xlplus	2 ~ 32	rg.xlarge	1 つの ra3.xlplus[1] ノードごとに 1 つの rg.xlarge のノードで始めます。
ra3.large	1 ~ 16	rg.large	1 つの ra3.large[1、2] ノードごとに 1 つの rg.large ノードで始めます。
dc2.8xlarge	2 ~ 15	rg.4xlarge	2 つの dc2.8xlarge[1] ノードごとに 3 つの rg.4xlarge ノードで始めます。
dc2.8xlarge	16 ~ 128	rg.12xlarge	2 つの dc2.8xlarge[1] ノードごとに 1 つの rg.12xlarge ノードで始めます。
dc2.large	1 ~ 3	rg.large	1 つの dc2.large[1、2] ノードごとに 1 つの rg.large ノードで始めます。
dc2.large	4	rg.large	rg.large[1] の 3 つのノードから始めます。

既存のノードタイプ	既存のノード数	推奨される新しいノードタイプ	アップグレードアクション
dc2.large	5 ~ 15	rg.xlarge	8 つの dc2.large[1] ノードごとに 3 つの rg.xlarge ノードで始めます。
dc2.large	16 ~ 32	rg.4xlarge	10 の dc2.large[1] ノードごとに 1 つの rg.4xlarge ノードで始めます。

[1]ワークロードの要件に応じて、追加のノードが必要になる場合があります。必要なクエリパフォーマンスのコンピューティング要件に基づいて、ノードを追加または削除します。RG インスタンスは RA3 と比較して vCPU あたりの料金が低いため、ノードを追加しても、以前の設定と比較して全体的なコストが低下する可能性があります。詳細については、「Amazon Redshift の料金」を参照してください。

[2]単一ノードクラスターについては、「RG ノードにアップグレードする際の考慮事項」の「単一ノードクラスター」を参照してください。

次の表に、RA3 ノードタイプにアップグレードする際の推奨事項を示します。

この表では、開始時のクラスターノードの種類とサイズを基に推奨事項を示しますが、推奨事項はワークロードのコンピューティング要件によって異なります。要件をより正確に見積もるには、Test Drive を使用して潜在的な構成を実行する概念実証 (POC) の実施を検討してください。Redshift Serverless の代わりに POC データウェアハウス用のクラスターをプロビジョニングします。概念実証の実施に関する詳細については、「Amazon Redshift データベース開発者ガイド」の「Amazon Redshift の概念実証 (POC) を実施する」を参照してください。

既存のノードタイプ	既存のノード数	推奨される新しいノードタイプ	アップグレードアクション
dc2.8xlarge	2 ~ 15	ra3.4xlarge	1 つの dc2.8xlarge[1] ノードごとに 2 つの

既存のノードタイプ	既存のノード数	推奨される新しいノードタイプ	アップグレードアクション
			ra3.4xlarge ノードで始めます。
dc2.8xlarge	16 ~ 128	ra3.16xlarge	2 つの dc2.8xlarge[1] ノードごとに 1 つの ra3.16xlarge ノードで始めます。
dc2.large	1 ~ 4	ra3.large	1 つの dc2.large[1] ノードごとに 1 つの ra3.large ノードで始めます。 2 つの dc2.large[1] ノードごとに 2 つの ra3.large ノードで始めます。 3 つの dc2.large[1] ノードごとに 3 つの ra3.large ノードで始めます。 4 つの dc2.large[1] ノードごとに 3 つの ra3.large ノードで始めます。
dc2.large	5 ~ 15	ra3.xlplus	8 つの dc2.large[1] ノードごとに 3 つの ra3.xlplus のノードで始めます。

既存のノードタイプ	既存のノード数	推奨される新しいノードタイプ	アップグレードアクション
dc2.large	16 ~ 32	ra3.4xlarge	8 つの dc2.large[1、2] ノードごとに 1 つの ra3.4xlarge ノードで始めます。

[1]ワークロードの要件に応じて、追加のノードが必要になる場合があります。必要なクエリパフォーマンスのコンピューティング要件に基づいて、ノードを追加または削除します。

[2]dc2.large ノードタイプを使用するクラスターは、32 ノードに制限されます。

一部の RA3 ノードタイプでは、ノードの最小数は 2 個です。RA3 クラスターを作成するときは、この点を考慮してください。

RG または RA3 ノードがサポートするネットワーク機能

RG または RA3 ノードは、他のノードでは利用できないネットワーク機能のコレクションをサポートしています。このセクションでは、各機能の簡単な説明とその他のドキュメントへのリンクを示します。

- プロビジョニングされたクラスター VPC エンドポイント — RG または RA3 クラスターを作成または復元すると、Amazon Redshift は 5431 ~ 5455 または 8191 ~ 8215 の範囲のポートを使用します。クラスターがこれらの範囲のいずれかのポートに設定されると、Amazon Redshift はクラスターの VPC エンドポイントを AWS アカウントに自動的に作成し、プライベート IP アドレスをアタッチします。クラスターをパブリックアクセス可能に設定すると、Redshift は AWS アカウントに Elastic IP アドレスを作成し、その IP アドレスを VPC エンドポイントにアタッチします。詳細については、「Amazon Redshift クラスターまたは Amazon Redshift Serverless ワークグループのセキュリティグループ通信設定の構成」を参照してください。
- 単一サブネット RG または RA3 クラスター — 1 つのサブネットで RG または RA3 クラスターを作成することはできますが、ディザスタリカバリ機能は使用できません。サブネットに複数のアベイラビリティーゾーン (AZ) がない場合にクラスターの再配置を有効にすると、例外が発生します。
- マルチサブネット RG または RA3 クラスターおよびサブネットグループ — 仮想プライベートクラウド (VPC) にクラスターをプロビジョニングする際にサブネットグループを作成することで、複数のサブネットを持つ RG または RA3 クラスターを作成できます。クラスターサブネッ

トグループにより、VPC 内にサブネットセットを指定でき、Amazon Redshift はそれらの内の 1 つにクラスターを作成します。サブネットグループの作成後、以前に追加したサブネットを削除したり、さらにサブネットを追加したりできます。詳細については、「Amazon Redshift cluster subnet groups」を参照してください。

- クロスアカウントまたはクロス VPC エンドポイントアクセス — Redshift 管理の VPC エンドポイントを設定することで、プロビジョニングされたクラスターまたは Amazon Redshift Serverless ワークグループにアクセスできます。例えば、クラスターまたはワークグループを含む VPC とクライアントツールを実行する VPC 間のプライベート接続としてセットアップできます。これにより、パブリック IP アドレスを使用したり、インターネット経由でトラフィックをルーティングしたりすることなく、データウェアハウスにアクセスできます。詳細については、「Redshift が管理する VPC エンドポイントの操作」を参照してください。
- クラスターの再配置 — サービスが中断しても、データを失うことなく、クラスターを別のアベイラビリティーゾーン (AZ) に移動できます。この機能は、コンソールで有効にします。詳細については、「クラスターの再配置」を参照してください。
- カスタムドメイン名 - Amazon Redshift クラスター用のカスタムドメイン名 (カスタム URL とも呼ばれます) を作成できます。これは、SQL クライアント接続をクラスターエンドポイントにルーティングする、読みやすい DNS レコードです。詳細については、「クライアント接続のカスタムドメイン名」を参照してください。

RG ノードにアップグレードする際の考慮事項

パッチバージョン

RA3 から RG に移行するには、ソースクラスターが必須の最小パッチバージョンを満たしている必要があります。rg.xlarge と rg.4xlarge の場合は P201、マルチノードの rg.large と rg.12xlarge の場合は P202、シングルノードの rg.large の場合は P204 です。DC2 クラスターは、パッチバージョンに関係なく RG に移行できます。移行前に、本番トラフィックを移動する前にスナップショットを RG クラスターに復元してワークロードをテストします。詳細については、「Amazon Redshift パッチ 201」、「Amazon Redshift パッチ 202」、「Amazon Redshift パッチ 204」を参照してください。

単一ノードクラスター

- 単一ノード RG クラスターは rg.large でのみサポートされています。
- 単一ノード rg.large クラスターは、単一ノード ra3.xlplus クラスターの最大 4 TB と比較して、最大 1 TB のマネージドストレージをサポートします。データが 1 TB より大きい場合、またはそれを超えると予想される場合は、マルチノード rg.large クラスターを使用してください。

- 単一ノードクラスターは、開発ワークロードやテストワークロードを対象としています。本番ワークロードの場合は、ノードを 2 つ以上使用してください。詳細については、「Amazon Redshift のクラスターとノード」および「ノードタイプの詳細」を参照してください。

Spectrum ワークロード

Spectrum を頻繁に使用するワークロードを持つクラスターは、RG に移行した後、クエリのパフォーマンスが遅くなる可能性があります。RA3 や DC2 とは異なり、RG は個別の Spectrum フリートではなくクラスターコンピューティングノードで統合データレイククエリエンジンを直接実行します。Spectrum のパフォーマンスが低下した場合は、RG クラスター内のノード数を増やすか、より大きな RG ノードタイプに移行します。

RG クラスターは、別の AWS リージョンにある Amazon S3 バケット内のデータのクエリもサポートしているため、Spectrum を使用する RA3 クラスターと DC2 クラスターに適用される同一リージョンの制限がなくなります。クロスリージョンクエリでは、追加のデータ転送料金が発生します。詳細については、「データレイクのクエリ」を参照してください。

カーソルベースのワークロード

RG ノードは、RA3 および DC2 と比較してサポートされているカーソルサイズが異なります。カーソルサイズ制限については、「カーソルの制約」を参照してください。ワークロードがカーソルに大きく依存していて、カーソルサイズの制限が発生した場合は、より大きな RG ノードタイプに移行します。ノードの数を比例的に減らして、クラスターの合計サイズとコストを同じように維持できます。

同時実行スケーリングの適格性

クラスターが同時実行スケーリングを使用していて、32 個を超えるノードがあり、rg.12xlarge に移行する場合は、伸縮自在なサイズ変更またはスナップショット復元を使用して同時実行スケーリングのサポートを維持します。クラスターに 32 個を超えるノードがあり、rg.12xlarge への従来のサイズ変更を実行すると、クラスターは同時実行スケーリングのサポートを失います。32 個を超えるノードを持つ、新規に作成された rg.12xlarge クラスターは、同時実行スケーリングをサポートしていません。詳細については、「同時実行スケーリングの候補」を参照してください。

リザーブドノード

クラスターがリザーブドノードを使用している場合、ノードタイプを変更しても予約は自動的に変更されません。既存の予約は、期間が終了するまで元のノードタイプに関連付けられたままになります。リザーブドノードを使用するクラスターを移行する前に、「リザーブドインスタンスを使用するクラスターのノードタイプを変更する前に考慮すべき事項」を参照してください。

クラスターオペレーション

クラスターを作成したら、クラスターオペレーションを実行してパフォーマンスを最適化し、コストを制御し、高可用性を確保できます。クラスターオペレーションを使用すると、データウェアハウスのニーズの変化に応じて、クラスターのサイズ変更、一時停止、再開、または再作成を行うことができます。

一般的なユースケースには、ピークワークロードのコンピューティングキャパシティのスケーリング、非アクティブ期間中にクラスターを一時停止してコストを削減する、ディザスタリカバリのために異なる設定または異なるアベイラビリティーゾーンでクラスターを再作成するなどがあります。以下のセクションでは、Amazon Redshift 環境を効果的に管理するために、さまざまなクラスターオペレーションを実行する方法の詳細について説明します。

クラスターの作成

Amazon Redshift を使用すると、プロビジョニングされたクラスターを作成して新しいデータウェアハウスを起動できます。プロビジョニングされたクラスターは、ノードと呼ばれるコンピューティングリソースのコレクションであり、単一の超並列処理 (MPP) システムに編成されています。

クラスターを作成する前に、「Amazon Redshift でプロビジョニングされたクラスター」および「Amazon Redshift のクラスターとノード」を参照してください。

クラスターを作成するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [クラスター] を選択します。現在のAWSリージョンにあるアカウントのクラスターがリストされています。各クラスターのプロパティのサブセットが、リストの列に表示されます。
3. [クラスターを作成] を選択して、クラスターを作成します。
4. コンソールページの指示に従って [クラスター設定] のプロパティを入力します。

   以下のステップでは、RG または RA3 ノードタイプをサポートする AWS リージョンで実行されている、Amazon Redshift コンソールについて説明します。RG または RA3 ノードタイプをサポートする AWS リージョンのリストについては、「Amazon Redshift 管理ガイド」の「AWS リージョンでの RG ノードタイプの可用性」および「AWS リージョンでの RA3 ノードタイプの可用性」を参照してください。

クラスターのサイズがわからない場合は、[選択のヘルプ] を選んでください。これにより、データウェアハウスに保存する予定のデータのサイズとクエリの特性について質問するサイジング計算ツールが起動されます。クラスターの必須サイズ (ノードタイプとノード数) がわかっている場合は、[I'll choose (選択する)] を選んでください。次に、ノードの種類とノードの数量を選択して、概念実証のためにクラスターのサイズを設定します。

> **Note**
>
> 組織が適格であり、Amazon Redshift Serverless が利用できない AWS リージョン でクラスターが作成されている場合、Amazon Redshift 無料トライアルプログラムでクラスターを作成できる場合があります。[このクラスターを何に使用する予定ですか?] という質問に対して、[本番稼働用] または [無料トライアル] のいずれかを選択します。[無料トライアル] を選択したときには、ra3.large ノードタイプの設定を作成します。無料トライアルの選択に関する詳細については、「Amazon Redshift 無料トライアル」を参照してください。Amazon Redshift Serverless が利用可能な AWS リージョン の一覧については、Amazon Web Services 全般のリファレンスの「Redshift Serverless API」に記載されているエンドポイントを参照してください。

5. [データベース設定] セクションで、[管理者ユーザー名] の値を指定します。[管理者パスワード] では、以下のオプションの中から選択します。

   - [パスワードの生成] – Amazon Redshift によって生成されたパスワードを使用します。
   - [管理者パスワードを手動で追加する] – 独自のパスワードを使用します。
   - [AWS Secrets Manager での管理者認証情報の管理] – Amazon Redshift は管理者パスワードの生成と管理に AWS Secrets Manager を使用します。AWS Secrets Manager を使用してパスワードのシークレットの生成と管理を行うには料金がかかります。AWS Secrets Manager の料金の詳細については、「AWS Secrets Manager の料金」を参照してください。

6. (オプション) コンソールページの指示に従って [Cluster permission (クラスタのアクセス許可)] のプロパティを入力します。Amazon S3 からデータをロードするなど、クラスターが他の AWS のサービスにアクセスする必要がある場合は、クラスターのアクセス許可を付与します。

7. [クラスターを作成] を選択して、クラスターを作成します。クラスターの使用準備ができるまで、数分かかることがあります。

追加の設定

クラスターを作成する際、追加のプロパティを指定してカスタマイズを行うことができます。これらのプロパティに関する詳細は、次のリストを参照してください。

IP アドレスタイプ

クラスターの IP アドレスタイプを選択します。リソースを IPv4 アドレスプロトコルでのみ通信させるか、IPv4 と IPv6 の両方でリソースを通信させるデュアルスタックモードを選択できます。この機能は、AWS GovCloud (米国東部) および AWS GovCloud (米国西部) の各リージョンでのみ利用可能です。AWS リージョンの詳細については、「リージョンとアベイラビリティーゾーン」を参照してください。

仮想プライベートクラウド (VPC)

クラスターサブネットグループを持つ VPC を選択します。クラスターの作成後は、クラスターサブネットグループを変更することはできません。

パラメータグループ

クラスターに関連付けるパラメータグループを選択します。選択しない場合、デフォルトのパラメータグループが使用されます。

暗号化:

クラスターとスナップショット内のデータをすべて暗号化するかどうかを選択します。デフォルト設定の [なし] のままにしておくと、暗号化は有効になりません。暗号化を有効にする場合は、AWS Key Management Service (AWS KMS) またはハードウェアセキュリティモジュール (HSM) のどちらを使用するか選択して、関連する設定を指定します。Amazon Redshift の暗号化の詳細については、Amazon Redshift データベース暗号化 を参照してください。

- KMS

  暗号化を有効にして、AWS KMS を使用して暗号化キーを管理する場合は [AWS Key Management Service (AWS KMS) の使用] を選択します。また、使用するキーを選択します。デフォルトキー、現在のアカウントのキー、別のアカウントのキーを選択できます。

  > **Note**
  >
  > 別の AWS アカウントのキーを使用する場合、使用するキーの Amazon リソースネーム (ARN) を入力します。キーを使用するアクセス権限が必要です。AWS KMS での

キーアクセスの詳細については、AWS Key Management Service デベロッパーガイドの「Controlling access to your keys」を参照してください。

Amazon Redshift での AWS KMS 暗号化キーの使用についての詳細は、「AWS KMS を使用した暗号化」を参照してください。

- HSM

  暗号化を有効にし、ハードウェアセキュリティモジュール (HSM) を使用して暗号化キーを管理する場合は、[HSM] を選択します。

  [HSM] を選択した場合は、[HSM 接続] と [HSM クライアント証明書] から値を選択します。これらの値は、Amazon Redshift と HSM がクラスターキーを渡すことができる信頼された接続を確立するために必要です。HSM 接続とクライアント証明書は、クラスターを起動する前に Amazon Redshift でセットアップする必要があります。HSM 接続とクライアント証明書のセットアップの詳細については、「ハードウェアセキュリティモジュールを使用した暗号化」を参照してください。

メンテナンストラック

使用するクラスターバージョンが、現行、末尾、または場合によってはプレビュートラックのいずれかを選択できます。

モニタリング:

CloudWatch アラームを作成するかどうかを選択できます。

クロスリージョンスナップショットを設定する

クロスリージョンスナップショットを有効化するかどうかを選択できます。

自動スナップショットの保持期間

これらのスナップショットを保持する日数 (35 日以内) を選択できます。ノードタイプが DC2 の場合、自動スナップショットを作成しないようにするため、ゼロ (0) 日を選択できます。

手動スナップショット保持期間

これらのスナップショットを保持する日数または `Indefinitely` を選択できます。

自動最適化のための追加のコンピューティングリソース

使用量が多い期間でも、自動最適化を実行するための追加のコンピューティングリソースを割り当てるかどうかを選択できます。詳細については、「Amazon Redshift データベースデベロッ

パーガイド」の「データベースの自動最適化のための追加のコンピューティングリソースの割り当て」を参照してください。

ディスク容量アラームの作成

ディスク容量の使用状況をモニタリングし、ディスク容量がクラスターの指定されたしきい値を超えたときに通知されるアラームを設定できます。ディスク容量使用状況アラームを作成すると、ストレージ容量をプロアクティブに管理し、クエリの失敗やデータインジェストエラーなど、ディスク容量不足による問題を防ぐことができます。次の手順では、ディスク容量使用量アラームを作成するプロセスについて説明します。

クラスターのディスク容量使用アラームを作成するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [Alarms] (アラーム) を選択します。
3. [アクション] で [アラームの作成] を選択します。[アラームの作成] ページが表示されます。
4. ページに表示された手順に従います。
5. [アラームの作成] を選択します。

クラスターの表示

クラスターを表示すると、クラスターの設定、ステータス、パフォーマンスメトリクスをモニタリングおよび管理できます。クラスターの詳細を表示することで、リソース使用率、クエリ実行時間、システムの状態に関するインサイトを得ることができます。次の手順では、クラスターの情報にアクセスする方法を示します。

クラスターを表示するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [クラスター] を選択します。現在のAWSリージョンにあるアカウントのクラスターがリストされています。各クラスターのプロパティのサブセットが、リストの列に表示されます。クラスターがない場合、[クラスターを作成] を選択して作成します。
3. リストでクラスター名を選択して、クラスターの詳細を表示します。

クラスターの変更

クラスターを変更すると、以下のオプションに対する変更が直ちに適用されます。

- VPC セキュリティグループ
- パブリックアクセス可能
- 管理者ユーザーパスワード
- HSM 接続
- HSM Client Certificate
- メンテナンスの詳細
- スナップショット設定

以下のオプションに対する変更は、クラスターを再起動した後に限り、有効になります。

- クラスター識別子

  クラスター識別子を変更すると、Amazon Redshift はクラスターを自動的に再起動します。
- 拡張された VPC のルーティング

  拡張 VPC ルーティングを変更すると、Amazon Redshift はクラスターを自動的に再起動します。
- クラスターパラメータグループ
- IP アドレスタイプ

  この機能は、AWS GovCloud (米国東部) および AWS GovCloud (米国西部) の各リージョンでのみ利用可能です。AWS リージョンの詳細については、「リージョンとアベイラビリティーゾーン」を参照してください。

自動作成されたスナップショットの保持期間を短縮すると、新しい保持期間に含まれなくなった、既存の自動作成されたスナップショットは削除されます。詳細については、「Amazon Redshift スナップショットとバックアップ」を参照してください。

クラスタープロパティの詳細については、「追加の設定」を参照してください。

クラスターを変更するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。

2. ナビゲーションメニューで [クラスター] を選択します。
3. 変更するクラスターを選択します。
4. [Edit] を選択します。[Edit cluster] (クラスターの編集) ページが表示されます。
5. クラスターのプロパティを更新します。変更できるプロパティには、次のものがあります。

   - クラスター識別子
   - スナップショット保持期限
   - クラスターの再配置

   ネットワークとセキュリティ、メンテナンス、およびデータベース構成の設定を編集するために、コンソールには適切なクラスターの詳細タブへのリンクが付いています。
6. [変更を保存] をクリックします。

クラスターのサイズ変更

データウェアハウスの容量とパフォーマンスのニーズが変わるため、クラスターサイズを変更して、Amazon Redshift のコンピューティングとストレージオプションを最大限に活用することができます。

クラスターのサイズを変更する場合、クラスターの現在の設定と異なっているノード数またはノードタイプを指定します。クラスターのサイズ変更処理が実行中の間は、クラスターに対する書き込みクエリまたは読み取り/書き込みクエリは実行できません。読み込みクエリのみ実行できます。

さまざまな方法を使用してクラスターのサイズを変更するチュートリアルも含めて、クラスターのサイズ変更方法の詳細については、「クラスターのサイズ変更」を参照してください。

クラスターのサイズを変更するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [クラスター] を選択します。
3. サイズを変更するクラスターを選択します。
4. [アクション] で、[サイズ変更] を選択します。[クラスターのサイズ変更] ページが表示されます。
5. ページに表示された手順に従います。クラスターのサイズ変更は、特定の時刻に 1 回行うことも、スケジュールに従ってクラスターのサイズを増減することもできます。

6. 選択に応じて、[Resize now (今すぐサイズ変更)] または [Schedule resize (サイズ変更のスケジュール)] を選択します。

リザーブドノードがある場合は、RG または RA3 リザーブドノードにアップグレードできます。このアップグレードは、コンソールを使用してスナップショットからの復元を実行する場合や、伸縮自在なリサイズを実行する際に利用できます。コンソールを使用している場合は、このプロセスに関するガイドが提供されます。RG または RA3 ノードタイプへのアップグレードの詳細については、「RG または RA3 ノードタイプへのアップグレード」を参照してください。

サイズ変更オペレーションを実行して DC2.large ノードタイプから RA3.large ノードタイプにアップグレードすると、Amazon Redshift は自動的にインターリーブソートキーを複合ソートキーに変換します。この変換により、同時実行スケーリング機能にアクセスできます。この機能は、インターリーブソートキーを持つテーブルに対するクエリをサポートしていません。この自動変換により RA3 機能との互換性は保証されますが、既存のクエリパフォーマンスパターンに影響する可能性があります。

RA3 ノードにアップグレード後にインターリーブソートキーを維持する場合は、サイズ変更オペレーションの完了後に、必要なソートキー設定でテーブルを再作成できます。ただし、このオプションを選択すると、これらのテーブルで同時実行スケーリングを使用することはできません。

サイズ変更操作には次の 2 つのタイプがあります。

- 伸縮自在なサイズ変更 - クラスターにノードを追加または削除できます。DC2 ノードから RG または RA3 ノードへの変更など、ノードタイプを変更することもできます。伸縮自在なサイズ変更は、通常は短時間で完了し、平均で 10 分かかります。このため、最初のオプションとしてお勧めします。伸縮自在なサイズ変更を実行すると、データスライスを再分配します。データスライスは、各ノードにメモリとディスク領域に割り当てられるパーティションです。伸縮自在なサイズ変更は、次のような場合に適しています。
  - 既存のクラスターにノードを追加または削減するが、ノードタイプは変更しない場合 - これは一般的にインプレースサイズ変更と呼ばれます。このタイプのサイズ変更を実行すると、実行中のクエリの一部は正常に完了しますが、他のクエリは操作の一部として破棄される場合があります。
  - クラスターのノードタイプの変更 - ノードタイプを変更すると、スナップショットが作成されて、ソースクラスターから新しいノードタイプで構成されるクラスターにデータが再分配されます。完了すると、実行中のクエリは破棄されます。インプレースのサイズ変更のように、すぐに完了します。

- 従来のサイズ変更 - ノードタイプ、ノード数、またはその両方を、伸縮自在なサイズ変更と同様に変更できます。従来のサイズ変更は完了するまでに時間がかかりますが、ノード数の変更または移行先のノードタイプが、伸縮自在なサイズ変更の範囲内に収まらない場合は便利です。例えば、ノード数の変更が非常に大規模な場合に当てはまります。

トピック

- 伸縮自在なサイズ変更
- 従来のサイズ変更

伸縮自在なサイズ変更

同じタイプのノードを追加または削除する場合、伸縮自在なサイズ変更操作には、次の段階があります。

1. 伸縮自在なサイズ変更は、クラスターのスナップショットを作成します。バックアップ用ではないテーブルは DC2 ノードでのみサポートされます。RG および RA3 クラスターでは、バックアップ用ではないテーブルはスナップショットに含まれます。詳細については、「スナップショットのテーブルを除く」を参照してください。自動スナップショットを無効にしているため、クラスターに最近のスナップショットがない場合、バックアップオペレーションに時間がかかることがあります。(サイズ変更操作を開始する前の時間を最小限に抑えるため、自動スナップショットを有効にするか、サイズ変更を開始する前に手動スナップショットを作成することをお勧めします。) 伸縮自在なサイズ変更を開始し、スナップショット操作が現在進行中の場合、スナップショット操作が数分以内に完了しないと、伸縮自在なサイズ変更が失敗することがあります。詳細については、「Amazon Redshift スナップショットとバックアップ」を参照してください。
2. このオペレーションはクラスターのメタデータを移行します。クラスターは数分間使用できません。クエリの大部分は一時的に停止され、接続は開いた状態になります。ただし、一部のクエリは削除される可能性があります。この段階は短い。
3. セッション接続が回復し、クエリが再開します。
4. 伸縮自在なサイズ変更は、バックグラウンドでノードスライスにデータを再分配します。クラスターは読み取りと書き込み操作に利用できますが、一部のクエリは実行に時間がかかる可能性があります。
5. 操作が完了すると、Amazon Redshift はイベント通知を送信します。

伸縮自在なサイズ変更を使用してノードタイプを変更する操作は、同じタイプのノードを追加または削除する操作と似ています。まず、スナップショットが作成されます。新しいターゲットクラスターはスナップショットの最新データでプロビジョニングされ、データはバックグラウンドの新しいクラスターに転送されます。この期間中、データは読み取りのみ可能です。サイズ変更が完了間近になると、Amazon Redshift はエンドポイントを更新して新しいクラスターを指し、ソースクラスターへのすべての接続が破棄されます。

伸縮自在なサイズ変更が失敗することはまずありません。ただし、障害が発生した場合、ほとんどのケースでロールバックが自動的に行われ、手動による介入は必要ありません。

リザーブドノード (DS2 リザーブドノードなど) がある場合、サイズ変更を実行する際に RA3 リザーブドノードにアップグレードできます。このアップグレードは、伸縮自在なリサイズを実行するか、コンソールを使用してスナップショットからの復元を実行するときに使用できます。このコンソールは、このプロセスについて説明します。RG または RA3 ノードタイプへのアップグレードの詳細については、「RG または RA3 ノードタイプへのアップグレード」を参照してください。

伸縮自在なサイズ変更は、テーブルをソートしたり、ディスク容量を解放したりしないため、バキュームオペレーションに代わるものではありません。詳細については、「テーブルのバキューム処理」を参照してください。

伸縮自在なサイズ変更には以下の制約があります。

- 伸縮自在なサイズ変更とデータ共有クラスター - データ共有のプロデューサーであるクラスターでノードを追加または削除すると、Amazon Redshift がクラスターメタデータを移行している間、コンシューマーから接続できません。同様に、伸縮自在なサイズ変更を実行して新しいノードタイプを選択した場合、接続がドロップされ、新しいターゲットクラスターに転送される間、データ共有は利用できません。どちらのタイプの伸縮自在なサイズ変更でも、プロデューサーは数分間利用できません。
- 共有スナップショットからのデータ転送 - 共有スナップショットからデータを転送しているクラスターで伸縮自在なサイズ変更を実行するには、クラスターで少なくとも 1 つのバックアップを使用できる必要があります。バックアップは、Amazon Redshift コンソールのスナップショットのリスト、`describe-cluster-snapshots` CLI コマンド、または `DescribeClusterSnapshots` API オペレーションで表示できます。
- プラットフォーム制限 - 伸縮自在なサイズ変更は、EC2-VPC プラットフォームを使用するクラスターでのみ利用できます。詳細については、「EC2 を使用してクラスターを作成する」を参照してください。
- ストレージの考慮事項 - 新しいノード設定では、既存データに十分なストレージを確保する必要があります。ノードの追加または設定の変更が必要な場合があります。

- ソース vs ターゲットクラスターサイズ - 伸縮自在なサイズ変更によってサイズ変更可能なノード数と種類は、ソースクラスターのノード数と、サイズ変更したクラスター用に選択されたノードタイプによって決まります。使用可能な設定を確認するには、コンソールを使用します。また、`action-type resize-cluster` オプションで `describe-node-configuration-options` AWS CLI コマンドを使用することもできます。Amazon Redshift コンソールを使用したメタデータの編集の詳細については、クラスターのサイズ変更 を参照してください。

  次の CLI コマンドの例では、使用可能な設定オプションを確認できます。この例では、`mycluster` という名前のクラスターは `rg.4xl` 8 ノードクラスターです。

  ```
  aws redshift describe-node-configuration-options --cluster-identifier mycluster --
  region eu-west-1 --action-type resize-cluster
  ```

  このコマンドは、各オプションの推奨ノードタイプ、ノード数、およびディスク使用率を含むオプションリストを返します。返される設定は、特定の入力クラスターに基づいて異なります。`resize-cluster` CLI コマンドのオプションを指定するときに、これらの返された設定のいずれかを選択できます。

- 追加ノードの上限 - 伸縮自在なサイズ変更には、クラスターに追加できるノードに制限があります。例えば、RG および RA3 クラスターでは、ノード数が 4 倍までの伸縮自在なサイズ変更をサポートします。例えば、4 ノード型 rg.4xl クラスターにノードを追加して 5 ノードのクラスターにしたり、16 ノードになるまでさらにノードを追加できます。

  > **Note**
  >
  > 拡大と縮小の制限は、元のノードタイプ、元のクラスター内のノード数、または最後に行った従来のサイズ変更に基づいて決まります。伸縮自在なサイズ変更が拡大または縮小の制限を超える場合は、従来のサイズ変更を使用してください。

  rg および ra3 ノードタイプには、ノード数を既存の数の 4 倍まで増やすことができるものもあります。例えば、クラスターが ra3.4xlarge ノードまたは ra3.16xlarge ノードで構成されているとします。この場合、伸縮自在なサイズ変更を使用して、8 ノードのクラスターのノード数を 32 に増やすことができます。または、制限値を下回る値も選択できます。(クラスターを 4 倍に拡張できるかどうかは、ソースクラスターのサイズによることに注意してください。) クラスターに ra3.xlplus ノードがある場合、制限値は 2 倍になります。

すべての rg および ra3 ノードタイプでは、ノード数を既存の数の 4 分の 1 に減らすことができます。例えば、ra3.4xlarge ノードを持つクラスターのサイズを 12 ノードから 3 ノードに、または最小値を超えた値に減らすことができます。

次の表は、伸縮自在なサイズ変更がサポートされている各ノードタイプの増加の制限値と削減の制限値を示しています。

元のノードタイプ	増加制限値	削減制限値
rg.12xlarge	4 倍	数字の 4 分の 1
rg.4xlarge	4 倍	数字の 4 分の 1
rg.xlarge	2 倍	数字の 2 分の 1
rg.large	2 倍	数字の 2 分の 1
ra3.16xlarge	4 倍 (例えば、4 ノードから 16 ノード)	数字の 4 分の 1 (例えば、16 から 4 ノード)
ra3.4xlarge	4 倍	数字の 4 分の 1
ra3.xlplus	2 倍 (例えば、4 ノードから 8 ノード)	数字の 4 分の 1
ra3.large	2 倍	数字の 2 分の 1
dc2.8xlarge	2 倍	数字の 2 分の 1 (例えば、16 から 8 ノード)
dc2.large	2 倍	数字の 2 分の 1

> **Note**
>
> RG または RA3 クラスターのサイズを変更するときのレガシーノードタイプの選択 — RG または RA3 ノードを含むクラスターから別のノードタイプ (DC2 など) にサイズを変更しようとすると、検証警告メッセージがコンソールに表示され、サイズ変更オペレーションは完了しません。これは、レガシーノードタイプへのサイズ変更がサポートされていない

ためです。これにより、お客様が非推奨または間もなく非推奨になるノードタイプへのサイズ変更をできないようにしています。これは、伸縮自在なサイズ変更と従来のサイズ変更の両方に当てはまります。

従来のサイズ変更

伸縮自在なサイズ変更でサポートされないクラスターサイズやノードタイプの変更が伴うユースケースは、従来のサイズ変更で処理します。従来のサイズ変更を実行すると、Amazon Redshift はターゲットクラスターを作成し、データとメタデータをソースクラスターからそのクラスターに移行します。

RG または RA3 への従来のサイズ変更では、可用性が向上します

ターゲットノードタイプが RG または RA3 の場合、従来のサイズ変更が強化されています。これを行うために、ソースとターゲットのクラスター間でバックアップと復元オペレーションを利用します。サイズ変更が始まると、ソースクラスターが再起動し、数分間使用できなくなります。その後、クラスターは読み取りおよび書き込みオペレーションで使用可能になり、サイズ変更はバックグラウンドで続行します。

クラスターの確認

RG または RA3 クラスターへの従来のサイズ変更を実行したときに最高のパフォーマンスと結果を得るには、次のチェックリストを完了します。チェックリストに従わないと、読み取りまたは書き込みオペレーションの実行など、RG または RA3 ノードでの従来のサイズ変更のメリットの一部が得られない場合があります。

1. データのサイズは 2 ペタバイト未満でなければなりません。(1 ペタバイトは 1,000 テラバイトに相当します)。データのサイズを検証するには、スナップショットを作成してそのサイズを確認します。次のクエリを実行してサイズを確認することもできます。

   ```
   SELECT
   sum(case when lower(diststyle) like ('%key%') then size else 0 end) distkey_blocks,
   sum(size) as total_blocks,
   ((distkey_blocks/(total_blocks*1.00)))*100 as Blocks_need_redist
   FROM svv_table_info;
   ```

   svv_table_info テーブルはスーパーユーザーにのみ表示されます。

2. 従来のサイズ変更を開始する前に、10 時間以内の手動スナップショットがあることを確認してください。存在しない場合は、スナップショットを作成します。
3. 従来のサイズ変更の実行に使用したスナップショットは、テーブルの復元やその他の目的には使用できません。
4. クラスターは VPC 内にある必要があります。

RG または RA3 への従来のサイズ変更によるソートおよび分散オペレーション

RG または RA3 への従来のサイズ変更中に、EVEN 分散として移行された分キー分散のあるテーブルは、元の分散スタイルに戻されます。この期間は、データのサイズとクラスターの負荷状況によって異なります。クエリワークロードは、データ移行よりも実行が優先されます。詳細については、「分配スタイル」を参照してください。この移行プロセス中は、データベースの読み取りと書き込みの両方が機能しますが、クエリが完了するまでに時間がかかることがあります。ただし、同時実行スケーリングでは、クエリワークロード用のリソースを追加することでこの間にパフォーマンスを向上させることができます。SYS_RESTORE_STATE ビューと SYS_RESTORE_LOG ビューの結果を表示することで、データ移行の進行状況を確認できます。モニタリングの詳細については、以下を参照してください。

クラスターのサイズが完全に変更されると、次のソート動作が発生します。

- サイズ変更によってクラスターのスライス数が増えると、KEY 分散テーブルは部分的にソートされなくなりますが、EVEN テーブルはソートされたままになります。また、ソートされたデータの量に関する情報は、サイズ変更の直後は最新でない可能性があります。キーの回復後、自動バキュームによってテーブルが時間の経過とともにソートされます。
- サイズ変更によってクラスターのスライス数が少なくなると、KEY 分散テーブルと EVEN 分散テーブルの両方が部分的にソートされなくなります。自動バキュームにより、テーブルが時間の経過とともにソートされます。

テーブルの自動バキュームの詳細については、「テーブルのバキューム処理」を参照してください。コンピューティングノードのスライスの詳細については、「データウェアハウスシステムアーキテクチャ」を参照してください。

ターゲットクラスターが RG または RA3 である場合の従来のサイズ変更手順

従来のサイズ変更は、ターゲットクラスタータイプが RG または RA3 で、前のセクションで説明した前提条件を満たしている場合、次の手順で構成されます。

1. ソースクラスターからターゲットクラスターへの移行が開始されます。新しいターゲットクラスターがプロビジョニングされると、Amazon Redshift はサイズ変更が開始された旨のイベント通知を送信します。これにより、既存のクラスターが再起動され、すべての接続が閉じられます。既存のクラスターがデータ共有プロデューサークラスターの場合、コンシューマークラスターとの接続も閉じられます。再起動には数分かかります。
2. 再起動後、データベースは読み取りと書き込みが可能になります。さらに、データ共有が再開されます。これにはさらに数分かかります。
3. データがターゲットクラスターに移行されます。ターゲットノードタイプが RG または RA3 の場合、データ移行中に読み取りと書き込みが可能です。
4. サイズ変更プロセスが完了間近になると、Amazon Redshift はターゲットクラスターのエンドポイントを更新し、ソースクラスターへのすべての接続は終了します。ターゲットクラスターは、データ共有のプロデューサーになります。
5. サイズ変更の完了です。Amazon Redshift がイベント通知を送信します。

サイズ変更の進行状況は、Amazon Redshift コンソールで確認できます。クラスターのサイズ変更にかかる時間は、データ量に左右されます。

> **Note**
>
> RG または RA3 クラスターのサイズを変更するときのレガシーノードタイプの選択 — RG または RA3 ノードを含むクラスターから別のノードタイプ (DC2 など) にサイズを変更しようとすると、検証警告メッセージがコンソールに表示され、サイズ変更オペレーションは完了しません。これは、レガシーノードタイプへのサイズ変更がサポートされていないためです。これにより、お客様が非推奨または間もなく非推奨になるノードタイプへのサイズ変更をできないようにしています。これは、伸縮自在なサイズ変更と従来のサイズ変更の両方に当てはまります。

ターゲットクラスターが RG または RA3 である場合の従来のサイズ変更のモニタリング

進行中のプロビジョニングされたクラスターの従来のサイズ変更 (キー分散を含む) をモニタリングするには、SYS_RESTORE_STATE を使用します。変換中のテーブルの完了率が表示されます。データにアクセスするには、スーパーユーザーである必要があります。

従来のサイズ変更を実行するときに不要なテーブルを削除します。これを行うと、既存のテーブルをより迅速に分散できます。

ターゲットクラスターが RG または RA3 でない場合の従来のサイズ変更手順

ターゲットノードタイプが RG または RA3 以外 (DC2 など) である場合、従来のサイズ変更の手順は次のとおりです。

1. ソースクラスターからターゲットクラスターへの移行が開始されます。新しいターゲットクラスターがプロビジョニングされると、Amazon Redshift はサイズ変更が開始された旨のイベント通知を送信します。これにより、既存のクラスターが再起動され、すべての接続が閉じられます。既存のクラスターがデータ共有プロデューサークラスターの場合、コンシューマークラスターとの接続も閉じられます。再起動には数分かかります。

   `BACKUP NO` で作成したテーブルやマテリアライズドビューなどのデータベースリレーションは、従来のサイズ変更では保持されないことに注意してください。詳細については、「CREATE MATERIALIZED VIEW」を参照してください。
2. 再起動後、データベースは読み取り専用になります。データ共有が再開されます。これにはさらに数分かかります。
3. データがターゲットクラスターに移行されます。データベースは読み取り専用のままです。
4. サイズ変更プロセスが完了間近になると、Amazon Redshift はターゲットクラスターのエンドポイントを更新し、ソースクラスターへのすべての接続は終了します。ターゲットクラスターは、データ共有のプロデューサーになります。
5. サイズ変更の完了です。Amazon Redshift がイベント通知を送信します。

サイズ変更の進行状況は、Amazon Redshift コンソールで確認できます。クラスターのサイズ変更にかかる時間は、データ量に左右されます。

> **Note**
>
> ターゲットクラスターが RG または RA3 でない場合、または前のセクションで説明した RA3 ターゲットクラスターの前提条件を満たしていない場合、大量のデータを含むクラスターのサイズを変更するには、数日または場合によっては数週間かかることがあります。また、クラスターの使用済みストレージ容量は、従来のサイズ変更後に増加する可能性があることにも注意してください。これは、従来のサイズ変更の結果としてクラスターにデータスライスが追加された場合、通常のシステム動作です。クラスター内のノード数が同じままでも、こうした追加容量の使用が発生する場合があります。

伸縮自在なサイズ変更 vs 従来のサイズ変更

次の表は、2 つのサイズ変更タイプの動作を比較しています。

動作	伸縮自在なサイズ変更	従来のサイズ変更	コメント
システムデータ保持	伸縮自在なサイズ変更は、システムログデータを保持します。	従来のサイズ変更は、システムテーブルとデータを保持しません。	ソースクラスターで監査ログ記録を有効にしている場合、サイズ変更をしたあと、Amazon S3 または CloudWatch でログへのアクセスを継続できま

動作	伸縮自在なサイズ変更	従来のサイズ変更	コメント			
			す。これらのログは、指定したデータポリシーに応じて保持または削除することができます。			

動作	伸縮自在なサイズ変更	従来のサイズ変更	コメント				
ノードタイプの変更	ノードタイプが変わらない場合、伸縮自在なサイズ変更: インプレースのサイズ変更すると、ほとんどのクエリが保持されます。 新しいノードタイプを選択した状態で伸縮自在なサイズ変更: 新しいクラスターが作成されます。サイズ変更プロセスが完了すると、クエリは破棄されます。	クラシックサイズ変更: 新しいクラスターが作成されます。サイズ変更プロセス中にクエリは破棄されます。					

動作	伸縮自在なサイズ変更	従来のサイズ変更	コメント			
セッションとクエリの保持	伸縮自在なサイズ変更では、ソースクラスターとターゲットでノードタイプが同じである場合、セッションとクエリが保持します。新しいノードタイプを選択する場合、クエリは破棄されます。	クラシックサイズ変更では、セッションとクエリは保持されません。クエリが破棄されます。	クエリを削除すると、パフォーマンスが低下することが予想されます。サイズ変更操作は、使用負荷が低い時間に行うのが最適です。			

動作	伸縮自在なサイズ変更	従来のサイズ変更	コメント				
サイズ変更操作のキャンセル	伸縮自在なサイズ変更をキャンセルすることはできません。	Amazon Redshift コンソールのクラスター詳細から [Cancel resize (サイズ変更のキャンセル)] を選択すると、従来のサイズ変更オペレーションが完了する前にキャンセルできます。	サイズ変更のキャンセルに要する時間は、キャンセルするとき、サイズ変更オペレーションのどのステージにあるかによって異なります。				

動作	伸縮自在なサイズ変更	従来のサイズ変更	コメント				
			この操作を実行すると、キャンセル操作が完了するまでクラスターを利用できません。サイズ変更操作が最終ステージに達している場合、キャ				

動作	伸縮自在なサイズ変更	従来のサイズ変更	コメント				
			ンセルできません。 RG または RA3 クラスターへの従来のサイズ変更では、キャンセルできません。				

サイズ変更のスケジューリング

クラスターのサイズ変更オペレーションをスケジュールして、使用率の増加を予想してスケールアップしたり、コスト削減のためにスケールダウンしたりすることができます。スケジューリングは、伸縮自在なサイズ変更と従来のサイズ変更の両方に使えます。Amazon Redshift コンソールでスケジュールのセットアップできます。詳細については、[Managing clusters using the console] (コンソールを使ってクラスター管理) の「クラスターのサイズ変更」を参照してください。AWS CLI または Amazon Redshift API 操作を使用して、サイズ変更をスケジュールすることもできます。詳細

については、AWS CLI コマンドリファレンスの「create-scheduled-action」または Amazon Redshift API リファレンスの「CreateScheduledAction」を参照してください。

スナップショット、リストア、およびサイズ変更

伸縮自在なサイズ変更は、Amazon Redshift クラスターのサイズを変更する最速の方法です。伸縮自在なサイズ変更オプションを選択できず、クラスターにほぼ恒常的な書き込みアクセスが必要な場合は、次のセクションで説明している、スナップショットと復元オペレーションを使用します。この方法では、スナップショットが作成された後でソースクラスターに書き込まれたデータは、ターゲットクラスターに切り替えた後、手動でコピーする必要があります。コピーにかかる時間によっては、両方のクラスター内のデータが同じになるまで、この操作を数回繰り返す必要がある場合もあります。その後で、ターゲットクラスターに切り替えられます。このプロセスは、ターゲットクラスターのすべてのデータが使用可能になるまでに、既存のクエリに悪影響を及ぼす可能性があります。ただし、データベースへの書き込みができない時間は最短になります。

スナップショット、復元、従来のサイズ変更アプローチでは、次のプロセスを使用します。

1. 既存のクラスターのスナップショットを作成します。既存のクラスターがソースクラスターです。
2. スナップショットを作成した時刻を記録します。そうすることで、スナップショット後のデータをターゲットデータベースにロードするための抽出、処理、ロード (ETL) プロセスを再実行する必要がある時点を識別できるようにします。
3. 新しいクラスターにスナップショットを復元します。この新しいクラスターがターゲットクラスターです。サンプルデータがターゲットクラスターにあることを確認します。
4. ターゲットクラスターのサイズを変更します。ターゲットクラスターに関して、新しいノードタイプ、ノード数、その他の設定を選択します。
5. ソースクラスターのスナップショット作成後に発生した ETL プロセスでロードされたデータを確認します。ターゲットクラスターには、同じデータを同じ順序で再ロードしてください。進行中のデータロードがある場合、ソースクラスターとターゲットクラスターのデータが同じになるまで、このプロセスを数回繰り返します。
6. ソースクラスターで実行中のすべてのクエリを停止します。これを行うには、クラスターを再起動するか、スーパーユーザーとしてログオンし、PG_CANCEL_BACKEND コマンドおよび PG_TERMINATE_BACKEND コマンドを使用できます。クラスターを再起動すると、クラスターが使用できないことを最も簡単に確認できます。
7. ソースクラスター名を変更します。たとえば、`examplecluster` から `examplecluster-source` に変更します。

8. 変更前のソースクラスター名を使用して、ターゲットクラスターの名前を変更します。たとえば、ターゲットクラスターの名前を examplecluster に変更します。これ以降、examplecluster を含むエンドポイントを使用するアプリケーションは、ターゲットクラスターに接続します。
9. ターゲットクラスターに切り替えた後、ソースクラスターを削除し、すべてのプロセスが期待どおりに動作することを確認します。

または、データをターゲットクラスターに再ロードする前にソースとターゲットクラスターの名前を変更することもできます。このアプローチは、依存するシステムやレポートをすぐに最新状態にしてターゲットクラスターに反映する必要がない場合に有効です。この場合、ステップ 6 は前述のプロセスの最後に移動されます。

名前変更プロセスは、アプリケーションが引き続き同じエンドポイントを使用してクラスターに接続する必要がある場合にのみ必要になります。これが必要ない場合は、クラスターの名前を変更せずにそのクラスターに接続するアプリケーションを、ターゲットクラスターのエンドポイントを使用するように更新することもできます。

クラスター名を再利用するのには、いくつかの利点があります。最初に、エンドポイントが変わらないため、基盤となるクラスターを変更しても、アプリケーションの接続文字列を更新する必要がありません。次に、Amazon CloudWatch アラームおよび Amazon Simple Notification Service (Amazon SNS) の通知などの関連項目が、クラスター名に固定されます。これは、クラスターにセットアップした同じアラームと通知を継続して使用することができるということです。この継続的な使用は、アラームや通知などの関連項目を再設定することなく、柔軟にクラスターのサイズを変更する必要がある本番稼働用環境では特に重要です。

クラスターの名前変更

クラスターに別の名前を使用する必要がある場合は、クラスターの名前を変更できます。クラスターのエンドポイントには、クラスター名 (クラスター識別子とも呼ばれる) が含まれているため、名前の変更が完了した後、新しい名前を使用するようにエンドポイントを変更します。たとえば、examplecluster という名前のクラスターを、newcluster という名前に変更した場合は、newcluster 識別子を使用するようにエンドポイントを変更します。このクラスターに接続するアプリケーションは、新しいエンドポイントで更新する必要があります。

アプリケーションのエンドポイントを変更せずにアプリケーションの接続先のクラスターを変更する場合は、クラスターの名前を変更できます。この場合は、最初に元のクラスターの名前を変更してから、新しい接続先のクラスターを元のクラスターの名前に変更する必要があります。クラスター識別

子はアカウントとリージョン内で一意にする必要があり、元のクラスターと変更後のクラスターを同じ名前にできないため、この操作が必要になります。スナップショットからクラスターを復元するときに、依存アプリケーションの接続プロパティを変更したくない場合も、この操作を行います。

> **Note**
>
> 元のクラスターを削除する場合は、不要なクラスターのスナップショットを削除してください。

クラスターの名前を変更すると、クラスターの状態は、このプロセスが終了するまで `renaming` に変わります。クラスターに使用していた古い DNS 名は直ちに削除されますが、キャッシュには数分間残っています。名前を変更したクラスターの新しい DNS 名は、10 分以内で有効になります。名前を変更したクラスターは、新しい名前が有効になるまで使用できません。クラスターが再起動され、クラスターへの既存の接続は削除されます。これが完了すると、新しい名前を使用するようにエンドポイントが変更されます。そのため、名前の変更を開始する前にクエリの実行を停止し、名前の変更後に再起動する必要があります。

クラスターのスナップショットは保持され、クラスターに関連付けられたすべてのスナップショットは、クラスターの名前を変更した後も関連付けを維持します。たとえば、本番稼働用データベースにサービスを提供するクラスターがあり、そのクラスターに複数のスナップショットがあるとします。クラスターの名前を変更し、スナップショットのある本番稼働用環境に置き換えると、名前を変更したクラスターに既存のスナップショットが関連付けられます。

Amazon CloudWatch アラームおよび Amazon Simple Notification Service (Amazon SNS) イベント通知は、クラスター名に関連付けられます。クラスターの名前を変更した場合は、これらも更新する必要があります。CloudWatch アラームは CloudWatch コンソールで更新できます。また、[Events (イベント)] ペインの Amazon Redshift コンソールで Amazon SNS イベント通知を更新できます。クラスターのロードおよびクエリデータには、名前変更前と名前変更後のデータが引き続き表示されます。ただし、パフォーマンスデータは、名前変更プロセスの完了後にリセットされます。

詳細については、「クラスターの変更」を参照してください。

クラスターのリリースバージョンのアップグレード

[リリースステータス] 値が [新しいリリースが利用可能] になっているクラスターのリリースメンテナンスバージョンをアップグレードできます。メンテナンスバージョンをアップグレードすると、すぐにアップグレードするか、次のメンテナンス期間にアップグレードするかを選択できます。

> **Important**
>
> すぐにアップグレードする場合、アップグレードが完了するまでクラスターがオフラインになります。

クラスターを新しいリリースバージョンにアップグレードするには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [クラスター] を選択します。
3. アップグレードするクラスターを選択します。
4. アクションで、Upgrade cluster version (クラスターバージョンのアップグレード)] を選択します。[Upgrade cluster version (クラスターバージョンのアップグレード)] ページが表示されます。
5. ページに表示された手順に従います。
6. [Upgrade cluster version (クラスターバージョンのアップグレード)] を選択します。

クラスターの一時停止と再開

特定の時間帯にのみ使用可能にする必要があるクラスターがある場合は、そのクラスターを一時停止して後で再開することができます。クラスターが一時停止している間は、オンデマンド課金は一時停止されます。課金されるのは、クラスターのストレージのみです。料金の詳細については、Amazon Redshift 料金表を参照してください。

クラスターを一時停止すると、Amazon Redshift はスナップショットを作成し、クエリの終了を開始して、クラスターを一時停止状態にします。一時停止したクラスターを最終スナップショットをリクエストせずに削除した場合、そのクラスターを復元することはできません。一時停止または再開オペレーションは、開始後にキャンセルまたはロールバックすることはできません。

クラスターの一時停止と再起動は、Amazon Redshift コンソール、AWS CLI、または Amazon Redshift API オペレーションで行うことができます。

クラスターの一時停止と再開は、アクションをスケジュールして行うことができます。新しい Amazon Redshift コンソールを使用して一時停止と再開を行う定期的なスケジュールを作成すると、選択した日付範囲に対して 2 つのスケジュールされたアクションが作成されます。スケジュールさ

れたアクションの名前には、末尾に -pause と -resume が付けられます。名前の合計の長さは、スケジュールされたアクション名の最大サイズに収まる必要があります。

次のタイプのクラスターは一時停止できません。

- EC2-Classic クラスター。
- アクティブではないクラスター (現在変更中のクラスターなど)。
- ハードウェアセキュリティモジュール (HSM) クラスター。
- 自動スナップショットがオフになっているクラスター。

クラスターを一時停止する場合は、次の点を考慮してください。

- クラスターへの接続またはクエリは行えません。
- 一時停止したクラスターのクエリのモニタリング情報を Amazon Redshift コンソールに表示することはできません。
- 一時停止したクラスターを変更することはできません。クラスターでスケジュールされたアクションは実行されません。これには、スナップショットの作成、クラスターのサイズ変更、クラスターのメンテナンスなどのオペレーションが含まれます。
- ハードウェアのメトリクスは作成されません。作成されないメトリクスに CloudWatch のアラームを設定している場合は、アラームを更新してください。
- 一時停止したクラスターの最新の自動スナップショットを手動スナップショットにコピーすることはできません。
- クラスターが一時停止している間は、一時停止オペレーションが完了するまで再開することはできません。
- クラスターを一時停止すると、課金は一時停止されます。ただし、一時停止オペレーションは、クラスターのサイズに応じて通常 15 分以内に完了します。
- 監査ログはアーカイブされ、再開時には復元されません。
- クラスターを一時停止すると、一時停止前に発生した問題のトラブルシューティングにトレースとログが使用できなくなる場合があります。
- AWS Secrets Manager を使用して管理者認証情報を管理していて、クラスターを一時停止しても、クラスターのシークレットは削除されず、シークレットの料金は引き続き請求されます。AWS Secrets Manager で Redshift 管理者パスワードを管理する方法の詳細については、「AWS Secrets Manager を使用した Amazon Redshift 管理者パスワードの管理」を参照してください。

- クラスター上のバックアップしないテーブルは、RG および RA3 インスタンスタイプの再開時に復元されません。DC2 インスタンスタイプの再開時には復元されません。バックアップ用ではないテーブルの詳細については、「スナップショットのテーブルを除く」を参照してください。

クラスターを再開する場合は、次の点を考慮してください。

- 再開されたクラスターのクラスターバージョンは、クラスターのメンテナンスウィンドウに基づいてメンテナンスバージョンに更新されます。
- 一時停止したクラスターに関連付けられているサブネットを削除すると、互換性のないネットワークが存在することになる場合があります。その場合は、最新のスナップショットからクラスターを復元してください。
- クラスターの一時停止中に Elastic IP アドレスを削除すると、新しい Elastic IP アドレスを求められます。
- Amazon Redshift が前の Elastic Network Interface でクラスターを再開できない場合、Amazon Redshift は新しい Elastic Network Interface を割り当てようとします。
- クラスターを再開すると、ノードの IP アドレスが変更される場合があります。それらの新しい IP アドレスを Secure Shell (SSH) の COPY や Amazon EMR の COPY などの機能でサポートするように、VPC の設定を更新する必要がある場合があります。
- 一時停止していないクラスターを再開しようとすると、再開オペレーションはエラーを返します。再開オペレーションがスケジュールされたアクションの一部である場合は、今後のエラーを防ぐためにスケジュールされたアクションを変更または削除してください。
- クラスターのサイズによっては、クラスターを再開してクエリを処理できるようになるまでに数分かかる場合があります。また、再開の完了後にクラスターが元の状態に戻るまでは、クエリのパフォーマンスにある程度の時間影響がある場合があります。

クラスターの再起動

クラスターの再起動は、再起動前と同じ設定でクラスターを再起動するクラスターオペレーションです。クラスターを再起動して、保留中のメンテナンス更新の適用、設定変更のリセット、特定の問題からの復旧、クラスター問題のトラブルシューティングを行うことができます。クラスターを再起動すると、Amazon Redshift 環境の最適なパフォーマンス、セキュリティ、安定性を確保できます。次の手順では、Amazon Redshift クラスターを再起動するための詳細な手順を示します。

クラスターを再起動すると、クラスターの状態が `rebooting` に設定されます。再起動が終了すると、クラスターイベントが作成されます。保留されていたクラスターへの変更はすべて、この再起動時に適用されます。

クラスターを再起動するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [クラスター] を選択します。
3. 再起動するクラスターを選択します。
4. [アクション] で、[クラスターの再起動] を選択します。[クラスターの再起動] ページが表示されます。
5. [クラスターの再起動] を選択します。

クラスターの再配置

Amazon Redshift で再配置を使用することにより、Amazon Redshift が、データの損失やアプリケーションへの変更なしに、クラスターを別のアベイラビリティーゾーン (AZ) に移動させることができます。再配置により、クラスター上でサービスが中断された場合でも、影響を最小限に抑えて操作を続行できます。

クラスターの再配置をオンにすると、Amazon Redshift は状況によってはクラスターの再配置を選択することがあります。特に、現在のアベイラビリティーゾーンの問題がクラスターの最適な動作を妨げている場合や、サービスの可用性を向上させる場合に再配置が発生します。特定のアベイラビリティーゾーンのリソース制約によってクラスター操作が中断される場合は、再配置関数を呼び出すこともできます。例として、クラスターを再開またはサイズ変更する機能があります。Amazon Redshift では、追加料金はかかりません。

Amazon Redshift クラスターが新しいアベイラビリティーゾーンに再配置されると、新しいクラスターは元のクラスターと同じエンドポイントを持ちます。アプリケーションはエンドポイントに再接続し、変更やデータの損失なしに操作を続行できます。ただし、特定のアベイラビリティーゾーンで潜在的なリソース制約が原因で、再配置が必ずしも可能とは限りません。

Amazon Redshift クラスターの再配置は、RG および RA3 インスタンスタイプでサポートされています。RG および RA3 インスタンスタイプでは、Redshift マネージドストレージ (RMS) を耐久性のあるストレージレイヤーとして使用します。クラスターデータの最新のコピーは、常に AWS リージョンの他のアベイラビリティーゾーンで使用できます。つまり、データを失うことなく、Amazon Redshift クラスターを別のアベイラビリティーゾーンに再配置できます。

クラスターの再配置をオンにすると、Amazon Redshift はクラスターをプロキシの背後に移行します。これにより、クラスタのコンピュートリソースへの口ケーションに依存しないアクセスを実装で

きます。移行により、クラスターが再起動されます。クラスターが別のアベイラビリティーゾーンに再配置されると、新しいクラスターが新しいアベイラビリティーゾーンでオンラインに戻される間、停止が発生します。ただし、クラスターが新しいアベイラビリティーゾーンに再配置された後もクラスターエンドポイントは変更されないため、アプリケーションを変更する必要はありません。

クラスターの再配置は、サブネットグループに複数のアベイラビリティーゾーンが含まれている、新しく作成または復元された RG または RA3 クラスターでデフォルトで有効になっています。Amazon Redshift は、プロビジョニングされたクラスターを作成するときに、5439 をデフォルトポートとして割り当てます。5431-5455 または 8191-8215 のポート範囲から別のポートに変更できます。(範囲外のポートには変更しないでください。エラーが発生します。) プロビジョニングされたクラスターのデフォルトポートを変更するには、Amazon Redshift コンソール、AWS CLI、または Amazon Redshift API を使用します。サーバーレスワークグループのデフォルトのポートを変更するには、AWS CLI または Amazon Redshift Serverless API を使用します。

再配置をオンにしていて、現在リーダーノードの IP アドレスを使用してクラスターまたは拡張された VPC ルーティングにアクセスしている場合は、そのアクセスを必ず変更してください。代わりに、クラスターの仮想プライベートクラウド (VPC) エンドポイントに関連付けられている IP アドレスを使用します。このクラスター IP アドレスを見つけるには、クラスターの詳細ページの [Network and security] (ネットワークとセキュリティ) セクションで VPC エンドポイントを見つけて使用します。VPC エンドポイントの詳細を表示するには、Amazon VPC コンソールにサインインします。

AWS Command Line Interface (AWS CLI) コマンド `describe-vpc-endpoints` を使用して、エンドポイントに関連付けられた Elastic Network Interface を取得することもできます。`describe-network-interfaces` コマンドを使用して、関連付けられた IP アドレスを取得できます。Amazon Redshift の AWS CLI コマンドの詳細については、AWS CLI コマンドリファレンスの「Available commands」を参照してください。

制限事項

Amazon Redshift 再配置を使用する際は、次の制限事項に注意してください。

- クラスターの再配置は、特定のアベイラビリティーゾーンで潜在的なリソース制限が原因で、すべてのシナリオで可能ではない場合があります。この場合、Amazon Redshift は元のクラスターを変更しません。
- 再配置は、製品の DC2 インスタンスファミリーではサポートされません。
- AWS リージョンをまたぐ再配置は実行できません。
- Amazon Redshift 再配置のデフォルトはポート番号 5439 です。5431 ~ 5455 または 8191 ~ 8215 のポート範囲から別のポートに変更できます。

コンソールを使用した再配置の管理

クラスターの再配置の設定は、Amazon Redshift コンソールを使用して管理できます。

新しいクラスターの作成時に再配置をオフにする

新しいクラスターを作成しているときに再配置をオフにするには、次の手順を使用します。

新しいクラスターの再配置をオフにするには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [クラスター] を選択します。
3. ここでクラスターを作成するには、[クラスターの作成] を選択します。クラスターの作成方法の詳細については、「Amazon Redshift 入門ガイド」の「Amazon Redshift でプロビジョニングされたデータウェアハウス」を参照してください。
4. [バックアップ]、[クラスターの再配置] で、[無効] を選択します。デフォルトでは、再配置は有効化されています。
5. [クラスターの作成] を選択します。

既存のクラスターの再配置の変更

既存のクラスターの再配置設定を変更するには、次の手順を使用します。

既存のクラスターの再配置設定を変更するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [クラスター] を選択します。現在のAWSリージョンにあるアカウントのクラスターがリストされています。各クラスターのプロパティのサブセットが、リストの列に表示されます。
3. クラスターのリストで、変更するクラスターの名前を選択します。クラスターの詳細ページが表示されます。
4. [Maintenance] (メンテナンス) タブを選択し、[Backup details] (バックアップの詳細) セクションで [Edit] (編集) を選択します。
5. [バックアップ] で、[有効] を選択します。デフォルトでは、再配置は有効化されています。
6. [Modify cluster] (クラスターの変更) を選択します。

クラスターの再配置

クラスターを別のアベイラビリティーゾーンに手動で再配置するには、次の手順を使用します。これは、セカンダリアベイラビリティーゾーンでネットワーク設定をテストする場合や、現在のアベイラビリティーゾーンでリソース制約が発生している場合に特に便利です。

クラスターを別のアベイラビリティーゾーンに再配置するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [クラスター] を選択します。現在のAWSリージョンにあるアカウントのクラスターがリストされています。各クラスターのプロパティのサブセットが、リストの列に表示されます。
3. リストから移動するクラスターの名前を指定します。クラスターの詳細ページが表示されます。
4. [Actions] (アクション) で、[Relocate](再配置) を選択します。[Relocate cluster] (クラスターの削除) ページが表示されます。
5. (オプション) [Availability Zone] (アベイラビリティーゾーン) を選択します。アベイラビリティーゾーンを選択しない場合、Amazon Redshift によってアベイラビリティーゾーンが選択されます。

Amazon Redshift によって再配置が開始され、クラスターが再配置として表示されます。再配置が完了すると、クラスタのステータスが [available] に変わります。

Amazon Redshift CLI を使用した再配置の管理

AWS Command Line Interface (CLI) を使用して、クラスターの再配置の設定を管理できます。

次のコマンド例では、AWS CLI を使用して、再配置がオンになっている **mycluster** という名前の Amazon Redshift クラスターを作成します。

```
aws redshift create-cluster --cluster-identifier mycluster --number-of-nodes 2 --
master-username enter a username --master-user-password enter a password --node-type
 ra3.4xlarge --port 5439 --no-availability-zone-relocation
```

現在のクラスターが別のポートを使用している場合は、再配置をオンにするよう変更する前に、5431 ~ 5455 または 8191 ~ 8215 のポート範囲から使用するように変更する必要があります。デフォルトは 5439 です。以下のコマンド例では、クラスターが指定範囲内のポートを使用していない場合に備えて、ポートを変更します。

```
aws redshift modify-cluster --cluster-identifier mycluster --port 5439
```

以下のコマンド例には、Amazon Redshift クラスター上のアベイラビリティーゾーン再配置パラメータが含まれています。

```
aws redshift modify-cluster --cluster-identifier mycluster --availability-zone-
relocation
```

以下のコマンド例では、Amazon Redshift クラスターでアベイラビリティーゾーン再配置パラメータを無効化します。

```
aws redshift modify-cluster --cluster-identifier mycluster --no-availability-zone-
relocation
```

以下のコマンド例では、Amazon Redshift クラスターでの再配置を呼び出します。

```
aws redshift modify-cluster --cluster-identifier mycluster --availability-zone us-
east-1b
```

クラスターの使用制限の設定

最大 4 つの使用制限を追加して、次のそれぞれの使用量を制御できます。

- 同時実行スケーリング
- 追加のコンピューティングリソースを使用した自動最適化の実行
- Redshift Spectrum の使用
- リージョン間でのデータ共有

プロビジョニングされたクラスターの使用制限の設定

プロビジョニングされたクラスターの使用制限を設定する手順は、次のとおりです。

クラスターの使用制限を設定するには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. 制限を設定するプロビジョニングされたクラスターに移動します。

3. クラスターの詳細ページで、[アクション] ドロップダウンメニューから [使用制限を管理] を選択します。クラスターの [メンテナンス] タブを選択し、下にスクロールして [使用制限を作成] を選択することもできます。
4. 設定する使用制限で [制限を追加] を選択します。特定の機能に対して最大 4 つの制限を追加できます。
5. 使用制限の [期間] に、[毎日]、[毎週]、または [毎月] のいずれかを設定します。
6. [使用制限] を設定します。

   - 追加のコンピューティングリソース制限を使用して同時実行スケーリングと自動最適化を実行する場合、使用制限は Amazon Redshift が特定の期間に機能を使用して費やす時間になります。この場合、使用制限は時間および分単位で設定されます。
   - Redshift Spectrum の場合、使用制限は Amazon S3 からスキャンされたデータの量です。この場合、使用制限はテラバイト (TB) で設定されます。
   - クロスリージョンデータ共有の場合、使用制限は、プロデューサーリージョンからコンシューマーがクエリできるコンシューマーリージョンに転送されるデータの量を制限になります。この場合、使用制限はテラバイト (TB) で設定されます。
7. クラスターが制限に達したときに実行する Amazon Redshift の [アクション] を設定します。これには次のものが含まれます。

   - システムテーブルへのログ記録 - システムビュー SYS_QUERY_HISTORY にレコードを追加します。このビューの usage_limit 列をクエリして、クエリが制限を超えたかどうかを判断できます。
   - [アラート] - Amazon SNS を使用して通知サブスクリプションを設定し、制限に違反した場合に通知を送信します。既存の Amazon SNS トピックを選択するか、新しいトピックを作成するか、または選択せずに続行することもできます。
   - 機能の無効化 - 機能を無効にします。Amazon SNS を使用して通知を送信することもできます。ユーザーは引き続きクラスターを他のタスクに使用できます。

   最初の 2 つのアクションは情報提供を目的としていますが、最後のアクションは機能の使用をオフにします。
8. ページの下部にある [変更の保存] をクリックして制限を保存します。一度に複数の制限を設定すると、[変更を保存] によってすべての制限が一度に保存されます。

クラスターのシャットダウンと削除

クラスターの実行を停止して料金の発生を防ぐ場合は、クラスターをシャットダウンできます。シャットダウンするときに、オプションで最終スナップショットを作成できます。最終スナップショットを作成する場合、Amazon Redshift はクラスターの手動スナップショットを作成した後、クラスターをシャットダウンします。削除したクラスターと同じデータと構成を使って新しいクラスターをプロビジョニングする場合、この手動スナップショットが必要になります。手動スナップショットを使用すると、後でスナップショットを復元して、クラスターの使用を再開できます。

クラスターとそのデータが不要になった場合は、最終スナップショットを作成しないでシャットダウンすることができます。この場合、クラスターとデータは完全に削除されます。

クラスターのシャットダウン時に最終的な手動スナップショットを作成するかどうかにかかわらず、クラスターのシャットダウン後、クラスターに関連付けられた自動スナップショットはすべて削除されます。クラスターに関連付けられた手動スナップショットは保持されます。オプションの最終スナップショットも含めて、保持された手動スナップショットは、クラスターをシャットダウンするときに実行中のクラスターが他にない場合、または、実行中の Amazon Redshift クラスターで利用できる無料ストレージ枠を超えている場合、Amazon Simple Storage Service ストレージ料金が課金されます。スナップショットのストレージ料金の詳細については、Amazon Redshift の料金表ページを参照してください。

クラスターを削除すると、関連する AWS Secrets Manager シークレットもすべて削除されます。

クラスターを削除するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [クラスター] を選択します。
3. 削除するクラスターを選択します。
4. [アクション] で、[削除] を選択します。[クラスターの削除] ページが表示されます。
5. [クラスターを削除] を選択します。

> **Note**
>
> クラスターを削除して最終スナップショットを作成することを選択すると、クラスターで復元オペレーションが進行中の場合、Amazon Redshift は削除リクエストを停止します。こ

の場合、最終スナップショットなしでクラスターを削除することも、復元の完了後に最終スナップショットを作成してクラスターを削除することもできます。

Amazon Redshift スナップショットとバックアップ

スナップショットはクラスターのポイントインタイムバックアップです。スナップショットには、自動と手動の 2 つのタイプがあります。Amazon Redshift は、暗号化された Secure Sockets Layer (SSL) 接続を使用して、これらのスナップショットを Amazon S3 の内部に保存できます。

Amazon Redshift は、前回のスナップショット以降にクラスターに加えられた増分変更を追跡する、増分スナップショットを自動的に作成します。自動スナップショットは、スナップショットからクラスターを復元するために必要なすべてのデータを保持します。自動スナップショットをいつ作成するかを制御するためにスナップショットスケジュールを作成できます。また、いつでも手動スナップショットを作成することもできます。

スナップショットから復元すると、Amazon Redshift は新しいクラスターを作成し、すべてのデータをロードする前に新しいクラスターを使用できるようにするので、すぐに新しいクラスターのクエリを開始できます。クラスターは、アクティブなクエリに応じてスナップショットからデータをオンデマンドでストリーミングし、次に残りのデータをバックグラウンドでロードします。

クラスターを起動するとき、自動スナップショットと手動スナップショットの保持期間を設定できます。クラスターを変更して、自動スナップショットと手動スナップショットの保持期間を変更できます。スナップショットを作成するか、スナップショットを変更して、手動スナップショット保持期間を変更できます。

AWS マネジメントコンソール でスナップショットの詳細を表示するか、CLI または DescribeClusterSnapshots API アクションで describe-cluster-snapshots を呼び出して、スナップショットの進行状況をモニタリングできます。これにより、進行中のスナップショットについて、差分スナップショットのサイズ、転送速度、経過時間、および推定残り時間などの情報が表示されます。

バックアップを常にクラスターで使用できるようにするために、Amazon Redshift は、Amazon Redshift の管理対象の内部管理 Amazon S3 バケットにスナップショットを保存します。ストレージの料金を管理するには、自動スナップショットを保持する必要がある日数を評価し、それに応じて保持期間を設定します。不要になった手動スナップショットを削除します。バックアップストレージのコストに関する詳細については、Amazon Redshift の料金表ページを参照してください。

AWS Backup を使用してスナップショットを作成および復元することもできます。Backup は、AWS のサービス、クラウド、オンプレミスにわたってデータ保護の一元化と自動化に役立つフルマネージドサービスです。詳細については、「AWS Backup と Amazon Redshift の統合」を参照してください。AWS Backup の詳細については、「AWS Backup Developer Guide」の「What is AWS Backup?」を参照してください。

Amazon Redshift Serverless でのスナップショットとバックアップの操作

プロビジョニングされたクラスターと同様に、Amazon Redshift Serverless では、名前空間内のポイントインタイムのオブジェクトとデータとしてバックアップを行うことができます。Amazon Redshift Serverless のバックアップには、手動で作成するスナップショットと、Amazon Redshift Serverless が自動的に作成する復旧ポイントの 2 種類があります。Amazon Redshift Serverless のスナップショットの操作の詳細については、「スナップショットと復旧ポイント」を参照してください。

プロビジョニングされたクラスターからサーバーレス名前空間にスナップショットを復元することもできます。詳細については、「スナップショットからサーバーレス名前空間の復元」を参照してください。

自動スナップショット

自動スナップショットがクラスターに対して有効になると、Amazon Redshift は定期的にそのクラスターのスナップショットが作成されます。デフォルトでは、Amazon Redshift は約 8 時間ごと、または 1 ノードのデータが変更されるごとに 5 GB ごと (あるいはそのいずれか早い方) にスナップショットを作成します。データがノード数の 5 GB * を超える場合、自動スナップショット作成間の最短時間は 15 分です。または、自動スナップショットを作成するタイミングを制御するためにスナップショットスケジュールを作成することができます。カスタムスケジュールを使用している場合は、自動スナップショット間の最小時間は 1 時間です。自動スナップショットは、クラスターを作成するときデフォルトで有効になります。

> **Note**
>
> Amazon Redshift は、一時停止したデータ共有プロデューサークラスターの自動スナップショットを作成します。コンシューマーは一時停止したプロデューサーに書き込むことができるため、これらのスナップショットはそれらのデータ共有の書き込みをキャプチャし、復旧ポイントを提供します。以前は、Amazon Redshift はこれらのクラスターの一時停止中に自動スナップショットを作成しませんでした。これらのスナップショットは、クラスターの既存の自動スナップショットのスケジュールと保持期間に従い、アクションは必要ありませ

ん。データ共有プロデューサーではない一時停止中のクラスターの場合、Amazon Redshift はクラスターの一時停止中に自動スナップショットを作成しません。

自動スナップショットは、保持期間の終了時に削除されます。デフォルトの保持期間は 1 日ですが、Amazon Redshift コンソールを使用するか、Amazon Redshift API または CLI を使用してプログラムにより変更できます。

自動スナップショットを無効にするには、保持期間を 0 に設定します。自動スナップショットを無効にした場合、Amazon Redshift はスナップショットの取得を停止し、クラスターの既存の自動スナップショットを削除します。RG または RA3 ノードタイプでは、自動スナップショットを無効にすることはできません。RG または RA3 ノードタイプの自動保存期間を 1～35 日に設定できます。

自動スナップショットを削除できるのは Amazon Redshift のみです。手動で削除することはできません。Amazon Redshift では、スナップショットの保存期間終了時、クラスターの自動スナップショットを無効にした場合、またはクラスターを削除した場合に、自動スナップショットが削除されます。Amazon Redshift は、自動スナップショットを無効にするか、クラスターを削除するまで、最新の自動スナップショットを保持します。

自動スナップショットをもっと長い期間保持する場合は、そのコピーを手動スナップショットとして作成します。自動スナップショットは、保持期間が終わるまで保持されますが、対応する手動スナップショットは手動で削除するまで、または保持期間が終わるまで保持されます。

自動スナップショットのスケジュール

スナップショットを作成するタイミングを正確に制御するために、スナップショットスケジュールを作成し、それを 1 つ以上のクラスターにアタッチすることができます。スナップショットスケジュールを変更すると、関連付けられているすべてのクラスターのスケジュールが変更されます。クラスターにスナップショットスケジュールがアタッチされていない場合、クラスターはデフォルトの自動スナップショットスケジュールを使用します。

スナップショットスケジュール は、一連のスケジュールルールです。指定した間隔 (8 時間ごと、12 時間ごとなど) に基づいてシンプルなスケジュールルールを定義できます。特定の曜日、特定の時間、特定の期間にスナップショットを作成するためのルールを追加することもできます。ルールは Unix 互換の cron 式を使って定義することもできます。

スナップショットスケジュール形式

Amazon Redshift コンソールで、スナップショットスケジュールを作成できます。その後、スケジュールをクラスターにアタッチしてシステムスナップショットの作成をトリガーできます。スケ

ジュールは複数のクラスターにアタッチでき、スナップショットをトリガーするためにスケジュール内に複数の cron 定義を作成できます。

cron 構文を使用してスナップショットのスケジュールを定義できます。これらのスケジュールの定義は、変更された Unix 互換の cron 構文を使用します。協定世界時 (UTC) で時間を指定します。最大頻度 1 時間、最小精度 1 分のスケジュールを作成できます。

Amazon Redshift の変更された cron 式には 3 つの必須フィールドがあり、それらは空白で区切られます。

構文

```
cron(Minutes Hours Day-of-month Month Day-of-week Year)
```

フィールド	値	ワイルドカード
分	0 ~ 59	, - * /
時間	0 ~ 23	, - * /
日	1 ~ 31	, - * ? / L W
月	1 ~ 12 または JAN ~ DEC	, - * /
曜日	1 ~ 7 または SUN ~ SAT	, - * ? L #
年	1970 ~ 2199	, - * /

ワイルドカード

- ワイルドカード [,] (カンマ) には追加の値が含まれます。Day-of-week フィールドの、MON,WED,FRI は、月曜日、水曜日、金曜日を含みます。合計値はフィールドあたり 24 に制限されています。
- ワイルドカード [-] (ダッシュ) は範囲を指定します。Hour フィールドの、「1 ~ 15」は、指定した日の 1 時間から 15 時間を含みます。
- ワイルドカード [*] (アスタリスク) にはフィールドのすべての値が含まれます。Hours フィールドの、* にはすべての時間が含まれています。

- ワイルドカード [/] (スラッシュ) で増分を指定します。Hours フィールドで、**1/10** と入力して、その日の最初の時間から始めて、10 時間毎を指定できます (01:00、11:00、21:00 など)。
- [?] (疑問符) のワイルドカードは、任意を意味します。Day-of-month フィールドで 7 と入力し、7 日が何曜日であってもかまわない場合、Day-of-week フィールドに ? を入力できます。
- Day-of-month フィールドまたは Day-of-week フィールドにある [L] のワイルドカードは、月または週の最終日を指定します。
- Day-of-month フィールドの、ワイルドカード W は、平日を指定します。Day-of-month フィールドで、3W は月の 3 番目の平日に最も近い日を指定します。
- Day-of-week フィールドの # ワイルドカードは、月の指定された曜日の特定のインスタンスを指定します。例えば、3#2 は、月の第 2 火曜日を示します。3 は週の 3 番目の日 (火曜日) を示し、2 は月のそのタイプの 2 番目の日を示します。

> **Note**
>
> 「#」文字を使用する場合、曜日フィールドには 1 つの式しか定義できません。例えば、「3#1,6#3」は 2 つの式として解釈されるため、無効です。

制限

- Cron 式の Day-of-month フィールドと Day-of-week フィールドを同時に指定することはできません。一方のフィールドに値を指定する場合、もう一方のフィールドで [?] (疑問符) を使用する必要があります。
- スナップショットスケジュールは以下の頻度をサポートしていません。
  - 1 時間に 1 回を超える頻度でスケジュールされるスナップショット。
  - 1 日 (24 時間) に 1 回未満の頻度でスケジュールされるスナップショット。

  1 時間以内にスナップショットをスケジュールする結果になる重複したスケジュールがある場合、検証エラーが発生します。

スケジュールを作成するときは、以下のサンプルの cron 文字列を使用できます。

分	時間	曜日	意味			
0	14-20/1	火	毎週火曜日の午後 2 時から午後 8 時の間。			
0	21	MON-FRI	毎晩、月曜日～金曜日の午後9時。			
30	0/6	土 - 日	土曜日と日曜日は、その日の深夜 30 分過ぎ (00:30) から、6 時間ごとに増分されます。これにより、各日とも [00:30、06:30、12:30、および 18:30] にスナップショットが作成されます。			
30	12/4	*	毎日 12:30 から 4 時間ごとに増分します。これにより [12:30、16:30、20:30] となります。			

たとえば、毎日 15:15 から 2 時間ごとの増分のスケジュールに従って実行するとします。これにより [15:15、17:15、19:15、21:15、23:15] となりますが、次のように指定します。

```
cron(15 15/2 *)
```

スケジュールとして複数の cron スケジュール定義を作成できます。たとえば、次の AWS CLI コマンドでは、1 つのスケジュールに 2 つの cron スケジュールが含まれています。

```
create-snapshot-schedule --schedule-identifier "my-test" --schedule-definition "cron(0
 17 SAT,SUN)" "cron(0 9,17 MON-FRI)"
```

手動スナップショット

手動スナップショットはいつでも取得できます。デフォルトでは、手動スナップショットは、クラスターを削除した後も、無限に保持されます。手動スナップショットを作成するときに保持期間を指定できます。スナップショットを変更して保持期間を変更することもできます。ログ保持期間の変更の詳細については、「手動スナップショットの保持期間を変更する」を参照してください。

スナップショットを削除した場合、そのスナップショットを参照する新しいオペレーションを開始することはできません。ただし、復元操作が進行中である場合、その復元操作は完了するまで実行されます。

Amazon Redshift には、作成できる手動スナップショットの合計数を制限するクォータがあります。このクォータは AWS アカウントごと、AWS リージョンごとにあります。デフォルトのクォータは Amazon Redshift でのクォータと制限 に一覧表示されています。

スナップショットストレージ

手動スナップショットにはストレージ料金が発生するので、スナップショットが不要になった場合は削除することが重要です。Amazon Redshift は、それぞれのスナップショット保持期間の終了時に自動スナップショットおよび手動スナップショットを削除します。また、AWS マネジメントコンソール または batch-delete-cluster-snapshots CLI コマンドを使用して、手動スナップショットを削除できます。

手動スナップショット設定を変更して、手動スナップショット保持期間を変更できます。

Amazon Redshift コンソールまたは CLI コマンド describe-storage を使用すると、スナップショットが使用しているストレージ容量に関する情報を取得できます。

スナップショットのテーブルを除く

デフォルトでは、スナップショットにすべてのユーザー定義の永続テーブルが含まれます。ステージングテーブルなど、テーブルをバックアップする必要のない場合は、スナップショットの作成やスナップショットからの復元にかかる時間を大幅に短縮できます。さらに、バックアップしないテーブルを使用して、Amazon S3 のストレージ領域を節約することができます。バックアップしないテーブルを作成するには、テーブルの作成時に BACKUP NO のパラメータを含めてください。詳細については、Amazon Redshift データベースデベロッパーガイドの CREATE TABLE および CREATE TABLE AS を参照してください。

> **Note**
>
> RG または RA3 でプロビジョニングされたクラスターと Amazon Redshift Serverless ワークグループでは、バックアップ用ではないテーブルはサポートされていません。RG または RA3 クラスターおよびサーバーレスワークグループでバックアップしないとマークされたテーブルは、スナップショットの作成中に常にバックアップされ、スナップショットから復元するときに復元される永続テーブルとして扱われます。バックアップ用ではないテーブルのスナップショットコストを回避するには、スナップショットを作成する前にそれらを切り捨てます。

手動スナップショットの作成

クラスターの手動スナップショットは、以下に示すスナップショットリストから作成できます。あるいは、クラスター設定ペインでクラスターのスナップショットを取得することもできます。詳細については、「Amazon Redshift スナップショットとバックアップ」を参照してください。

> **Note**
>
> RG または RA3 でプロビジョニングされたクラスターと Amazon Redshift Serverless ワークグループでは、バックアップ用ではないテーブルはサポートされていません。RG または RA3 クラスターおよびサーバーレスワークグループでバックアップしないとマークされたテーブルは、スナップショットの作成中に常にバックアップされ、スナップショットから復元するときに復元される永続テーブルとして扱われます。バックアップ用ではないテーブルのスナップショットコストを回避するには、スナップショットを作成する前にそれらを切り捨てます。

スナップショットを手動で作成するには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Clusters] (クラスター)、[Snapshots] (スナップショット)、次に [Create snapshot] (スナップショットの作成) を選択します。手動でスナップショットを作成するスナップショットページが表示されます。
3. スケジュール定義のプロパティを入力してから、[スナップショットの作成] を選択します。スナップショットが使用できるようになるまではしばらくかかります。

スナップショットスケジュールの作成

Amazon Redshift は、データの自動的な増分スナップショットを定期的に取得し、Amazon S3 に保存します。さらに、いつでも好きなときにデータの手動スナップショットを取得することもできます。

Amazon Redshift コンソールでのスナップショットタスクはすべてスナップショットリストから開始します。時間範囲、スナップショットのタイプ、およびスナップショットに関連付けられたクラスターを使用して、リストをフィルタリングすることができます。さらに、日付、サイズ、スナップショットのタイプでリストを並べ替えることができます。スナップショットを操作する場合に使用できるオプションは、選択するスナップショットのタイプに応じて異なる場合があります。

スナップショットを作成するタイミングを正確に制御するために、スナップショットスケジュールを作成し、それを 1 つ以上のクラスターにアタッチすることができます。クラスターを作成するとき、またはクラスターを変更することによってスケジュールをアタッチできます。詳細については、「自動スナップショットのスケジュール」を参照してください。

スナップショットスケジュールを作成するには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Clusters] (クラスター)、[Snapshots] (スナップショット、次に [Snapshot schedules] (スナップショットスケジュール) タブを選択します。スナップショットスケジュールが表示されます。
3. [Add schedule (スケジュールの追加)] を選択して、スケジュールを追加します。
4. スケジュール定義のプロパティを入力してから、[Add schedule (スケジュールの追加)] を選択します。
5. 表示されたページでは、新しいスナップショットスケジュールにクラスターをアタッチし、[OK] を選択できます。

スナップショットの共有

1 つの既存の手動スナップショットについては、そのスナップショットへのアクセスを許可することにより、他の AWS 顧客アカウントのユーザーと共有することができます。各スナップショットは最大 20 個、各 AWS Key Management Service (AWS KMS) キーは最大 100 個まで許可できます。つまり、1 つの KMS キーで暗号化された 10 個のスナップショットがある場合、10 個の AWS アカウントに各スナップショットを復元することを許可できます。または、最大 100 個のアカウントのその他の組み合わせや、スナップショットごとに 20 個のアカウントを超えないその他の組み合わせを

許可できます。アクセス権限が付与されたいずれかのアカウントのユーザーとしてログインされた担当者は、スナップショットを表示することも、当該アカウントでスナップショットを復元して新しい Amazon Redshift クラスターを作成することもできます。例えば、実稼働用およびテスト用に個別の AWS 顧客アカウントを使用する場合、ユーザーは本番用アカウントを使用してログオンし、テスト用アカウントのユーザーとスナップショットを共有することができます。テスト用アカウントのユーザーとしてログオンされた担当者は、テストまたは診断作業のためのテスト用アカウントによって所有される新しいクラスターを作成するためにスナップショットを復元することができます。

手動スナップショットは、それが作成された AWS 顧客アカウントによって永続的に所有されます。スナップショットを所有するアカウントのユーザーのみが、スナップショットへのアクセスを他のアカウントに許可したり、アクセス許可を取り消したりすることができます。アクセス権限が付与されたアカウントのユーザーは、そのアカウントと共有されているスナップショットの表示または復元が行えるだけで、共有されているスナップショットのコピーや削除を行うことはできません。アクセス許可はスナップショットの所有者がそれを取り消すまで有効です。アクセス許可が取り消されると、前にアクセス権限を付与されたユーザーはスナップショットの可視性を失い、スナップショットを参照する新しいアクションを起動できなくなります。アクセス権限が取り消される際、アカウントがスナップショットを復元するプロセスの途中にあった場合、復元は完了するまで実行されます。スナップショットにアクティブ認可がある限り、そのスナップショットを削除することはできません。まず、すべてのアクセス許可を取り消す必要があります。

AWS 顧客アカウントには、該当するアカウントによって所有されるスナップショットへのアクセスが常に許可されます。所有者アカウントへのアクセスを許可する試みまたは取り消す試みを行うと、エラーが発生します。非アクティブ AWS 顧客アカウントによって所有されているスナップショットを復元または表示することはできません。

AWS カスタマーアカウントへのアクセスを許可した場合、そのアカウントのユーザーがスナップショットに対してアクションを実行するには、それを許可するポリシーを持つロールを引き受ける必要があります。

- スナップショット所有者アカウントのユーザーがスナップショットへのアクセスを許可および取り消しできるのは、当該スナップショットを含むリソース仕様でそのようなアクションの実行を許可する IAM ポリシーを持つロールを引き受けた場合に限られます。例えば、次のポリシーでは、AWS アカウント `012345678912` のユーザーまたはロールは、`my-snapshot20130829` という名前のスナップショットへのアクセスを他のアカウントに許可できます。

  JSON

```
{
  "Version":"2012-10-17",
```

```
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
          "redshift:AuthorizeSnapshotAccess",
          "redshift:RevokeSnapshotAccess"
          ],
      "Resource":[
           "arn:aws:redshift:us-east-1:012345678912:snapshot:*/my-
snapshot20130829"
          ]
    }
  ]
}
```

- スナップショットを共有している AWS アカウントのユーザーがそのスナップショットに対してアクションを実行するには、そのようなアクションを実行するためのアクセス許可が必要です。そのためには、ポリシーをロールに割り当て、そのロールを引き受けます。
  - スナップショットを一覧表示するか、または表示するためには、前述ユーザーは DescribeClusterSnapshots アクションを許可する IAM ポリシーを持っている必要があります。コードの例を以下に示します。

    JSON

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
          "redshift:DescribeClusterSnapshots"
          ],
      "Resource":[
           "*"
          ]
    }
  ]
}
```

  - ユーザーがスナップショットを復元するには、RestoreFromClusterSnapshot アクションを許可する IAM ポリシーを持つロールを引き受ける必要があり、その IAM ポリシーにはユーザーが作成するクラスターとスナップショットの両方に対応するリソース要素が含まれてい

る必要があります。例えば、アカウント 012345678912 のユーザーがスナップショット my-snapshot20130829 をアカウント 219876543210 と共有している場合、スナップショットを復元してクラスターを作成するには、アカウント 219876543210 のユーザーが次のようなポリシーを持つロールを引き受ける必要があります。

JSON

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
          "redshift:RestoreFromClusterSnapshot"
          ],
      "Resource":[
           "arn:aws:redshift:us-east-1:012345678912:snapshot:*/my-
snapshot20130829",
           "arn:aws:redshift:us-east-1:219876543210:cluster:from-another-
account"
          ]
    }
  ]
}
```

- スナップショットへのアクセスが AWS アカウントから取り消された後、そのアカウントのユーザーはスナップショットにアクセスできなくなります。これは、これらのアカウント内に、以前に共有したスナップショットリソースへのアクションを許可する IAM ポリシーがある場合でも同様です。

コンソールを使用したクラススナップショットの共有

コンソールでは、自分が所有する手動スナップショットへのアクセスを他のユーザーに許可することができます。そのアクセス許可については後で不要になった場合に取り消すことができます。

別のアカウントとスナップショットを共有するには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Clusters] (クラスター)、[Snapshots] (スナップショット)、次に共有する手動スナップショットを選択します。

3. [アクション] に [Manual snapshot settings (手動スナップショット設定)] を選択して、手動スナップショットのプロパティを表示します。
4. [アクセスの管理] セクションで共有するアカウント (複数可) を入力してから、[保存] を選択します。

暗号化されたスナップショットの共有に関するセキュリティ上の考慮事項

Redshift は、暗号化されたスナップショットに対するアクセス権を提供するときに、スナップショットの作成に使用された AWS KMS カスタマーマネージドキーがアカウントまたは復元を実行するアカウントと共有されることを必要とします。キーが共有されていない場合にスナップショットの復元を試みると、アクセス拒否エラーが発生します。受信側のアカウントには、共有スナップショットを復元するための追加のアクセス許可は必要ありません。スナップショットアクセスを認可し、キーを共有する場合、ID 認可アクセスにはスナップショットの暗号化に使用されたキーに対する `kms:DescribeKey` のアクセス許可が必要です。この権限については、「AWS KMS アクセス権限」で詳しく説明します。詳細については、Amazon Redshift API リファレンスドキュメントの「DescribeKey」を参照してください。

カスタマーマネージドキーポリシーは、プログラム的に、または AWS Key Management Service コンソールで更新できます。

> **Note**
>
> デフォルトの KMS キーを使用している場合は、スナップショットを共有するために AWS KMS でアクションを実行したり、何かを変更したりする必要はありません。

暗号化されたスナップショットの AWS KMS キーへのアクセスの許可

暗号化されたスナップショットの AWS KMS カスタマーマネージドキーを共有するには、以下の手順を実行してキーポリシーを更新します。

1. キーを共有する AWS アカウントの Amazon リソースネーム (ARN) を KMS キーポリシーの `Principal` として使用して KMS キーポリシーを更新します。
2. `kms:Decrypt` アクションを許可します。

以下のキーポリシー例では、ユーザー 111122223333 が KMS キーの所有者であり、ユーザー 444455556666 がキーを共有するアカウントです。このキーポリシーは、ユーザー 444455556666

のルート AWS アカウント ID の ARN をポリシーの `Principal` として含め、`kms:Decrypt` アクションを許可することによって、サンプル KMS キーへのアクセス権を AWS アカウントに付与します。

JSON

```
{
    "Id": "key-policy-1",
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Allow use of the key",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:user/KeyUser",
                    "arn:aws:iam::444455556666:root"
                ]
            },
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "*"
        }
    ]
}
```

カスタマーマネージド KMS キーに対するアクセス権を付与した後は、暗号化されたスナップショットを復元するアカウントが AWS Identity and Access Management (IAM) ロールまたはユーザーを作成する必要があります (まだ作成していない場合)。さらに、その AWS アカウントは、KMS キーを使用して暗号化されたデータベーススナップショットを復元することを許可する IAM ポリシーをその IAM ロールまたはユーザーにアタッチする必要もあります。

AWS KMS キーに対するアクセス権の付与に関する詳細については、「AWS Key Management Service デベロッパーガイド」の「他のアカウントのユーザーに KMS キーの使用を許可する」を参照してください。

キーポリシーの概要については、「Amazon Redshift が AWS KMS を使用する方法」を参照してください。

自動スナップショットのコピー

自動スナップショットを無効にした場合、またはクラスターを削除した場合、自動スナップショットはその保持期間が過ぎると自動的に削除されます。自動スナップショットをもっと長い期間保持したい場合は、それを手動スナップショットにコピーします。

自動スナップショットをコピーする方法

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Clusters] (クラスター)、[Snapshots] (スナップショット)、次にコピーするスナップショットを選択します。
3. [アクション] で [自動スナップショットのコピー] を選択してスナップショットをコピーします。
4. 新しいスナップショットのプロパティを更新し、次に [コピー] を選択します。

スナップショットを別の AWS リージョンにコピーする

クラスターのスナップショット (自動または手動) を自動的に別の AWS リージョンにコピーするように Amazon Redshift を設定できます。スナップショットがクラスターのプライマリ AWS リージョンで作成されると、そのスナップショットはセカンダリ AWS リージョンにコピーされます。この 2 つの AWS リージョンは、それぞれソース AWS リージョンとコピー先 AWS リージョンとして知られています。スナップショットのコピーを別の AWS リージョンに保存しておくと、プライマリ AWS リージョンに何かあった場合、最新のデータからクラスターを復元できます。一度に 1 つのコピー先 AWS リージョンにのみスナップショットをコピーするようにクラスターを設定できます。Amazon Redshift リージョンのリストについては、Amazon Web Services 全般のリファレンスの「リージョンとエンドポイント」を参照してください。

Amazon Redshift でスナップショットを別の AWS リージョンに自動的にコピーできるようにする場合、スナップショットのコピー先 AWS リージョンを指定します。自動スナップショットの場合、コピー先 AWS リージョンにスナップショットを保持する保持期間も設定できます。自動スナップショットがコピー先 AWS リージョンにコピーされ、保持期間に達すると、そのスナップショットはコピー先 AWS リージョンから削除されます。これにより、スナップショットの使用率が低く保たれます。自動スナップショットがコピー先 AWS リージョンで保持される時間を短くまたは長くするには、この保持期間を変更します。

コピー先 AWS リージョンにコピーされる自動スナップショットに対して設定する期間は、ソース AWS リージョンの自動スナップショットの保持期間とは異なります。コピーされたスナップショッ

トのデフォルトの保持期間は 7 日です。その 7 日間は、自動スナップショットにのみ適用されます。ソースおよびコピー先の AWS リージョン両方で、手動スナップショットは、スナップショット保持期間の終了時に削除されます。または手動でも削除できます。

クラスターの自動スナップショットコピーはいつでも無効にできます。この機能を無効にすると、スナップショットはソース AWS リージョンからコピー先 AWS リージョンにコピーされなくなります。コピー先 AWS リージョンにコピーされた自動スナップショットは、手動スナップショットコピーを作成しない限り、保持期間の制限に達すると削除されます。これらの手動スナップショットおよびコピー先 AWS リージョンからコピーされた手動スナップショットは、手動で削除するまでコピー先 AWS リージョンに保持されます。

スナップショットのコピー先 AWS リージョンを変更するには、まず自動コピー機能を無効にします。次に、新しいコピー先 AWS リージョンを指定して、再度有効にします。

スナップショットがコピー先 AWS リージョンにコピーされると、そのリージョンがアクティブになり、復元目的で利用できるようになります。

AWS KMS で暗号化されたクラスターのスナップショットを別の AWS リージョンにコピーするには、コピー先の AWS リージョンでカスタマー管理のキーを使用する、Amazon Redshift のための許可を作成します。次に、ソース AWS リージョンでスナップショットのコピーを有効にするときにその許可を選択します。スナップショットコピー許可の設定に関する詳細については、「別の AWS リージョンに AWS KMS 暗号化スナップショットをコピーする」を参照してください。

スナップショットからのクラスターの復元

スナップショットには、クラスターで実行されているデータベースのデータが含まれます。また、ノード数、ノードタイプ、管理者ユーザー名など、クラスターに関する情報も含まれています。スナップショットからクラスターを復元する場合、Amazon Redshift がクラスター情報を使用して新しいクラスターを作成します。次に、スナップショットデータからすべてのデータベースを復元します。

> **Note**
>
> RG または RA3 でプロビジョニングされたクラスターと Amazon Redshift Serverless ワークグループでは、バックアップ用ではないテーブルはサポートされていません。RG または RA3 クラスターおよびサーバーレスワークグループでバックアップしないとマークされたテーブルは、スナップショットの作成中に常にバックアップされ、スナップショットから復元するときに復元される永続テーブルとして扱われます。

元のスナップショットから作成された新しいクラスターの場合、ノードタイプやノード数などの構成を選択できます。クラスターは同じ AWS リージョンに復元されます。アベイラビリティーゾーンに関しては、リクエスト時にアベイラビリティーゾーンを指定しない限り、システムによりランダムに選択されます。クラスターをスナップショットから復元する場合、新しいクラスターの互換性のあるメンテナンストラックをオプションで選択できます。

> **Note**
>
> 異なる構成のクラスターにスナップショットを復元する場合、スナップショットはクラスターバージョン 1.0.10013 以降のクラスターで作成されている必要があります。

リストアの進行中は、通常、イベントは次の順序で生成されます。

1. RESTORE_STARTED – REDSHIFT-EVENT-2008 は、復元プロセスの開始時に送信されます。
2. RESTORE_SUCCEEDED – 新しいクラスターが作成されたときに、REDSHIFT-EVENT-3003 が送信されます。

   クラスターはクエリに使用できます。
3. DATA_TRANSFER_COMPLETED – データ転送が完了すると、REDSHIFT-EVENT-3537 が送信されます。

> **Note**
>
> RG および RA3 クラスターは、RESTORE_STARTED イベントと RESTORE_SUCCEEDED イベントのみを発行します。RG および RA3 ノードタイプは Amazon Redshift マネージドストレージにデータを格納するため、RESTORE が成功した後に明示的にデータ転送を行う必要はありません。RG および RA3 ノードでは、通常のクエリ処理の一環として、RG および RA3 ノードと Amazon Redshift マネージドストレージ間でデータが継続的に転送されます。RG および RA3 ノードは、ホットデータをローカルにキャッシュし、クエリ頻度の低いブロックを Amazon Redshift 管理ストレージに自動的に保持します。

DescribeClusters API アクションを呼び出すか、AWS マネジメントコンソールでクラスターの詳細を表示することにより、復元の進行状況をモニタリングできます。これにより、進行中の復元について、スナップショットデータのサイズ、転送速度、経過時間、および推定残り時間などの情報が表示されます。これらのメトリクスの説明については、「RestoreStatus」を参照してください。

スナップショットを使用して、アクティブなクラスターを前の状態に切り替えることはできません。

> **Note**
>
> 新しいクラスターにスナップショットを復元する場合、別の値を指定しない限り、デフォルトのセキュリティグループおよびパラメータグループが使用されます。

異なる構成のクラスターにスナップショットを復元する理由には、次のようなものがあります。

- クラスターがより小さなノードタイプで構成されており、それをより少ないノードでより大きなノードタイプに統合する場合。
- ワークロードを監視し、より多くの CPU とストレージを備えたノードタイプに移行する必要があると判断した場合。
- 異なるノードタイプでテストワークロードのパフォーマンスを測定する場合。

復元には次の制約があります。

- 新しいノード設定では、既存のデータに対して十分なストレージが必要です。ノードを追加するときでも、データが再分配される方法のために、新しい設定に十分なストレージがない場合があります。
- 復元操作は、新しいクラスターのクラスターバージョンと互換性のあるクラスターバージョンでスナップショットが作成されたかどうかをチェックします。新しいクラスターのバージョンレベルが早すぎる場合、復元操作は失敗し、エラーメッセージに詳細情報が報告されます。
- 復元可能な設定 (ノードの数とノードの種類) は、元のクラスター内のノード数と、新しいクラスターのターゲットノードタイプによって決まります。利用可能な設定を確認するには、`action-type restore-cluster` で `describe-node-configuration-options` AWS CLI コマンド、または Amazon Redshift コンソールを使用できます。Amazon Redshift コンソールを使用した復元の詳細については、スナップショットからのクラスターの復元 を参照してください。

次の手順で、多数のノードを持つクラスターを取得し、AWS CLI を使用して少数のノードを持つより大きなノードタイプに統合します。この例では、24 ノードのソースクラスターから始めます。この場合、このクラスターのスナップショットを既に作成しており、それをより大きなノードタイプに復元するとします。

1. 次のコマンドを実行して、24 ノードの クラスターの詳細を取得します。

```
aws redshift describe-clusters --region eu-west-1 --cluster-identifier
 mycluster-123456789012
```

2. 次のコマンドを実行して、スナップショットの詳細を取得します。

```
aws redshift describe-cluster-snapshots --region eu-west-1 --snapshot-identifier
 mycluster-snapshot
```

3. 次のコマンドを実行して、このスナップショットで使用可能なオプションの詳細を表示します。

```
aws redshift describe-node-configuration-options --snapshot-identifier mycluster-
snapshot --region eu-west-1 --action-type restore-cluster
```

このコマンドは、各オプションの推奨ノードタイプ、ノード数、およびディスク使用率を含むオプションリストを返します。この例では、前述のコマンドは次の可能なノード構成をリストします。4 ノードのクラスターに復元することを選択します。

```
{
    "NodeConfigurationOptionList": [
        {
            "NodeType": "ra3.16xlarge",
            "NumberOfNodes": 2,
            "EstimatedDiskUtilizationPercent": 0.1
        },
        {
            "NodeType": "ra3.16xlarge",
            "NumberOfNodes": 4,
            "EstimatedDiskUtilizationPercent": 0.05
        },
        {
            "NodeType": "ra3.4xlarge",
            "NumberOfNodes": 8,
            "EstimatedDiskUtilizationPercent": 0.03
        },
        {
            "NodeType": "ra3.4xlarge",
            "NumberOfNodes": 10,
            "EstimatedDiskUtilizationPercent": 0.02
        },
        {
            "NodeType": "ra3.4xlarge",
```

```
            "NumberOfNodes": 12,
            "EstimatedDiskUtilizationPercent": 0.02
        },
        {
            "NodeType": "ra3.xlplus",
            "NumberOfNodes": 8,
            "EstimatedDiskUtilizationPercent": 0.11
        },
        {
            "NodeType": "ra3.xlplus",
            "NumberOfNodes": 16,
            "EstimatedDiskUtilizationPercent": 0.06
        }
    ]
}
```

4. 次のコマンドを実行して、選択したクラスター構成にスナップショットを復元します。このクラスターが復元された後、ソースクラスターと同じコンテンツがありますが、データは 4 つの ra3.16xlarge ノードに統合されています。

```
aws redshift restore-from-cluster-snapshot --region eu-west-1 --snapshot-identifier
 mycluster-snapshot --cluster-identifier mycluster-123456789012-x --node-type
 ra3.16xlarge --number-of-nodes 4
```

リザーブドノード (RA3 リザーブドノードなど) がある場合は、RG リザーブドノードにアップグレードできます。このアップグレードは、スナップショットからの復元を実行する場合、あるいは伸縮自在なリサイズを実行する場合に利用できます。コンソールを使用している場合は、このプロセスに関するガイドが提供されます。RG ノードへのアップグレードの詳細については、「RG または RA3 ノードタイプへのアップグレード」を参照してください。

コンソールでスナップショットからクラスターを復元する方法

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [Clusters] (クラスター)、[Snapshots] (スナップショット)、次に復元するスナップショットを選択します。
3. [スナップショットからの復元] を選択して、スナップショット情報を使って作成される新しいクラスターの [クラスターの設定] と [クラスター詳細] 値を表示します。

4. 新しいクラスターのプロパティを更新してから、[スナップショットからのクラスターの復元] を選択します。

クラススナップショットを復元すると、復元されたデータウェアハウスはスナップショット作成時に使用されていたのと同じカスタム AWS KMS キーで暗号化されます。スナップショットにカスタム KMS キーがない場合、Amazon Redshift のバックアップ暗号化ロジックは次の要因によって決まります。

- スナップショットを復元する Amazon Redshift データウェアハウスのタイプ。
- スナップショット作成時のクラスターの暗号化タイプ。

クラススナップショットから復元した後のデータウェアハウスの暗号化方法については、次の表を参照してください。

送信先タイプ	スナップショットの暗号化タイプ	復元先の暗号化タイプ
プロビジョニングされたクラスター	AWS マネージドキー で暗号化	AWS マネージドキー で暗号化
プロビジョニングされたクラスター	AWS 所有のキー で暗号化	AWS 所有のキー で暗号化
サーバーレス名前空間	AWS マネージドキー で暗号化	AWS 所有のキー で暗号化
サーバーレス名前空間	AWS 所有のキー で暗号化	AWS 所有のキー で暗号化

スナップショットの撮影時に AWS Secrets Manager が管理するクラスターの管理者パスワードを取得した場合は、引き続き AWS Secrets Manager を使用して管理者パスワードを管理する必要があります。クラスター詳細ページでクラスターの管理者認証情報を更新することで、クラスターの復元後にシークレットの使用をオプトアウトできます。

リザーブドノードがある場合は、RG リザーブドノードにアップグレードできます。このアップグレードは、スナップショットからの復元を実行する場合、あるいは伸縮自在なリサイズを実行する場合に利用できます。コンソールを使用している場合は、このプロセスに関するガイドが提供されま

す。RG ノードへのアップグレードの詳細については、「RG または RA3 ノードタイプへのアップグレード」を参照してください。

スナップショットからのテーブルの復元

クラスター全体を復元する代わりに、スナップショットから単一のテーブルを復元できます。スナップショットから単一のテーブルを復元する場合、ソースのスナップショット、データベース、スキーマ、テーブル名、ターゲットのデータベース、スキーマ、および復元されるテーブル用の新しいテーブル名を指定します。

> **Note**
>
> RG または RA3 でプロビジョニングされたクラスターと Amazon Redshift Serverless ワークグループでは、バックアップ用ではないテーブルはサポートされていません。RG または RA3 クラスターおよびサーバーレスワークグループでバックアップしないとマークされたテーブルは、スナップショットの作成中に常にバックアップされ、スナップショットから復元するときに復元される永続テーブルとして扱われます。ただし、バックアップ用ではないテーブルの選択的な復元はサポートされていません。

新しいテーブル名を、既存のテーブルの名前にすることはできません。既存のテーブルを、スナップショットから復元されるテーブルに置き換えるには、スナップショットからテーブルを復元する前に、既存のテーブルの名前を変更するか、削除します。

ターゲットテーブルは、ソーステーブルの列の定義、テーブル属性、および外部キーを除く列の属性を使って作成されます。依存関係による競合を回避するため、ターゲットテーブルはソーステーブルから外部キーを継承しません。ソーステーブルで付与されたビューや許可などの依存関係は、ターゲットテーブルに適用されません。

ソーステーブルの所有者が存在する場合、そのデータベースユーザーが復元したテーブルの所有者となるのは、指定したデータベースやスキーマの関係の所有者となる十分なアクセス許可を持っている場合のみです。それ以外の場合には、クラスターの起動時に作成した管理者ユーザーが、復元されたテーブルを所有します。

復元されたテーブルは、バックアップが作成された時の状態に戻されます。これには、Amazon Redshift の直列化分離への準拠により定義されるトランザクションの可視性のルールが含まれます。つまり、バックアップ後に開始した実行中のトランザクションにデータがすぐに見えるようになるということです。

スナップショットからのテーブルの復元には、以下の制限があります。

- テーブルは、実行中のアクティブなクラスターのみに復元でき、そのクラスターから作成されたスナップショットのみから復元できます。
- 一度に復元できるのは 1 つのテーブルのみです。
- クラスターのサイズを変更する前に作成されたクラスターのスナップショットからテーブルを復元することはできません。例外として、ノードタイプが変更されていない場合は、伸縮自在にサイズを変更した後にテーブルを復元できます。
- ソーステーブルで付与されたビューや許可などの依存関係は、ターゲットテーブルに適用されません。
- 復元中のテーブルに対して行レベルのセキュリティが有効になっている場合、Amazon Redshift は、行レベルのセキュリティがオンになっているテーブルを復元します。

スナップショットからテーブルを復元するには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Clusters] (クラスター) を選択して、テーブルを復元するクラスターを選択します。
3. [アクション] で、[テーブルの復元] を選択して [テーブルの復元] ページを表示します。
4. どのスナップショット、ソーステーブル、およびターゲットテーブルを使うかに関する情報を入力し、次に [テーブルの復元] を選択します。

Example例: AWS CLI を使用してスナップショットからテーブルを復元する

次の例では、`restore-table-from-cluster-snapshot` AWS CLI コマンドを使用して、`my-source-table` の `sample-database` スキーマから `my-snapshot-id` テーブルを復元します。AWS CLI コマンド `describe-table-restore-status` を使用して、復元操作のステータスを確認できます。例では、新しいテーブルの名前 `mycluster-example` を使用して、`my-new-table` クラスターにスナップショットを復元します。

```
aws redshift restore-table-from-cluster-snapshot --cluster-identifier mycluster-
example
                                                 --new-table-name my-new-table
                                                 --snapshot-identifier my-snapshot-id
```

```
                                                --source-database-name sample-
database
                                                --source-table-name my-source-table
```

スナップショットからサーバーレス名前空間の復元

スナップショットからサーバーレス名前空間を復元すると、名前空間のすべてのデータベースをスナップショット内のデータベースに置き換えます。サーバーレススナップショットの詳細については、「スナップショットと復元ポイント」を参照してください。Amazon Redshift は、プロビジョニングされたクラススナップショットを Amazon Redshift Serverless 名前空間に復元するときに、インターリーブキーを含むテーブルを自動的に複合キーに変換します。ソートキーの詳細については、「ソートキーの使用」を参照してください。

プロビジョニングされたクラスターからサーバーレス名前空間にスナップショットを復元する方法。

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [Clusters] (クラスター) と [Snapshots] (スナップショット) を選択したら、使用するスナップショットを選択します。
3. [Restore from snapshot] (スナップショットから復元) と [Restore to serverless namespace] (サーバーレス名前空間に復元) の順に選択します。
4. 復元先の名前空間を選択します。
5. スナップショットから復元することを確認します。[restore] (復元) を選択します。この操作により、サーバーレス名前空間内のすべてのデータベースが、プロビジョニングされたクラスターのデータで置き換えられます。

暗号化されていないクラスターのクロスリージョンスナップショットのコピーを設定する

クラスターのスナップショットを別の AWS リージョンにコピーするように Amazon Redshift を設定できます。クロスリージョンスナップショットのコピーを設定するには、各クラスターでこのコピー機能を有効にし、スナップショットをコピーする場所と、コピーされた自動または手動のスナップショットをコピー先 AWS リージョンに保持する期間を設定する必要があります。クロスリージョンコピーがクラスターで有効になると、すべての新しい手動および自動スナップショットが、指定された AWS リージョンにコピーされます。コピーされたスナップショットには **`copy:`** というプレフィックスが付きます。

クロスリージョンスナップショットのコピーを設定するには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Clusters] (クラスター) を選択して、スナップショットを移動するクラスターを選択します。
3. [Actions] (アクション) で、[Configure cross-region snapshot] (クロスリージョンスナップショットの設定) を選択します。

   [クロスリージョンの設定] ダイアログボックスが表示されます。
4. を使用する場合[スナップショットのコピー] で、はい。
5. [コピー先 AWS リージョン] で、スナップショットをコピーする AWS リージョンを選択します。
6. [自動スナップショットの保持期間 (日数)] で、自動スナップショットが削除される前にコピー先 AWS リージョンに保持される日数を選択します。
7. [手動スナップショットの保持期間] で、手動スナップショットが削除される前にコピー先 AWS リージョンに保持される日数を選択します。を選択すると、カスタム値の場合、保持期間は 1 ~ 3653 日間でなければなりません。
8. [保存] を選択します。

AWS KMS 暗号化クラスターのクロスリージョンスナップショットのコピーを設定する

Amazon Redshift クラスターを起動する際に、コピー先の AWS リージョンのアカウントでルートキーのスナップショットコピー権限を設定できます。権限を設定しない場合、コピー先リージョンのスナップショットはデフォルトの AWS が所有するキーを使用して暗号化されます。こうすることにより、Amazon Redshift がコピー先 AWS リージョンで暗号化オペレーションを実行できるようになります。

次の手順は、AWS KMS 暗号化クラスターのクロスリージョンスナップショットコピーを有効化するプロセスを示しています。Amazon Redshift およびスナップショットコピー権限での暗号化の詳細については、別の AWS リージョンに AWS KMS 暗号化スナップショットをコピーする を参照してください。

AWS KMS で暗号化されたクラスターのクロスリージョンスナップショットを設定するには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Clusters] (クラスター) を選択して、スナップショットを移動するクラスターを選択します。
3. [Actions] (アクション) で、[Configure cross-region snapshot] (クロスリージョンスナップショットの設定) を選択します。

   [クロスリージョンの設定] ダイアログボックスが表示されます。
4. を使用する場合[スナップショットのコピー] で、はい。
5. [コピー先 AWS リージョン] で、スナップショットをコピーする AWS リージョンを選択します。
6. [自動スナップショットの保持期間 (日数)] で、自動スナップショットが削除される前にコピー先 AWS リージョンに保持される日数を選択します。
7. [手動スナップショットの保持期間] で、手動スナップショットが削除される前にコピー先 AWS リージョンに保持される日数を選択します。を選択すると、カスタム値の場合、保持期間は 1 ~ 3653 日間でなければなりません。
8. [保存] を選択します。

手動スナップショットの保持期間を変更する

スナップショット設定を変更して、手動スナップショット保持期間を変更できます。

手動スナップショット保持期間を変更するには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Clusters] (クラスター)、[Snapshots] (スナップショット)、次に変更する手動スナップショットを選択します。
3. [アクション] に [Manual snapshot settings (手動スナップショット設定)] を選択して、手動スナップショットのプロパティを表示します。
4. スケジュール定義の改訂済みプロパティを入力してから、[保存] を選択します。

クロスリージョンスナップショットのコピーの保持期間を修正する

クロスリージョンスナップショットのコピーを設定した後で、設定を変更できます。新しい日数を選択し、変更を保存することにより、保持期間を簡単に変更できます。

> **Warning**
>
> クロスリージョンスナップショットのコピーを設定した後、コピー先 AWS リージョンを変更することはできません。
> 異なる AWS リージョンにスナップショットをコピーする場合、最初にクロスリージョンスナップショットのコピーを無効にします。その後、新しいコピー先 AWS リージョンと保持期間で再有効にします。コピーされた自動スナップショットは、クロスリージョンスナップショットコピーを無効化した後削除されます。そのため、クロスリージョンスナップショットコピーを無効化する前に、保持して手動スナップショットにコピーしたいものがないかどうか決定する必要があります。

クロスリージョンスナップショットを変更するには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Clusters] (クラスター) を選択して、スナップショットを変更するクラスターを選択します。
3. [アクション] に、[Configure cross-region snapshot (クロスリージョンスナップショットの設定)] を選択してスナップショットのプロパティを表示します。
4. スケジュール定義の改訂済みプロパティを入力してから、[保存] を選択します。

手動スナップショットの削除

スナップショットリストで 1 つ以上のスナップショットを選択して手動スナップショットを削除できます。

手動スナップショットを削除する方法

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Clusters] (クラスター)、[Snapshots] (スナップショット)、次に削除するスナップショットを選択します。

3. [アクション] で [スナップショットの削除] を選択してスナップショットを削除します。
4. リストされたスナップショットの削除を確認してから、[削除] を選択します。

AWS Glue Data Catalog にクラスターを登録する

クラスター全体を AWS Glue Data Catalog に登録して、AWS Glue によって管理されるカタログを作成できます。これらのカタログには、Apache Iceberg REST API に対応した任意の SQL エンジンでアクセスできます。Amazon Redshift から Apache Iceberg 互換のカタログを作成する方法については、「Amazon Redshift データベースデベロッパーガイド」の「Apache Iceberg の Amazon Redshift との互換性」を参照してください。

クラスターを AWS Glue Data Catalog に登録するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [クラスター] を選択します。現在の AWS リージョン にあるアカウントのクラスターがリストされています。各クラスターのプロパティのサブセットが、リストの列に表示されます。クラスターがない場合、[クラスターを作成] を選択して作成します。
3. 登録するクラスターの名前を選択します。
4. [アクション] から [AWS Glue Data Catalog に登録] を選択します。[AWS Glue Data Catalog に登録] ポップアップボックスが表示されます。
5. クラスターの登録先の AWS アカウント ID を [登録先アカウント ID] に入力します。このアカウント ID が、AWS Glue Data Catalog でカタログを保持します。
6. [名前空間の登録名] に名前を入力します。これが、Data Catalog でクラスターの名前になります。
7. [登録] を選択します。AWS Lake Formation コンソールに自動的に移動します。
8. AWS Lake Formation でカタログ作成のプロセスに従います。カタログ作成の詳細については、「AWS Lake Formation デベロッパーガイド」の「Amazon Redshift データ AWS Glue Data Catalog への取り込み」を参照してください。

マルチ AZ デプロイ

Amazon Redshift は、プロビジョニングされた RG または RA3 クラスター用に複数のアベイラビリティーゾーン (マルチ AZ) 配置をサポートしています。マルチ AZ 配置を使用することで、アベイラ

ビリティーゾーンで予期しないイベントが発生した障害シナリオでも Amazon Redshift データウェアハウスを引き続き運用できます。マルチ AZ 配置は、2 つのアベイラビリティーゾーン (AZ) にコンピューティングリソースをデプロイし、これらのコンピューティングリソースは 1 つのエンドポイントからアクセスできます。アベイラビリティーゾーン全体で障害が発生しても、2 つ目のアベイラビリティーゾーンの残りのコンピューティングリソースは引き続きワークロードの処理に使用できます。Amazon Redshift は、マルチ AZ データウェアハウスを実行する場合、RG または RA3 に対して同じ時間単位のコンピューティング料金を請求します。すべてのアベイラビリティーゾーンおよび AWS リージョン で共有されるため、ストレージコストは変わりません。

現在、Amazon Redshift はゼロ目標復旧時点 (RPO) をサポートしています。これにより、障害発生時にデータを現在かつ最新の状態に保つことができます。マルチ AZ 配置により、Amazon Redshift は既存の復旧機能をさらに強化し、目標復旧時間 (RTO) を短縮します。これが可能なのは、マルチ AZ 配置では障害や災害からより迅速に復旧できるため、Amazon Redshift サービスレベルアグリーメント (SLA) が、シングル AZ データウェアハウスの場合の 99.9% と比較して 99.99% に引き上げられるためです。

マルチ AZ 配置のセットアップ

マルチ AZ 配置をセットアップするには、[マルチ AZ] オプションを選択し、各アベイラビリティーゾーンにプロビジョニングするコンピューティングノードの数を指定します。Amazon Redshift は 2 つのアベイラビリティーゾーンに同等のコンピューティングリソースを自動的にデプロイし、通常の運用中はすべてのコンピューティングリソースを読み取りと書き込みの両方の処理にいつでも利用でき、セカンダリクラスターの余分なコンピューティングリソースを使用することで、パフォーマンスの向上に役立てることができます。そのため、マルチ AZ 配置は 1 つのエンドポイントを持つ単一のデータウェアハウスとして機能し、災害発生時にアプリケーションを変更する必要がなくなります。マルチ AZ 配置では、1 つのアベイラビリティーゾーンのみにあるコンピューティングリソースを使用して個別のクエリを処理しますが、一部の同時クエリの処理を両方のアベイラビリティーゾーンに自動的に分散することができ、これにより同時実行性の高いワークロードの全体的なスループットが向上する場合もありますが、常に向上するとは限りません。マルチ AZ の主な機能に留意することが重要です。それは、予期しないダウンタイムや停止イベントから迅速に復旧できるように、クラスターに高可用性ソリューションを提供することです。

分析ワークロードのパフォーマンスを高めることが主な動機である場合は、マルチ AZ 以外の次のようなソリューションをお勧めします。

- より大きいノードインスタンスタイプへのサイズ変更 (クラスターのサイズ変更)
- クラスターでの同時実行スケーリングの有効化 (同時実行スケーリング)
- データ共有アーキテクチャを使用してワークロードを分離する (データ共有の概要)

既存のシングル AZ データウェアハウスをマルチ AZ データウェアハウスに変換したり、その逆を行うこともできます。2 つ目のアベイラビリティーゾーンに追加のコンピューティングリソースがプロビジョニングされる点以外は、すべて変わりません。既存のシングル AZ クラスターからマルチ AZ クラスターに移行する場合、単一クエリのパフォーマンスを維持しやすくするために、必要なクラスターノードの数を 2 倍にする必要がある場合があります。

アベイラビリティーゾーンで障害が発生した場合、Amazon Redshift は残りのアベイラビリティーゾーンのリソースを自動的に使用して運用を継続します。ただし、ユーザー接続については失われる可能性があるため、再確立する必要があります。また、障害が発生したアベイラビリティーゾーンで実行されていたクエリは失敗する可能性があり、再試行する必要があります。ただし、クラスターに再接続してクエリをすぐに再スケジュールできます。そうすることで、Amazon Redshift は残りのアベイラビリティーゾーンでクエリを処理します。障害発生時または障害発生後に発行されたクエリでは、マルチ AZ データウェアハウスの復旧中に実行時の遅延が発生する可能性があります。

> **Note**
>
> パフォーマンスと可用性を向上させるには、マルチ AZ クラスターで SNAPSHOT ISOLATION を使用することをお勧めします。詳細については、「データベースの作成」を参照してください。

制限事項

マルチ AZ データウェアハウスは、マルチ AZ データウェアハウスに適用される以下の制限を除いて、シングル AZ データウェアハウスと同じ機能を備えています。

- 暗号化されていないマルチ AZ データウェアハウスを作成することはできません。マルチ AZ データウェアハウスを新規作成する場合、シングル AZ データウェアハウスをマルチ AZ データウェアハウスに変換する場合、またはシングル AZ データウェアハウスをマルチ AZ データウェアハウスに変換する場合は、必ず暗号化を追加してください。
- いずれの RG または RA3 インスタンスタイプに対しても、単一ノードのマルチ AZ 配置を作成することはできません。マルチ AZ 配置を作成しているときにノードを 2 つ以上選択します。
- Amazon Redshift は、3 つ未満のアベイラビリティーゾーンをサポートできるサブネット設定をサポートしていません。つまり、設定されたサブネットグループには 3 つ以上のサブネットが必要です。
- マルチ AZ 配置を別のアベイラビリティーゾーンに再配置することはできません。マルチ AZ 配置を使用している場合、再配置は Amazon Redshift によって自動的に決定および実行されます。

- マルチ AZ 配置を一時停止または再開することはできません。
- マルチ AZ 配置は、サポートされている 5431～5455 と 8191～8215 の範囲のポート以外では実行できません。
- STL、SVCS、SVL、SVV、STV ビューはシステムモニタリングビュー (SYS_* ビュー) のみをサポートしているため、マルチ AZ 配置では使用できません。システムモニタリングビュー (SYS_* ビュー) を使用するようにモニタリングクエリを変更してください。
- マルチ AZ が有効になっている既存のクラスターに Elastic IP アドレスをアタッチすることはできません。
- Elastic IP アドレスがアタッチされたクラスターをシングル AZ からマルチ AZ に変換することはできません。
- Amazon Redshift マルチ AZ 配置は以下の AWS リージョン で利用できます。

> **Note**
>
> RG インスタンスは指定したリージョンで利用できます。「AWS リージョンでの RG ノードタイプの可用性」を参照してください。

  - 米国東部 (オハイオ) (us-east-2)
  - 米国東部 (バージニア北部) (us-east-1)
  - 米国西部 (オレゴン) (us-west-2)
  - アフリカ (ケープタウン) (af-south-1)
  - アジアパシフィック (香港) (ap-east-1)
  - アジアパシフィック (台北) (ap-east-2)
  - アジアパシフィック (ハイデラバード) (ap-south-2)
  - アジアパシフィック (ジャカルタ) (ap-southeast-3)
  - アジアパシフィック (マレーシア) (ap-southeast-5)
  - アジアパシフィック (メルボルン) (ap-southeast-4)
  - アジアパシフィック (ムンバイ) (ap-south-1)
  - アジアパシフィック (大阪) (ap-northeast-3)
  - アジアパシフィック (ソウル) (ap-northeast-2)
  - アジアパシフィック (シンガポール) (ap-southeast-1)
  - アジアパシフィック (シドニー) (ap-southeast-2)

- アジアパシフィック (ニュージーランド) (ap-southeast-6)
- アジアパシフィック (タイ) (ap-southeast-7)
- アジアパシフィック (東京) (ap-northeast-1)
- カナダ (中部) (ca-central-1)
- 中国 (北京) (cn-north-1)
- 中国 (寧夏) (cn-northwest-1)
- ヨーロッパ (フランクフルト) (eu-central-1)
- 欧州 (アイルランド) (eu-west-1)
- ヨーロッパ (ロンドン) (eu-west-2)
- 欧州 (ミラノ) (eu-south-1)
- 欧州 (パリ) (eu-west-3)
- 欧州 (スペイン) (eu-south-2)
- 欧州 (ストックホルム) (eu-north-1)
- 欧州 (チューリッヒ) (eu-central-2)
- イスラエル (テルアビブ) (il-central-1)
- メキシコ (中部) (mx-central-1)
- 中東 (バーレーン) (me-south-1)
- 中東 (UAE) (me-central-1)
- 南米 (サンパウロ) (sa-east-1)
- AWS GovCloud (米国東部) (us-gov-east-1)
- AWS GovCloud (米国西部) (us-gov-west-1)

- パブリックアクセス可能なマルチ AZ データウェアハウスは、シングル AZ およびプライベートアクセス可能なマルチ AZ ウェアハウスよりも 1 つ少ない VPC セキュリティグループをサポートします。

新しいクラスターを作成する際のマルチ AZ の設定

Amazon Redshift マルチ AZ は一度に 2 つのアベイラビリティーゾーンをサポートします。Amazon Redshift は、選択されたサブネットグループ設定に基づいてアベイラビリティーゾーンを自動的に選択します。既存のシングルアベイラビリティーゾーンデータウェアハウスをマルチ AZ 配置に変換するか、スナップショットから復元してマルチ AZ データウェアハウスに設定できます。

Amazon Redshift コンソールを使用すると、新しいマルチ AZ 配置を簡単に作成できます。Amazon Redshift コンソールを使用して新しいマルチ AZ 配置を作成するには、データウェアハウスを作成するときにマルチ AZ オプションを選択します。1 つのアベイラビリティーゾーンに必要なコンピューティングノードの数を指定すると、Amazon Redshift は 2 つのアベイラビリティーゾーンのそれぞれにノードをデプロイします。すべてのノードは、通常の運用中に読み取りと書き込みのワークロード処理を実行できます。AWS CLI `create-cluster` コマンドを使用して、`multi-az` パラメーターを使用する新しいマルチ AZ データウェアハウスを作成することもできます。

既存のシングル AZ データウェアハウスをマルチ AZ データウェアハウスに変換できます。Amazon Redshift コンソールまたは `multi-az` パラメータを使用する AWS CLI `modify-cluster` コマンドを使用できます。または、スナップショットから復元してシングル AZ データウェアハウスをマルチ AZ データウェアハウスに設定できます。これには、Amazon Redshift コンソールまたは `multi-az` パラメータを使用する AWS CLI `restore-from-cluster-snapshot` コマンドを使用できます。

マルチ AZ 配置は、Amazon Redshift マネージドストレージ (RMS) を使用する RG または RA3 ノードタイプのみをサポートします。Amazon Redshift は、Amazon S3 を利用した RMS にデータを保存し、Amazon Redshift レベルでデータを複製しなくても、AWS リージョン 内のすべてのアベイラビリティゾーンでアクセスすることが可能です。

新しいクラスターを作成する際にマルチ AZ 配置を設定できます。Amazon Redshift コンソールまたは AWS Command Line Interface を使用します。

コンソールを使用する

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [Provisioned clusters dashboard] (プロビジョニングされたクラスターダッシュボード) を選択し、[Clusters] (クラスター) を選択します。現在の AWS リージョンにあるアカウントのクラスターがリストされています。各クラスターのプロパティのサブセットが、リストの列に表示されます。
3. [クラスターを作成] ボタンを選択して、クラスターの作成ページを開きます。
4. クラスターのプロパティを入力します。クラスターの作成に関する全般情報については、「クラスターの作成」を参照してください。
5. [ノードタイプ] ドロップダウンリストから RG または RA3 ノードタイプの 1 つを選択します。AZ 設定オプションは、RG または RA3 ノードタイプを選択した場合にのみ使用可能になります。
6. [AZ 設定] で、[マルチ AZ] を選択します。

7. [AZ あたりのノード数] に、クラスターのノードを少なくとも 2 つ入力します。
8. サンプルデータをロードするか、独自のデータを導入できます。

   - [Sample data (サンプルデータ) で、[Load sample data] (サンプルデータをロード) をクリックして、サンプルのデータセットを Amazon Redshift クラスターにロードします。Amazon Redshift は、デフォルトの dev データベースと public スキーマに、サンプルのデータセット Tickit をロードします。Amazon Redshift が、サンプルデータセットを Amazon Redshift クラスターに自動的にロードします。これで、クエリエディタ v2 を使用しながらデータに対するクエリを開始できます。
   - Amazon Redshift クラスターに独自のデータを導入するには、[Bringing your own data to Amazon Redshift] (Amazon Redshift への独自のデータの導入) をクリックします。

9. [Additional configurations] (追加設定) までスクロールして [Network and security] (ネットワークとセキュリティ) を展開し、デフォルトのクラスターサブネットグループを受け入れるか、別のサブネットグループを選択していることを確認します。別のクラスターサブネットグループを選択する場合は、選択したサブネットグループに 3 つのアベイラビリティーゾーンがあることを確認します。
10. [Additional configurations] (追加設定)で、[Database configurations] (データベース構成) を展開します。
11. [データベース暗号化] で、デフォルトの AWS が所有するキーの代わりにカスタム AWS KMS キーを使用するには、[暗号化設定をカスタマイズ] をクリックします。
12. [Choose an KMS key] (KMS キーの選択) で、AWS Key Management Service キーを選択するか、ARN を入力できます。または、AWS Key Management Service コンソールで [AWS Key Management Service キーを作成] をクリックすることもできます。KMS キーの作成の詳細については、AWS Key Management Service デベロッパーガイドの キーの作成 を参照してください。
13. [Create cluster] (クラスターの作成) をクリックします。クラスターの作成が成功すると、クラスターの詳細ページで詳細を確認できます。SQL クライアントを使用して、データをロードしたりクエリしたりできます。

の使用AWS Command Line Interface

AWS Command Line Interface を使用してクラスターを作成する際にマルチ AZ をセットアップするには

- AWS CLI から `create-cluster` コマンドと `multi-az` パラメータを次のように使用します。

```
aws redshift create-cluster
    --port 5439
    --master-username master
    --master-user-password #####
    --node-type rg.4xlarge
    --number-of-nodes 2
    --profile maz-test
    --endpoint-url https://redshift.eu-west-1.amazonaws.com
    --region eu-west-1
    --cluster-identifier test-maz
    --multi-az
    --maintenance-track-name CURRENT
    --encrypted
```

スナップショットから復元されたデータウェアハウスのマルチ AZ の設定

スナップショットから復元して新しいマルチ AZ クラスターを作成するには、次の手順を実行します。

コンソールを使用する

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [Clusters] (クラスター) と [Snapshots] (スナップショット) を選択したら、使用するスナップショットを選択します。
3. [Restore snapshot] (スナップショットの復元)、[Restore to a provisioned cluster] (プロビジョニングされたクラスターへの復元) を選択します。
4. クラスターのプロパティを入力します。クラスターの作成に関する全般情報については、「クラスターの作成」を参照してください。
5. [ノードタイプ] ドロップダウンリストから RG または RA3 ノードタイプの 1 つを選択します。AZ 設定オプションは、RG または RA3 ノードタイプを選択した場合にのみ使用可能になります。
6. [AZ 設定] で、[マルチ AZ] を選択します。
7. [AZ あたりのノード数] に、クラスターのノードを少なくとも 2 つ入力します。
8. サンプルデータをロードするか、独自のデータを導入できます。

- [Sample data (サンプルデータ) で、[Load sample data] (サンプルデータをロード) をクリックして、サンプルのデータセットを Amazon Redshift クラスターにロードします。Amazon Redshift は、デフォルトの dev データベースと public スキーマに、サンプルのデータセット Tickit をロードします。Amazon Redshift が、サンプルデータセットを Amazon Redshift クラスターに自動的にロードします。これで、クエリエディタ v2 を使用しながらデータに対するクエリを開始できます。
- Amazon Redshift クラスターに独自のデータを持ち込むには、「Amazon S3 から Amazon Redshift にデータをロードする」の手順に従います。

9. [Additional configurations] (追加設定) までスクロールして [Network and security] (ネットワークとセキュリティ) を展開し、デフォルトのクラスターサブネットグループを受け入れるか、別のサブネットグループを選択していることを確認します。別のクラスターサブネットグループを選択する場合は、選択したサブネットグループに 3 つのアベイラビリティーゾーンがあることを確認します。
10. [Additional configurations] (追加設定)で、[Database configurations] (データベース構成) を展開します。
11. [データベース暗号化] で、デフォルトの AWS が所有するキー以外のカスタム KMS キーを使用するには、[暗号化設定をカスタマイズ] をクリックします。このオプションはデフォルトでは選択されていません。
12. [Choose an KMS key] (KMS キーの選択) で、AWS Key Management Service キーを選択するか、ARN を入力できます。または、AWS Key Management Service コンソールで [AWS Key Management Service キーを作成] をクリックすることもできます。KMS キーの作成の詳細については、AWS Key Management Service デベロッパーガイドの キーの作成 を参照してください。
13. [Restore cluster from snapshot] (スナップショットからクラスターを復元) をクリックします。クラスターの復元が成功すると、クラスターの詳細ページで詳細を確認できます。

の使用AWS Command Line Interface

- 次のように、AWS CLI から `restore-from-cluster-snapshot` コマンドを使用します。

```
aws redshift restore-from-cluster-snapshot
--region eu-west-1
--multi-az
--snapshot-identifier test-snap1
--cluster-identifier test-saz-11
```

```
--endpoint-url https://redshift.eu-west-1.amazonaws.com/
```

シングル AZ データウェアハウスのマルチ AZ データウェアハウスへの変換

シングル AZ データウェアハウスをマルチ AZ データウェアハウスに変換することで、データウェアハウスは 99.99% の SLA 保証で高い可用性を実現できます。マルチ AZ データウェアハウスを使用しても、個々のクエリのパフォーマンスは変わりません。同時実行性の高いワークロードでは、Amazon Redshift が 2 つのアベイラビリティーゾーンのコンピューティングリソースを使用してリクエストを実行できるため、全体的なスループットが向上します。

> **Note**
>
> Amazon Redshift では、シングル AZ からマルチ AZ に、またはその逆に変換する際に、既存のコンピューティングリソースを分割することはできません。個々のクエリのパフォーマンスを一定に保つため、このオペレーションはサポートされていません。

コンソールを使用する

コンソールを使用して、シングル AZ クラスターをマルチ AZ データウェアハウスに変換するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [Provisioned clusters dashboard] (プロビジョニングされたクラスターダッシュボード) を選択し、[Clusters] (クラスター) を選択します。現在の AWS リージョンにあるアカウントのクラスターがリストされています。各クラスターのプロパティのサブセットが、リストの列に表示されます。
3. マルチ AZ 配置に変換するクラスターを選択します。クラスターの詳細ページが表示されます。
4. [アクション] で [マルチ AZ をアクティブ化] を選択します。[変更の概要] が表示されます。[マルチ AZ をアクティブ化] をクリックします。
5. エラーが発生した場合は、次のいずれかを実行し、[マルチ AZ をアクティブ化] をクリックします。

   - クラスター暗号化 — クラスター詳細ページの [プロパティ] タブにある [データベース設定] セクションで、[プロパティ] を選択して暗号化設定を編集します。
   - サブネットグループ — [サブネットグループ] を選択し、サブネットグループリンクをクリックしてクラスターサブネットグループの設定を編集します。別のクラスターサブネットグルー

プを選択する場合は、選択したサブネットグループに 3 つのアベイラビリティーゾーンがあることを確認します。

- ポート設定 — クラスター詳細ページの [プロパティ] タブにある [データベース設定] セクションで、[プロパティ] を選択してポート設定を編集します。

6. SQL クライアントを使用して、データをロードしたりクエリしたりできます。

の使用AWS Command Line Interface

- AWS CLI から `modify-cluster` コマンドと `multi-az` パラメータを次のように使用します。

```
aws redshift modify-cluster
    --profile maz-test
    --endpoint-url https://redshift.eu-west-1.amazonaws.com
    --region eu-west-1
    --cluster-identifier test-maz-11
    --multi-az
```

STL、SVCS、SVL、SVV、STV ビューはシステムモニタリングビュー (SYS_* ビュー) のみをサポートしているため、マルチ AZ 配置では使用できません。システムモニタリングビュー (SYS_* ビュー) を使用するようにモニタリングクエリを変更してください。

マルチ AZ データウェアハウスのシングル AZ データウェアハウスへの変換

マルチ AZ データウェアハウスをシングル AZ データウェアハウスに変換すると、データウェアハウスは 99.99% の SLA 保証を実現できなくなります。個々のクエリのパフォーマンスは変わりませんが、2 番目のアベイラビリティーゾーンのコンピューティングリソースが使用できなくなるため、全体的なスループットに影響します。同時実行スケーリングを有効にして、シングル AZ でもパフォーマンスが一定になるようにスループットを自動的にスケーリングすることもできます。

> **Note**
>
> Amazon Redshift では、シングル AZ からマルチ AZ に、またはその逆に変換する際に、既存のコンピューティングリソースを分割することはできません。個々のクエリのパフォーマンスを一定に保つため、このオペレーションはサポートされていません。

コンソールを使用する

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [Provisioned clusters dashboard] (プロビジョニングされたクラスターダッシュボード) を選択し、[Clusters] (クラスター) を選択します。現在の AWS リージョンにあるアカウントのクラスターがリストされています。各クラスターのプロパティのサブセットが、リストの列に表示されます。
3. マルチ AZ 配置に変換するクラスターを選択します。クラスターの詳細ページが表示されます。
4. [アクション] で [マルチ AZ を非アクティブ化] を選択します。[変更の概要] が表示されます。[マルチ AZ を非アクティブ化] をクリックします。

の使用AWS Command Line Interface

- AWS CLI から `modify-cluster` コマンドと `no-multi-az` パラメータを次のように使用します。

```
aws redshift modify-cluster
    --profile maz-test
    --endpoint-url  https://redshift.eu-west-1.amazonaws.com
    --region eu-west-1
    --cluster-identifier test-maz-11
    --no-multi-az
```

データウェアハウスがシングル AZ に変換されると、99.99 の SLA 保証は失われます。全体的なスループットも影響を受けます。変更を保存すると、クラスターの詳細ページで詳細を確認できます。

マルチ AZ データウェアハウスのサイズ変更

マルチ AZ データウェアハウスをサイズ変更し、データウェアハウスの現在の設定と異なっているノード数またはノードタイプを指定できます。

コンソールを使用する

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [Provisioned clusters dashboard] (プロビジョニングされたクラスターダッシュボード) を選択し、[Clusters] (クラスター) を選択します。現在の AWS リージョン

にあるアカウントのクラスターがリストされています。各クラスターのプロパティのサブセットが、リストの列に表示されます。

3. マルチ AZ データウェアハウスのサイズを変更するクラスターを選択します。クラスターの詳細ページが表示されます。
4. [アクション] で、[サイズ変更] を選択します。[クラスターのサイズ変更] ページが表示されます。
5. ページに表示された手順に従います。クラスターのサイズ変更は、特定の時刻に 1 回行うことも、スケジュールに従ってクラスターのサイズを増減することもできます。
6. [新しい設定] にある [ノードタイプ] ドロップダウンリストから RG または RA3 ノードタイプの 1 つを選択します。
7. [クラスターのサイズ変更] をクリックします。

の使用AWS Command Line Interface

AWS Command Line Interface を使用してマルチ AZ データウェアハウスのサイズを変更するには

- AWS CLI から、以下のように `resize-cluster` コマンドを使用して 1 つのアベイラビリティーゾーンのノード数を変更します。

```
aws redshift resize-cluster \
    --cluster-identifier test-maz-11
    --cluster-type multi-node
    --node-type rg.4xlarge
    --number-of-nodes 6
```

マルチ AZ 配置のフェールオーバー

マルチ AZ データウェアハウスは、2 つのアベイラビリティーゾーンに同時にデプロイされるコンピューティングリソースのコレクションです。プライマリアベイラビリティーゾーンにデプロイされたコンピューティングリソースはプライマリコンピューティングと呼ばれ、セカンダリアベイラビリティーゾーンにデプロイされたコンピューティングリソースはセカンダリコンピューティングと呼ばれます。マルチ AZ データウェアハウスは、アベイラビリティーゾーンやインフラストラクチャの障害など、万が一発生してもユーザーの介入なしに自動的に復旧できます。復旧プロセスでは、プライマリコンピューティングからセカンダリコンピューティングへフェイルオーバーし、セカンダリコンピューティングリソースをプライマリとして指定します。さらに、新しいセカンダリコンピューティ

ングリソースは 3 番目のアベイラビリティーゾーンにプロビジョニングされます。自動復旧プロセスは RTO と RPO の観点から測定されます。

- 目標復旧時間 (RTO)– 災害後にシステムが稼働状態に戻るまでにかかる時間。つまり、RTO はダウンタイムを測定します。
- 目標復旧時点 (RPO) – 損失する可能性があるデータの量 (時間単位)。Amazon Redshift マルチ AZ データウェアハウスの場合、RPO は通常ゼロです。これは、すべてのデータが Amazon Redshift マネージドストレージ (RMS) に保存され、耐久性が高くデフォルトで利用可能な Amazon Simple Storage Service によってバックアップされるためです。

> **Note**
>
> フェイルオーバーが発生しても、個々のクエリのパフォーマンスのパフォーマンスは変わりません。アベイラビリティーゾーンのいずれかでコンピューティングリソースが使用できなくなるため、データウェアハウスの全体的なスループットが短時間低下します。ただし、Amazon Redshift は別のアベイラビリティーゾーンのキャパシティを自動的に取得して、同じデータウェアハウスの処理能力が回復するようにします。

自動復旧プロセスに加えて、[フェイルオーバープライマリコンピューティング] オプションを使用して、データウェアハウスでこのプロセスを手動で開始することもできます。このアプローチを使用して、マルチ AZ がアプリケーションの高可用性と継続性の向上にどのように役立つかをテストできます。

コンソールを使用する

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. 次のいずれかを行います。
   - ナビゲーションメニューで [クラスター] を選択します。[Clusters] (クラスター) で、クラスターを選択します。クラスターの詳細ページが表示されます。
   - クラスターダッシュボードから、クラスターを選択します。
3. [アクション] から [フェイルオーバープライマリコンピューティング] を選択します。
4. メッセージが表示されたら、[Confirm] (確認) をクリックします。

の使用AWS Command Line Interface

- 次のように、AWS CLI から `failover-primary-compute` コマンドを使用します。

```
aws redshift failover-primary-compute
    --profile maz-test
    --endpoint-url  https://redshift.eu-west-1.amazonaws.com
    --region eu-west-1
    --cluster-identifier test-maz-11
```

上記の操作が確認されると、Amazon Redshift はアベイラビリティーゾーンまたはインフラストラクチャ障害からの自動復旧と同じ手順を実行します。このプロセスにより、プライマリアベイラビリティーゾーンのコンピューティングノードが使用できなくなり、セカンダリアベイラビリティーゾーンのコンピューティングリソースがプライマリコンピューティングとして指定されます。クラスターの復旧が正常に完了すると、マルチ AZ 配置が使用可能になります。また、マルチ AZ データウェアハウスは、別の 3 番目のアベイラビリティーゾーンが利用可能になり次第、新しいセカンダリコンピューティングを自動的にプロビジョニングします。

このプロセス中、クラスターが自動的に回復してマルチ AZ 配置設定に再構成されるため、その時間中はコンソール上のクラスターステータスは変更中と表示されます。クラスターは新しい接続をすぐに受け入れることができます。既存の接続と実行中のクエリは削除される可能性があります。すぐに再試行できます。

マルチ AZ データウェアハウスのクエリとロードの表示

クラスターのタイプ、サイズ、ステータス (一時停止または再開) に関係なく、過去 7 日間に実行されたクエリに関する情報を表示できます。

クエリとロードページに表示される情報には、Amazon Redshift システムテーブル (SYS_* ビュー) からの情報が入力されます。この情報により、クエリやオファーに関する追加情報を表示できます。保持期間は 7 日間です。クエリ診断が速くなり、データベース、ユーザー名、または SQL ステートメントのタイプでデータをフィルタリングできます。これらの追加フィルターと実行されたすべてのクエリに関する情報を確認するには、次の前提条件に注意してください。

- [Connect to database] (データベースに接続) を選択してデータベースに接続する必要があります。
- クエリのモニタリングを実行するには、データベースユーザーに sys: operator または sys: monitor のロールと権限が必要です。システムロールの詳細については、「Amazon Redshift データベースデベロッパーガイド」の「Amazon Redshift でのシステム定義のロール」を参照してください。

データベースに接続すると、これらの追加のフィルターとクエリ情報が表示されます。

クエリとロードからクエリパフォーマンスデータを表示する方法

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Queries and loads] (クエリおよびロード) を選択し、アカウントのクエリのリストを表示します。
3. 追加のフィルターを表示するには、データベースに接続する必要がある場合があります。必要に応じて、[Connect to database] (データベースに接続) をクリックし、プロンプトに従ってデータベースに接続します。

   デフォルトでは、全クラスターの過去 24 時間のクエリがリストに表示されます。コンソールで表示日付の範囲を変更できます。

クエリのモニタリングからクエリパフォーマンスデータを表示する方法

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [クラスター] を選択します。[Clusters] (クラスター) で、クラスターを選択します。
3. [Query monitoring] (クエリのモニタリング) を選択します。
4. クラスターの構成やバージョンによっては、追加のフィルターを表示するためにデータベースに接続する必要がある場合があります。必要に応じて、[Connect to database] (データベースに接続) をクリックし、プロンプトに従ってデータベースに接続します。

マルチ AZ 配置でのクエリのモニタリング

マルチ AZ 配置では、両方のアベイラビリティーゾーンにデプロイされたコンピューティングリソースを使用し、特定のアベイラビリティーゾーンのリソースが使用できない場合でも運用を継続できます。すべてのコンピューティングリソースは常に使用されます。これにより、2 つのアベイラビリティーゾーンで、読み取りと書き込みの操作の両方をアクティブ-アクティブ構成で完全に操作できます。

pg_catalog スキーマの SYS_ ビューをクエリして、マルチ AZ 配置のクエリ実行時間をモニタリングできます。SYS_ ビューには、プライマリクラスターとセカンダリクラスターのクエリ実行時ア

クティビティまたは統計が表示されます。モニタリングビューの一覧については、「モニタリングビュー」を参照してください。

以下の手順に従って、マルチ AZ 配置内の各アベイラビリティーゾーンのクエリ実行時間をモニタリングします。

1. Amazon Redshift コンソールに移動し、マルチ AZ 配置のデータベースに接続し、クエリエディタを使用してクエリを実行します。
2. マルチ AZ の Amazon Redshift デプロイで任意のサンプルクエリを実行します。
3. マルチ AZ 配置の場合、SYS_QUERY_HISTORY テーブルの compute_type 列を使用して、クエリとその実行場所のアベイラビリティーゾーンを識別できます。プライマリはマルチ AZ 配置のプライマリクラスターで実行されるクエリを表し、セカンダリはマルチ AZ 配置のセカンダリクラスターで実行されるクエリを表します。

   次のクエリは、compute_type 列を使用してクエリをモニタリングします。

```
select (compute_type) as compute_type, left(query_text, 50) query_text from
 sys_query_history order by start_time desc;

 compute_type | query_text
--------------+--------------------------
   secondary  | select count(*) from t1;
```

クラスターのクエリを終了する

この手順は、マルチ AZ クラスターとシングル AZ クラスターの両方に適用されます。

クエリを終了するには

また、[クエリ] ページを使用して、現在進行中のクエリを終了することもできます。

実行中のクエリを終了するには、データベースユーザーに sys:operator ロールと権限が必要です。システムロールの詳細については、「Amazon Redshift データベースデベロッパーガイド」の「Amazon Redshift でのシステム定義のロール」を参照してください。

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Queries and loads] (クエリおよびロード) を選択し、アカウントのクエリのリストを表示します。

3. リストで終了する実行中クエリを選択してから、[クエリの終了] を選択します。

Amazon Redshift クラスターパフォーマンスのモニタリング

Amazon Redshift では、クラスターとデータベースの状態やパフォーマンスを追跡できるようにするため、パフォーマンスメトリクスとデータが提供されます。このセクションでは、Amazon Redshift (特に Amazon Redshift コンソール) で使用できるデータのタイプについて説明していきます。

Amazon Redshift コンソールで使用できるパフォーマンスデータは、次の 2 つのカテゴリに分類されます。

- Amazon CloudWatch メトリクス – Amazon CloudWatch メトリクスは、CPU 使用率、レイテンシー、スループットなど、クラスターの物理的な側面をモニタリングするのに役立ちます。メトリクスデータは Amazon Redshift コンソールに直接表示されます。これは CloudWatch コンソールで表示することもできます。または、AWS CLI や AWS SDK の 1 つなど、メトリクスを操作するその他の方法で使用することもできます。
- クエリ/ロードのパフォーマンスデータ – パフォーマンスデータは、データベースのアクティビティとパフォーマンスをモニタリングするのに役立ちます。このデータは Amazon Redshift コンソールに集計され、CloudWatch メトリクスで表示される情報を、特定のデータベースクエリ/ロードイベントに簡単に関連付けることができます。また、ユーザー独自のカスタムパフォーマンスクエリを作成し、データベースで直接実行することもできます。クエリ/ロードパフォーマンスデータは、Amazon Redshift コンソールのみで表示されます。これは、CloudWatch メトリクスとして公開されていません。

パフォーマンスデータは Amazon Redshift コンソールに統合され、次の方法でより高いエクスペリエンスが提供されます。

- クラスターに関連付けられたパフォーマンスデータは、クラスターを表示し、サイズ変更などクラスターに関する決定を行うためにパフォーマンスデータが必要になる可能性があるときに、コンテキストに応じて表示されます。
- 一部のパフォーマンスメトリクスは、Amazon Redshift と比較して、より適切にスケーリングされた単位で CloudWatch コンソールに表示されます。たとえば、`WriteThroughput` は GB-s で表示されますが (CloudWatch ではバイト/秒)、これはノードのストレージ容量としてより一般的な単位です。

- クラスターのノードのパフォーマンスデータを、同じグラフで簡単にまとめて表示できます。このようにして、クラスターのすべてのノードのパフォーマンスを簡単にモニタリングできます。また、各ノードのパフォーマンスデータを表示することもできます。

Amazon Redshift では、パフォーマンスデータ (CloudWatch メトリクスおよびクエリ/ロードデータの両方) が追加料金なしで提供されます。パフォーマンスデータは毎分記録されます。パフォーマンスデータの過去の値は、Amazon Redshift コンソール内でアクセスできます。CloudWatch を使用して CloudWatch メトリクスとして公開される Amazon Redshift パフォーマンスデータにアクセスする方法については、Amazon CloudWatch ユーザーガイドの「CloudWatch とは」を参照してください。

Amazon Redshift のパフォーマンスデータ

Amazon Redshift の CloudWatch メトリクスを使うと、クラスターのヘルスとパフォーマンスに関する情報を取得し、ノードレベルで情報を表示できます。これらのメトリクスを使用するときは、各メトリクスには 1 つ以上のディメンションが関連付けられていることに注意してください。これらのディメンションにより、メトリクスが適用される対象、つまりメトリクスの範囲がわかります。 には次の 2 つのディメンションがあります。Amazon Redshift には、次の 2 つのディメンションがあります。

- `NodeID` ディメンションを持つメトリクスは、クラスターのノードのパフォーマンスデータを提供するメトリクスです。このメトリクスのセットには、リーダーノードおよびコンピューティングノードが含まれます。これらのメトリクスの例には、`CPUUtilization`、`ReadIOPS`、`WriteIOPS` があります。
- `ClusterIdentifier` ディメンションのみを持つメトリクスは、クラスターのパフォーマンスデータを提供するメトリクスです。これらのメトリクスの例には、`HealthStatus` や `MaintenanceMode` があります。

> **Note**
>
> 一部のメトリクスでは、クラスター固有のメトリクスはノード動作の集計を表します。このような場合は、メトリクス値の解釈には注意が必要です。これは、リーダーノードの動作はコンピューティングノードとともに集計されるためです。

CloudWatch メトリクスとアラームの詳細については、Amazon CloudWatch ユーザーガイドの「CloudWatch の概念」を参照してください。

Amazon Redshift の CloudWatch メトリクスの詳細については、以下のセクションを参照してください。

トピック

- Amazon Redshift メトリクス
- Amazon Redshift メトリクスのディメンション
- Amazon Redshift クエリおよびロードパフォーマンスデータ

Amazon Redshift メトリクス

AWS/Redshift 名前空間には、次のメトリクスが含まれます。特に明記されていない限り、メトリクスは 1 分間隔で収集されます。

メトリクス	説明
CommitQueueLength	特定の時点のコミット待ちのトランザクションの数。 単位: 個 ディメンション: ClusterIdentifier
ConcurrencyScaling ActiveClusters	任意の時点でアクティブにクエリを処理している同時実行スケーリングクラスターの数。 単位: 個 ディメンション: ClusterIdentifier
ConcurrencyScaling Seconds	アクティブなクエリ処理アクティビティを持つ同時実行スケーリングクラスターによって使用される秒数。 単位: カウント ディメンション: ClusterIdentifier
CPUUtilization	CPU 使用率。このメトリクスは、クラスターに対するすべてのノード (リーダーおよびコンピューティング) の CPU 使用率の値の集計を表します。 単位: パーセント

メトリクス	説明
	ディメンション: ClusterIdentifier 、NodeID ディメンション: ClusterIdentifier
DatabaseConnections	クラスターへのデータベース接続の数。 単位: 個 ディメンション: ClusterIdentifier
HealthStatus	クラスターの状態を示します。クラスターは毎分データベースに接続し、簡単なクエリを実行します。この操作を正しく実行できる場合、クラスターは正常な状態であると見なされます。それ以外の場合、クラスターは正常な状態ではありません。正常でない状態が発生する可能性があるのは、クラスターデータベースに極端に高い負荷がかかっているか、クラスター上のデータベースに設定の問題がある場合です。 **Note** Amazon CloudWatch では、このメトリクスは 1 または 0 と報告されますが、Amazon Redshift コンソールでは、このメトリックはわかりやすくするために HEALTHY または UNHEALTHY という単語で表示されます。このメトリクスが Amazon Redshift コンソールで表示される場合、サンプリング平均は無視され、HEALTHY または UNHEALTHY のみが表示されます。Amazon CloudWatch では、サンプリングの問題により、1 または 0 以外の値が発生する場合があります。HealthStatus に対して 1 未満の値は 0 (UNHEALTHY) と報告されます。 単位: カウント (1/0) (Amazon Redshift コンソールの HEALTHY/UNHEALTHY) ディメンション: ClusterIdentifier

メトリクス	説明
MaintenanceMode	クラスターがメンテナンスモードかどうかを示します。 **Note** Amazon CloudWatch では、このメトリクスは 1 または 0 と報告されますが、Amazon Redshift コンソールでは、このメトリックはわかりやすくするために ON または OFF という単語で表示されます。このメトリクスが Amazon Redshift コンソールで表示される場合、サンプリング平均は無視され、ON または OFF のみが表示されます。Amazon CloudWatch では、サンプリングの問題により、1 または 0 以外の値が発生する場合があります。MaintenanceMode に対して 0 より大きい値は 1 (ON) と報告されます。 単位: カウント (1/0) (Amazon Redshift コンソールの ON/OFF) ディメンション: ClusterIdentifier
MaxConfiguredConcurrencyScalingClusters	パラメータグループから設定された同時実行スケーリングクラスターの最大数。詳細については、Amazon Redshift パラメータグループを作成します。 を参照してください。 単位: 個 ディメンション: ClusterIdentifier
NetworkReceiveThroughput	ノードまたはクラスターがデータを受け取るレート。 単位: バイト/秒 (Amazon Redshift コンソールでは MB/秒) ディメンション: ClusterIdentifier , NodeID ディメンション: ClusterIdentifier

メトリクス	説明
NetworkTransmitThroughput	ノードまたはクラスターがデータを書き込むレート。 単位: バイト/秒 (Amazon Redshift コンソールでは MB/秒) ディメンション: ClusterIdentifier , NodeID ディメンション: ClusterIdentifier
PercentageDiskSpaceUsed	使用中のディスク容量の割合。 単位: パーセント ディメンション: ClusterIdentifier ディメンション: ClusterIdentifier 、NodeID
QueriesCompletedPerSecond	1 秒あたりに完了したクエリの平均回数。5 分間隔で報告されます。このメトリクスは、単一ノードクラスターではサポートされていません。 単位: カウント/秒 ディメンション: ClusterIdentifier , latency ディメンション: ClusterIdentifier , wlmid
QueryDuration	クエリを完了するまでの平均時間。5 分間隔で報告されます。このメトリクスは、単一ノードクラスターではサポートされていません。 単位: マイクロ秒 ディメンション:ClusterIdentifier 、NodeID、latency ディメンション: ClusterIdentifier , latency ディメンション:ClusterIdentifier 、NodeID、wlmid

メトリクス	説明
QueryRuntimeBreakdown	クエリステージごとのクエリの実行にかかった合計時間。5 分間隔で報告されます。 単位: ミリ秒 ディメンション: ClusterIdentifier、NodeID、stage ディメンション: ClusterIdentifier、stage
ReadIOPS	1 秒あたりのディスク読み取り操作の平均回数。 単位: カウント/秒 ディメンション: ClusterIdentifier , NodeID ディメンション: ClusterIdentifier
ReadLatency	ディスク読み取り I/O 操作にかかる平均時間。 単位: Seconds ディメンション: ClusterIdentifier , NodeID ディメンション: ClusterIdentifier
ReadThroughput	1 秒あたりのディスクからの平均読み取りバイト数。 単位: バイト (Amazon Redshift コンソールでは GB-s) ディメンション: ClusterIdentifier , NodeID ディメンション: ClusterIdentifier
RedshiftManagedStorageTotalCapacity	マネージドストレージの総容量。 単位: メガバイト ディメンション: ClusterIdentifier

メトリクス	説明
RgUniqueSnapshotStorage	RG インスタンスファミリーのすべてのアクティブな手動スナップショットに保存されている一意のデータブロックのメガバイト数。 単位: メガバイト
TotalTableCount	特定の時点でのテーブルの総数。この合計には、永続テーブル、一時テーブル、データ共有テーブル、外部テーブル、およびマテリアライズドビューが含まれます。ビューおよびシステムテーブルはこの数には含まれません。 単位: カウント ディメンション: ClusterIdentifier
WLMQueueLength	ワークロード管理 (WLM) キューに入るために待機しているクエリの数。 単位: 個 ディメンション: ClusterIdentifier , service class ディメンション: ClusterIdentifier , QueueName
WLMQueueWaitTime	ワークロード管理 (WLM) キューでクエリの待機にかかった合計時間。5 分間隔で報告されます。 単位: ミリ秒 ディメンション: ClusterIdentifier , QueryPriority ディメンション: ClusterIdentifier , wlmid ディメンション: ClusterIdentifier , QueueName

メトリクス	説明
WLMQueriesCompletedPerSecond	ワークロード管理 (WLM) キューで完了した 1 秒あたりの平均クエリ数。5 分間隔で報告されます。このメトリクスは、単一ノードクラスターではサポートされていません。 単位: カウント/秒 ディメンション: ClusterIdentifier , wlmid ディメンション: ClusterIdentifier , QueueName
WLMQueryDuration	ワークロード管理 (WLM) キューのクエリを完了するまでの平均時間。5 分間隔で報告されます。このメトリクスは、単一ノードクラスターではサポートされていません。 単位: マイクロ秒 ディメンション: ClusterIdentifier , wlmid ディメンション: ClusterIdentifier , QueueName
WLMRunningQueries	WLM キューごとのメインクラスターと同時実行スケーリングクラスターの両方から実行されているクエリの数。 単位: 個 ディメンション: ClusterIdentifier , wlmid ディメンション: ClusterIdentifier , QueueName
WriteIOPS	1 秒あたりの書き込み操作の平均回数 単位: カウント/秒 ディメンション: ClusterIdentifier , NodeID ディメンション: ClusterIdentifier

メトリクス	説明
WriteLatency	ディスク書き込み I/O 操作にかかる平均時間。 単位: Seconds ディメンション: ClusterIdentifier , NodeID ディメンション: ClusterIdentifier
WriteThroughput	1 秒あたりのディスクへの平均書き込みバイト数。 単位: バイト (Amazon Redshift コンソールでは GB-s) ディメンション: ClusterIdentifier , NodeID ディメンション: ClusterIdentifier
SchemaQuota	スキーマの構成済みクォータ。 単位: メガバイト ディメンション:ClusterIdentifier 、Database、Schema 周期/プッシュ: Periodic 頻度: 5 minutes 停止条件:スキーマの削除またはクォータの削除
NumExceededSchemaQuotas	クォータを超過したスキーマの数。 単位: 個 ディメンション: ClusterIdentifier 周期/プッシュ: Periodic 頻度: 5 minutes 停止基準: 該当なし

メトリクス	説明
StorageUsed	スキーマによって使用されるディスクまたはストレージ領域。 単位: メガバイト ディメンション:ClusterIdentifier、Database、Schema 周期/プッシュ: Periodic 頻度: 5 minutes 停止条件:スキーマの削除またはクォータの削除
PercentageQuotaUsed	構成されたスキーマクォータを基準としたディスクまたは記憶域スペースの割合。 単位: パーセント ディメンション:ClusterIdentifier、Database、Schema 周期/プッシュ: Periodic 頻度: 5 minutes 停止条件:スキーマの削除またはクォータの削除

メトリクス	説明
UsageLimitAvailable	FeatureType に応じて、UsageLimitAvailable は次の値を返します。 • FeatureType が CONCURRENCY_SCALING の場合、UsageLimitAvailable は、同時実行スケーリングで使用できる合計時間を 1 分単位で返します。この時間は CONCURRENCY_SCALING_AUTO_TASK で使用できる時間としてはカウントされず、その逆も同様です。 • FeatureType が CONCURRENCY_SCALING_AUTO_TASK の場合、UsageLimitAvailable は、自動最適化タスクの同時実行スケーリングで使用できる合計時間を返します。この時間は CONCURRENCY_SCALING で使用できる時間としてはカウントされず、その逆も同様です。 • FeatureType が EXTRA_COMPUTE_FOR_AUTOMATIC_OPTIMIZATION の場合、UsageLimitAvailable は、自動最適化タスクに使用できる合計時間を返します。 • FeatureType が CROSS_REGION_DATASHARING の場合、UsageLimitAvailable は、スキャンできるデータの合計時間を 1-TB 単位で返します。 • FeatureType が SPECTRUM の場合、UsageLimitAvailable は、スキャンできるデータの合計時間を 1-TB 単位で返します。 単位: 分または TB ディメンション:ClusterIdentifier 、FeatureType 、UsageLimitId

メトリクス	説明
UsageLimitConsumed	FeatureType に応じて、UsageLimitConsumed は次の値を返します。 • FeatureType が CONCURRENCY_SCALING の場合、UsageLimitConsumed は、同時実行スケーリングで使用する合計時間を 1 分単位で返します。この時間は CONCURRENCY_SCALING_AUTO_TASK で使用される時間としてはカウントされず、その逆も同様です。 • FeatureType が CONCURRENCY_SCALING_AUTO_TASK の場合、UsageLimitConsumed は、自動最適化タスクの同時実行スケーリングで使用された合計時間を返します。この時間は CONCURRENCY_SCALING で使用される時間としてはカウントされず、その逆も同様です。 • FeatureType が EXTRA_COMPUTE_FOR_AUTOMATIC_OPTIMIZATION の場合、UsageLimitConsumed は、自動最適化タスクで使用された合計時間を返します。 • FeatureType が CROSS_REGION_DATASHARING の場合、UsageLimitConsumed は、スキャンできるデータの合計時間を 1-TB 単位で返します。 • FeatureType が SPECTRUM の場合、UsageLimitConsumed は、スキャンできるデータの合計時間を 1-TB 単位で返します。 単位: 分または TB ディメンション:ClusterIdentifier 、FeatureType 、UsageLimitId

Amazon Redshift メトリクスのディメンション

Amazon Redshift のデータは、次の表にあるいずれかのディメンションでフィルタリングできます。

ディメンション	説明
latency	指定できる値は次のとおりです。 • short – 10 秒未満 • medium – 10 秒 ~ 10 分 • long – 10 分超過
NodeID	フィルターにより、クラスターのノードに固有のデータが要求されました。NodeID は、「Leader」、「Shared」、または「Compute-N」のいずれかになります。ここで、N はクラスターのノード数を表す 0、1、... です。"Shared" は、クラスターには 1 つのノードのみがあることを意味します。つまり、リーダーノードとコンピューティングノードが組み合わされています。 CPUUtilization 、NetworkTransmitThroughput 、および ReadIOPS のメトリクスは、リーダーノードおよびコンピューティングノードに対してのみレポートされます。NodeId ディメンションを使用するその他のメトリクスは、コンピューティングノードに対してのみレポートされます。
ClusterIdentifier	フィルターにより、クラスターに固有のデータが要求されました。クラスターに固有のメトリクスには、HealthStatus 、MaintenanceMode 、DatabaseConnections があります。ノードのメトリクスでもある、このディメンションの全般的なメトリクス (ReadIOPS など) は、ノードメトリクスデータの集計を表します。これらのメトリクスは、リーダーノードおよびコンピューティングノードの動作を集計するため、その解釈には注意してください。
service class	WLM サービスクラスの識別子。
stage	クエリの実行ステージ。指定できる値は次のとおりです。 • QueryPlanning: SQL ステートメントの解析と最適化にかかった時間。 • QueryWaiting: WLM キューの待機にかかった時間。

ディメンション	説明
	• QueryExecutingRead: 読み込みクエリの実行にかかった時間。 • QueryExecutingInsert: 挿入クエリの実行にかかった時間。 • QueryExecutingDelete: 削除クエリの実行にかかった時間。 • QueryExecutingUpdate: 更新クエリの実行にかかった時間。 • QueryExecutingCtas: テーブル作成クエリの実行にかかった時間。 • QueryExecutingUnload: アップロードクエリの実行にかかった時間。 • QueryExecutingCopy: コピークエリの実行にかかった時間。 • QueryCommit: コミットにかかった時間。
wlmid	ワークロード管理キューの ID。
QueryPriority	クエリの優先度です。使用できる値は、CRITICAL、HIGHEST、HIGH、NORMAL、LOW、および LOWEST です。
QueueName	ワークロード管理キューの名前。
FeatureType	使用制限によって制限される機能。指定できる値は CONCURRENCY_SCALING 、CROSS_REGION_DATASHARING 、および SPECTRUM です。
UsageLimitId	使用制限の識別子。

Amazon Redshift クエリおよびロードパフォーマンスデータ

CloudWatch メトリクスに加えて、Amazon Redshift はクエリ/ロードパフォーマンスデータを提供します。クエリ/ロードパフォーマンスデータを使用すると、データベースのパフォーマンスとクラスターメトリクス間の関係を理解するのに役立ちます。例えば、クラスターの CPU 使用率が急上昇したことに気付いた場合、クラスターの CPU グラフでその上昇を見つけ、そのときに実行されていたクエリを表示できます。逆に、特定のクエリについて確認している場合、メトリクスデータ (CPU な

ど) がコンテキストに表示されるので、クラスターメトリクスに対するクエリの影響について理解できます。

クエリおよびロードパフォーマンスデータは CloudWatch メトリクスとして公開されず、Amazon Redshift コンソールのみで表示できます。クエリとロードのパフォーマンスデータは、データベースのシステムテーブルを使用したクエリから生成されます (詳細については、Amazon Redshift デベロッパーガイドの「システムテーブルのリファレンス」を参照してください)。ユーザー独自のカスタムデータベースパフォーマンスクエリを生成することもできますが、コンソールに表示されているクエリ/ロードパフォーマンスデータで開始することをお勧めします。データベースパフォーマンスを自分で測定およびモニタリングする方法については、Amazon Redshift デベロッパーガイドの「パフォーマンスの管理」を参照してください。

次の表に、Amazon Redshift コンソールでアクセスできるクエリ/ロードデータのさまざまな側面を示します。

クエリ/ロードデータ	説明
クエリの概要	指定された期間のクエリのリストです。このリストは、クエリ ID、クエリ実行時間、ステータスなどの値に基づいて並べ替えることができます。このデータは、クラスターの詳細ページの [クエリのモニタリング] タブに表示されます。
クエリの詳細	次のものを含む、特定のクエリの詳細を提供します。 • クエリ ID、タイプ、クエリが実行されたクラスター、実行時間などのクエリのプロパティ。 • クエリのステータス、エラー数などの詳細。 • 実行された SQL ステートメント。 • 利用可能な場合は説明プラン。 • クエリ実行中のクラスターのパフォーマンスデータ (詳細については、「クエリ履歴データの表示」を参照してください)。
ロードの概要	指定された期間のすべてのロードをリストします。このリストは、クエリ ID、クエリ実行時間、ステータスなどの値に基づいて並べ替えることができます。このデータは、クラスターの詳細ページの [Query monitoring (クエリのモニタリング)] タブに表示されます。

クエリ/ロードデータ	説明
ロードの詳細	次のものを含む、特定のロードオペレーションの詳細を提供します。 • クエリ ID、タイプ、クエリが実行されたクラスター、実行時間などのロードのプロパティ。 • ロードのステータス、エラー数などの詳細。 • 実行された SQL ステートメント。 • ロードされたファイルのリスト。 • ロード操作中のクラスターのパフォーマンスデータ (詳細については、「クエリ履歴データの表示」を参照してください)。

パフォーマンスデータの表示

このセクションでは、クラスターおよびクエリのパフォーマンスに関する情報を含むパフォーマンスデータを Amazon Redshift コンソールで表示する方法について説明します。さらに、Amazon Redshift コンソールから直接クラスターでアラームを作成できます。

Amazon Redshift コンソールでパフォーマンスデータを表示するときは、クラスター別に表示できます。クラスターのパフォーマンスデータグラフは、パフォーマンスに関するほとんどの一般的な疑問に答えるデータにアクセスできるように設計されています。パフォーマンスデータによっては (Amazon Redshift のパフォーマンスデータ を参照)、CloudWatch を使用してメトリクスグラフをさらにカスタマイズすることもできます。たとえば、より長い時間を選択したり、クラスター間でメトリクスを組み合わせたりできます。CloudWatch コンソールの使用の詳細については、「CloudWatch コンソールでのパフォーマンスメトリクス」を参照してください。

Amazon Redshift コンソールのクエリのモニタリング機能を使用して、クエリのモニタリング、分離、最適化を行う方法については、Query Monitoring with Amazon Redshift というビデオをご覧ください。

トピック

- クラスターのパフォーマンスデータを表示する
- クエリ履歴データの表示
- データベースパフォーマンスデータの表示
- ワークロードの同時実行と同時実行スケーリングデータの表示

- 自動最適化データの表示
- クエリとロードの表示
- クエリの詳細の表示と分析
- クエリの実行中にクラスターパフォーマンスを表示するには
- ロード操作中にクラスターメトリクスを表示する
- クラスターワークロードの内訳表の表示

クラスターのパフォーマンスデータを表示する

Amazon Redshift のクラスターメトリクスを使用することにより、次のような一般的なパフォーマンスタスクを実行することができるようになります。

- 指定された時間範囲でクラスターメトリクスが異常かどうかを判断し、異常な場合は、パフォーマンスに影響しているクエリを識別します。
- 過去または現在のクエリがクラスターのパフォーマンスに影響しているかどうかを確認します。問題があるクエリを特定した場合は、クエリ実行中のクラスターのパフォーマンスなど、それに関する詳細を確認できます。この情報を使用して、クエリが低速だった理由と、パフォーマンスを向上させるための操作を診断できます。

パフォーマンスデータを表示するには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [Clusters] (クラスター) を選択し、リストからクラスターの名前を選択してその詳細を開きます。クラスターの詳細が表示されます。これには、[クラスターのパフォーマンス]、[クエリのモニタリング]、[データベース]、[データ共有]、[スケジュール]、[メンテナンス]、および [プロパティ] タブなどがあります。
3. [Cluster performance (クラスターのパフォーマンス)] タブを選択して、次を含むパフォーマンス情報を表示します。

    - CPU 使用率
    - ディスク使用率
    - データベース接続
    - ヘルスステータス
    - クエリの期間

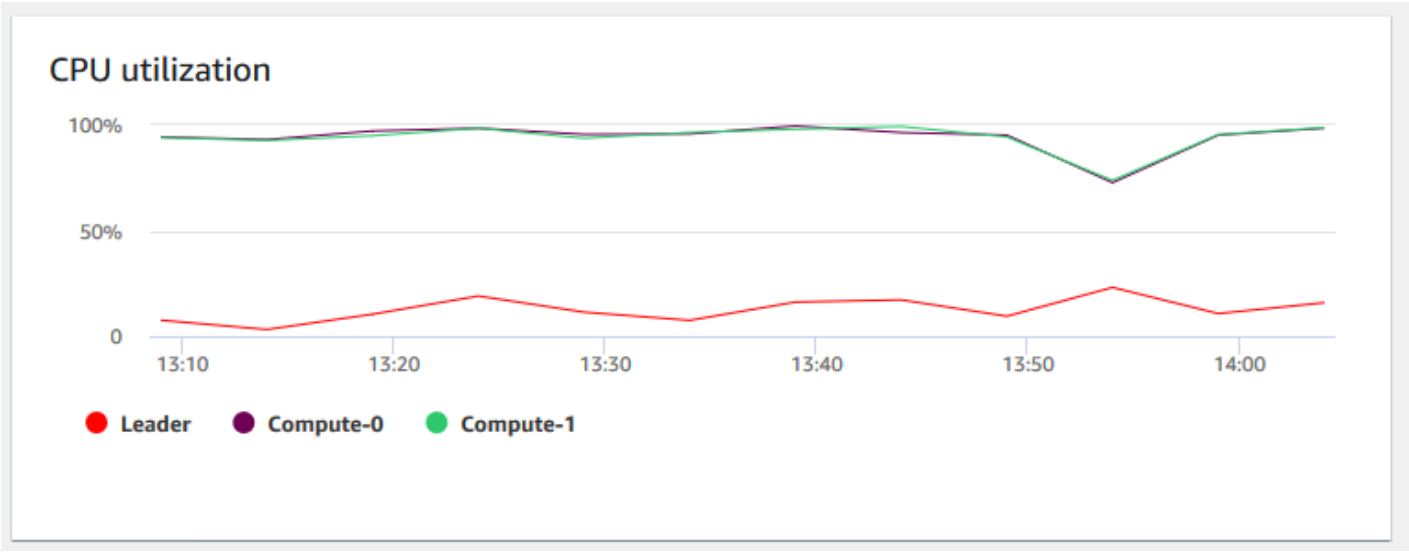
- クエリのスループット
- 同時実行スケーリングアクティビティ

利用可能なメトリクスが大幅に増えました。利用可能なメトリクスを確認して、表示するメトリクスを選択するには、[設定] アイコンを選択します。

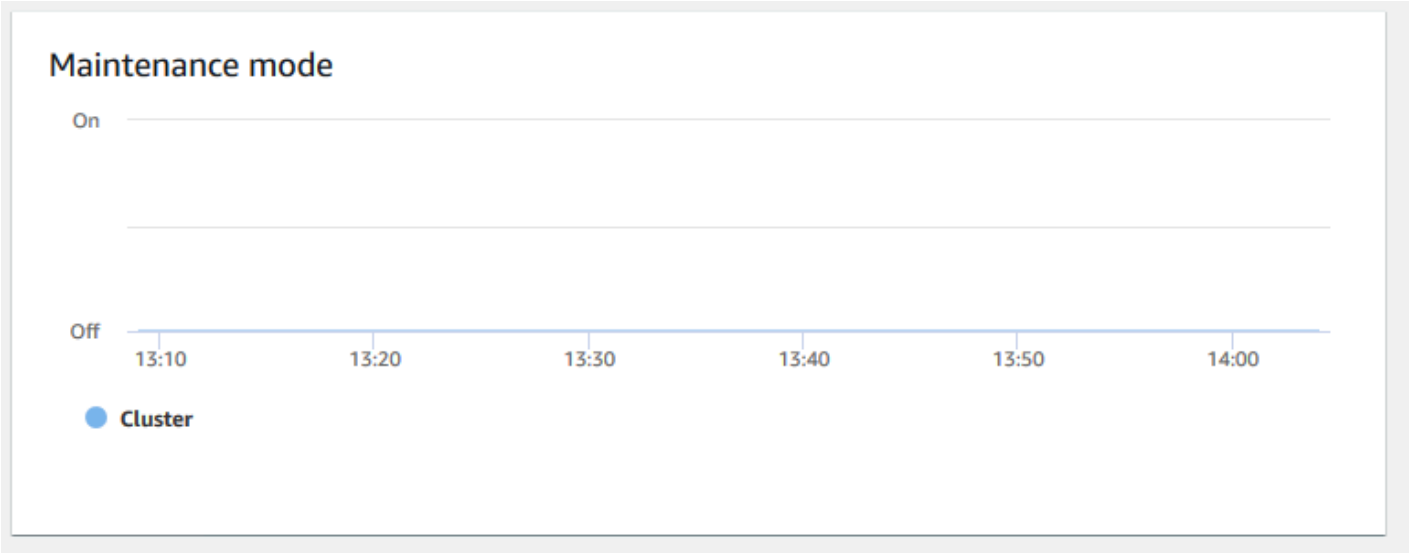
クラスターのパフォーマンスグラフ

次の例は、新しい Amazon Redshift コンソールに表示されるグラフの一部を示しています。

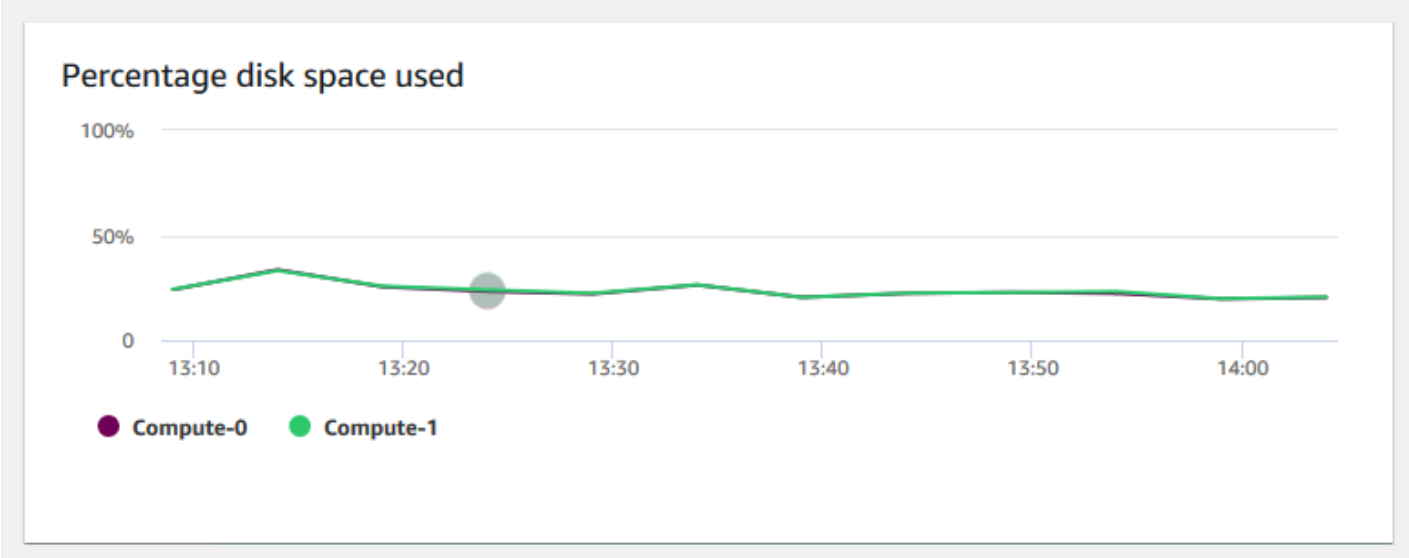
- CPU 使用率 – すべてのノード (リーダーとコンピューティング) の CPU 使用率を示します。クラスターの移行やリソースを消費するその他のオペレーションをスケジュールする前に、クラスターの使用率が最も低い時間を見つけるには、このグラフをモニタリングして、個々のノードまたはすべてのノードの CPU 使用率を確認します。



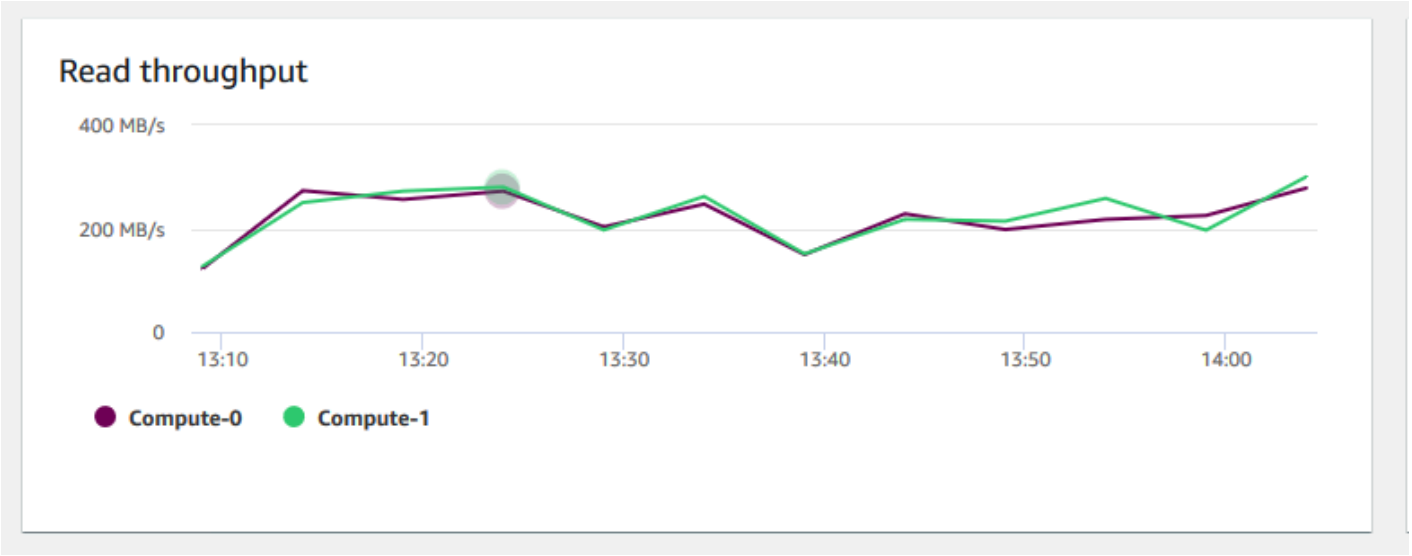
- メンテナンスモード – 選択した時間にクラスターがメンテナンスモードになっているかどうかを、On インジケータと Off インジケータで示します。クラスターがメンテナンス中である時間を確認できます。次に、この時間をクラスターに対して実行されたオペレーションと関連付けて、定期的なイベントの将来のダウンタイムを推定できます。



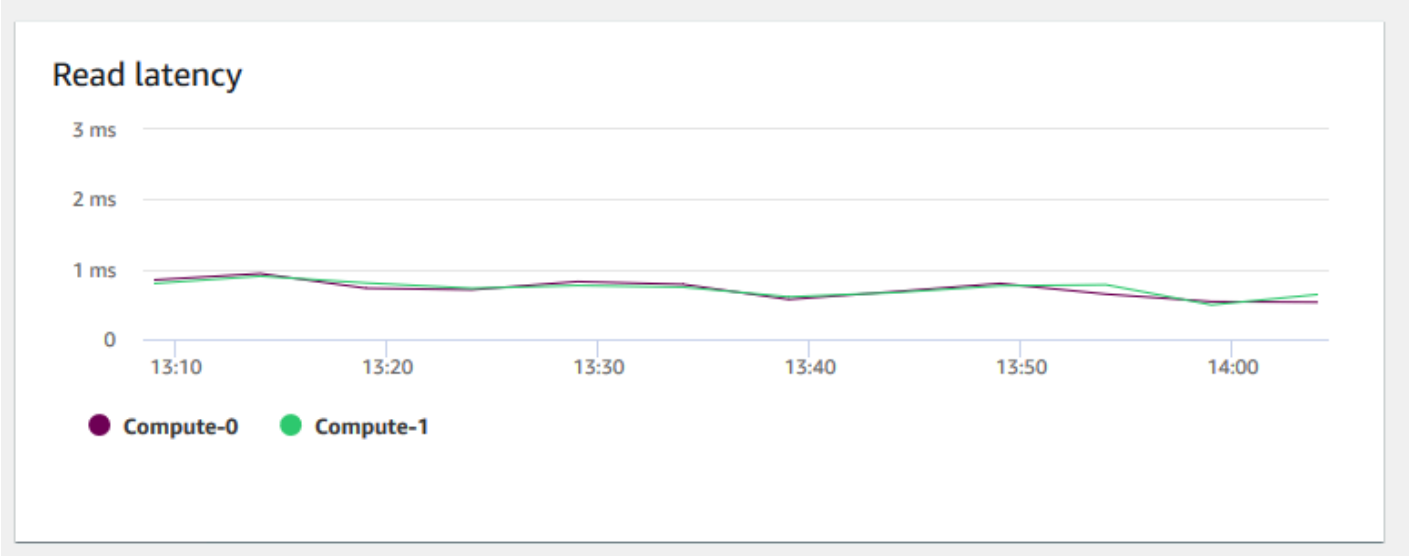
- ディスク容量使用率 – クラスター全体ではなく、コンピューティングノードあたりのディスク容量の使用率を示します。このグラフを表示して、ディスク使用率をモニタリングできます。VACUUM や COPY などのメンテナンスオペレーションでは、ソートオペレーションに中間の一時ストレージ領域が使用されるため、ディスク使用率のスパイクが予想されます。



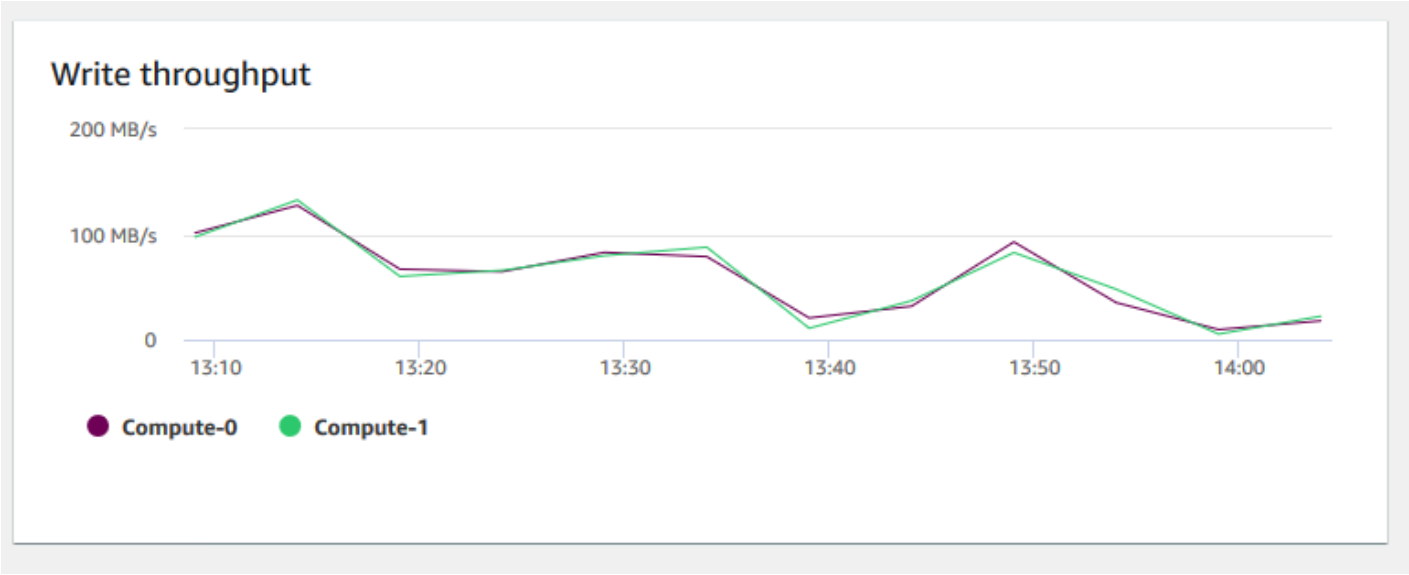
- 読み込みスループット – ディスクから読み取られた 1 秒あたりの平均メガバイト数を示します。このグラフを評価して、クラスターの対応する物理側面をモニタリングできます。クラスターとそのボリューム内のインスタンス間のネットワークトラフィックは、このスループットに含まれません。



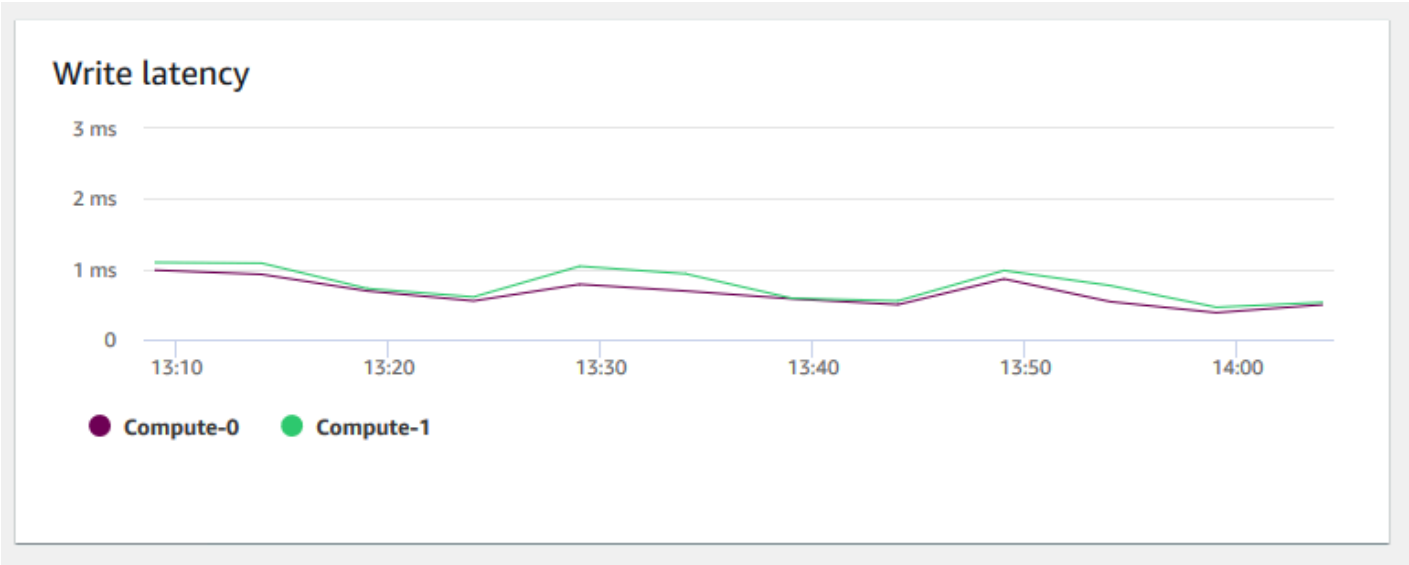
- 読み込みレイテンシー – ディスク読み取り I/O オペレーションにかかった平均時間 (ミリ秒単位) を示します。返されるデータの応答時間を表示できます。レイテンシーが高い場合、送信側がアイドル時間 (新しいパケットを送信しない時間) に費やす時間が長くなり、スループットが低下します。



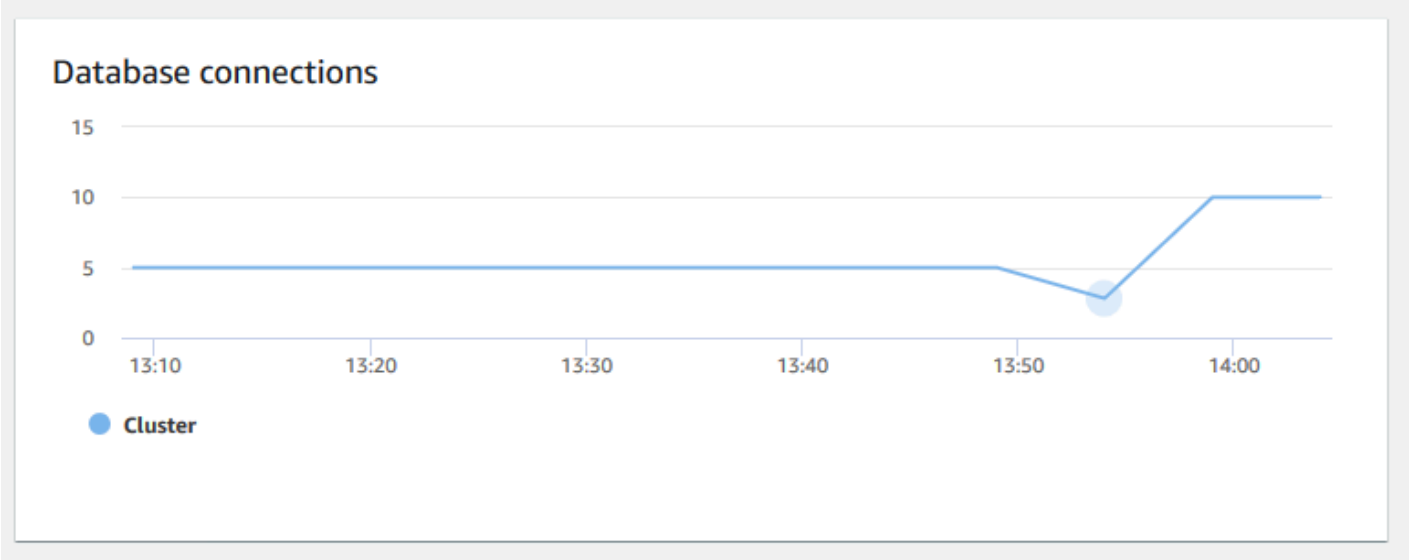
- 書き込みスループット – ディスクに書き込まれた 1 秒あたりの平均メガバイト数を示します。このメトリクスを評価して、クラスターの対応する物理側面をモニタリングできます。クラスターとそのボリューム内のインスタンス間のネットワークトラフィックは、このスループットに含まれません。



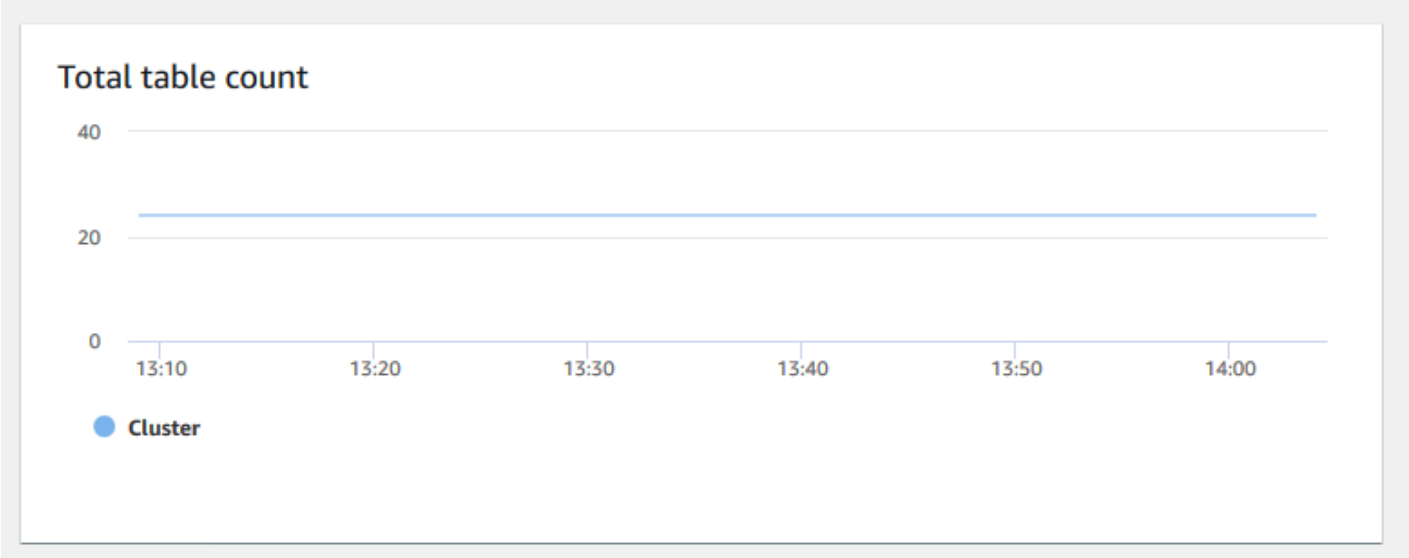
- 書き込みレイテンシー – ディスク書き込み I/O オペレーションにかかった平均時間 (ミリ秒単位) を示します。書き込み確認応答が返されるまでの時間を評価できます。レイテンシーが高い場合、送信側がアイドル時間 (新しいパケットを送信しない時間) に費やす時間が長くなり、スループットが低下します。



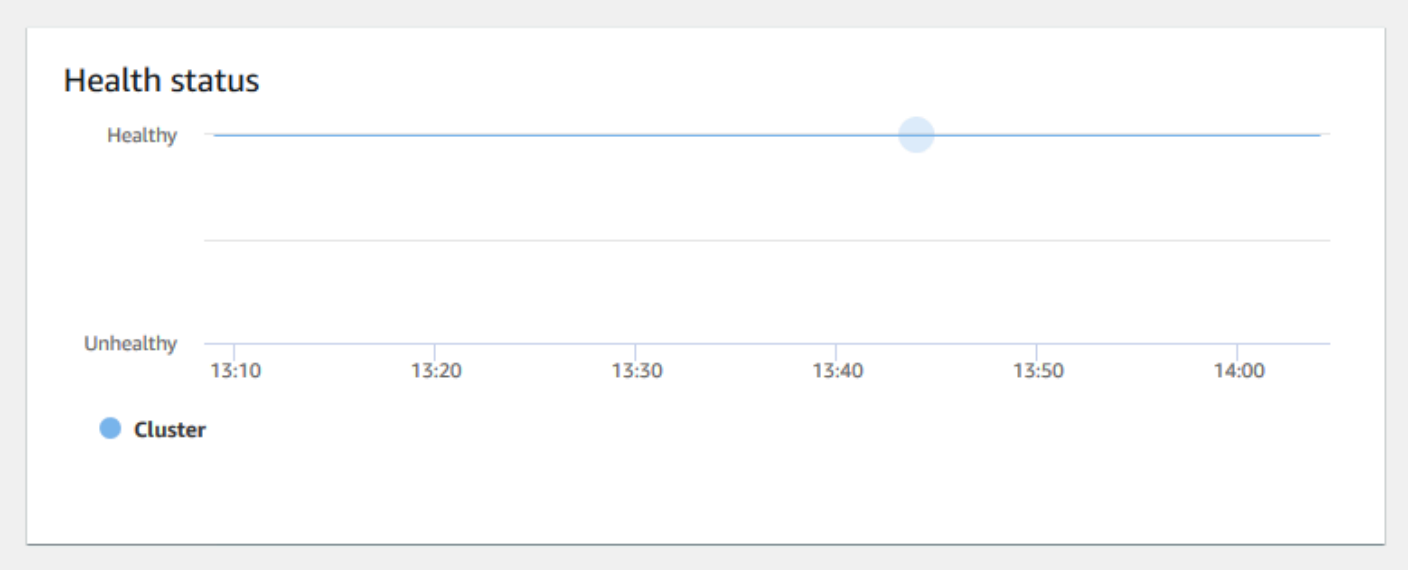
- データベース接続 – クラスターへのデータベース接続の数を示します。このグラフを使用して、データベースに対して確立されている接続数を確認し、クラスターの使用率が最も低い時間を見つけることができます。



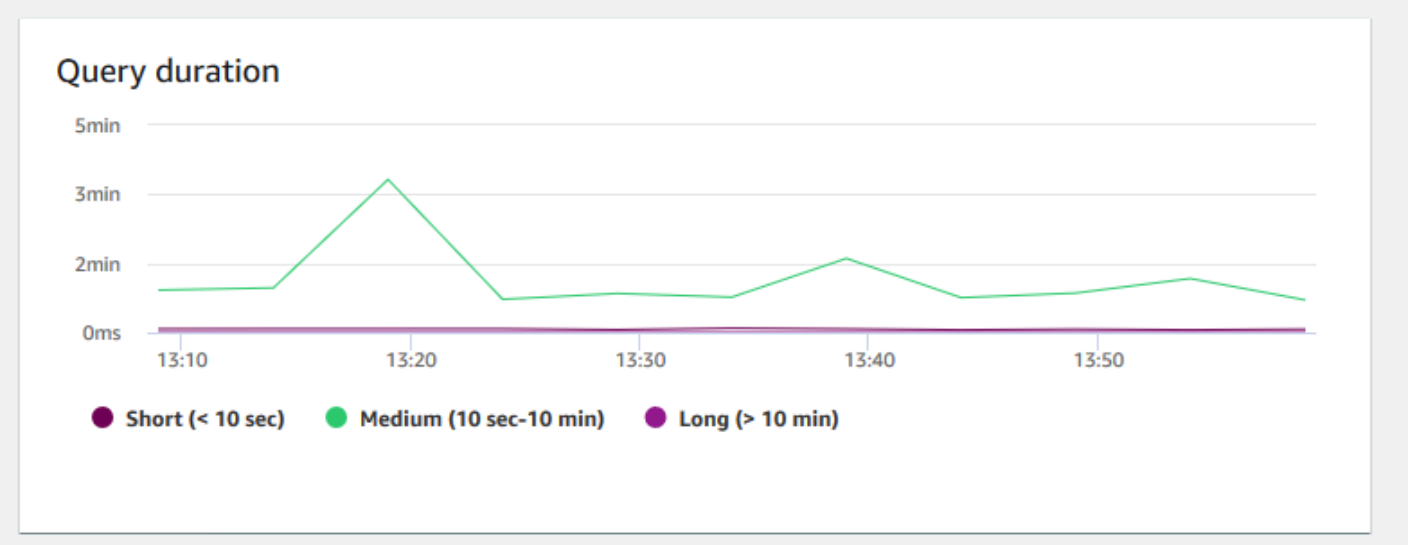
- 合計テーブル数 – 特定の時点でクラスター内で開いているユーザーテーブルの数を示します。開いているテーブルの数が多い場合は、クラスターのパフォーマンスをモニタリングできます。



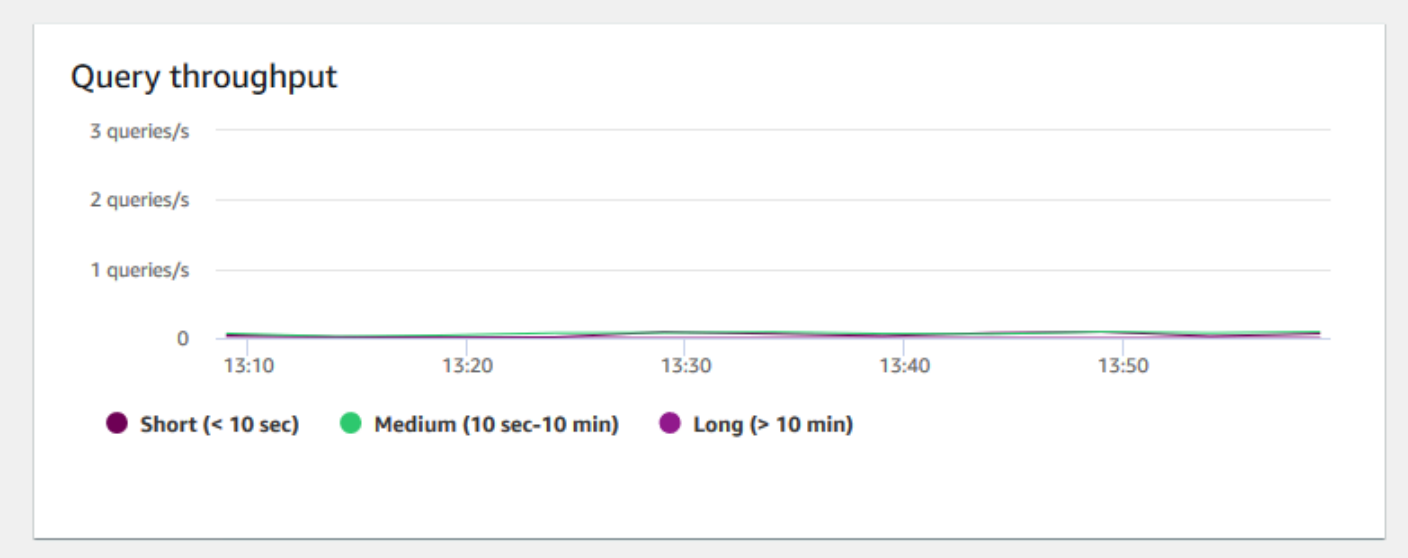
- ヘルスステータス – クラスターのヘルスを Healthy または Unhealthy で示します。クラスターがデータベースに接続でき、単純なクエリを正常に実行できる場合、クラスターは正常であると見なされます。それ以外の場合、クラスターは正常な状態ではありません。正常でない状態が発生する可能性があるのは、クラスターデータベースに極端に高い負荷がかかっているか、クラスター上のデータベースに設定の問題がある場合です。



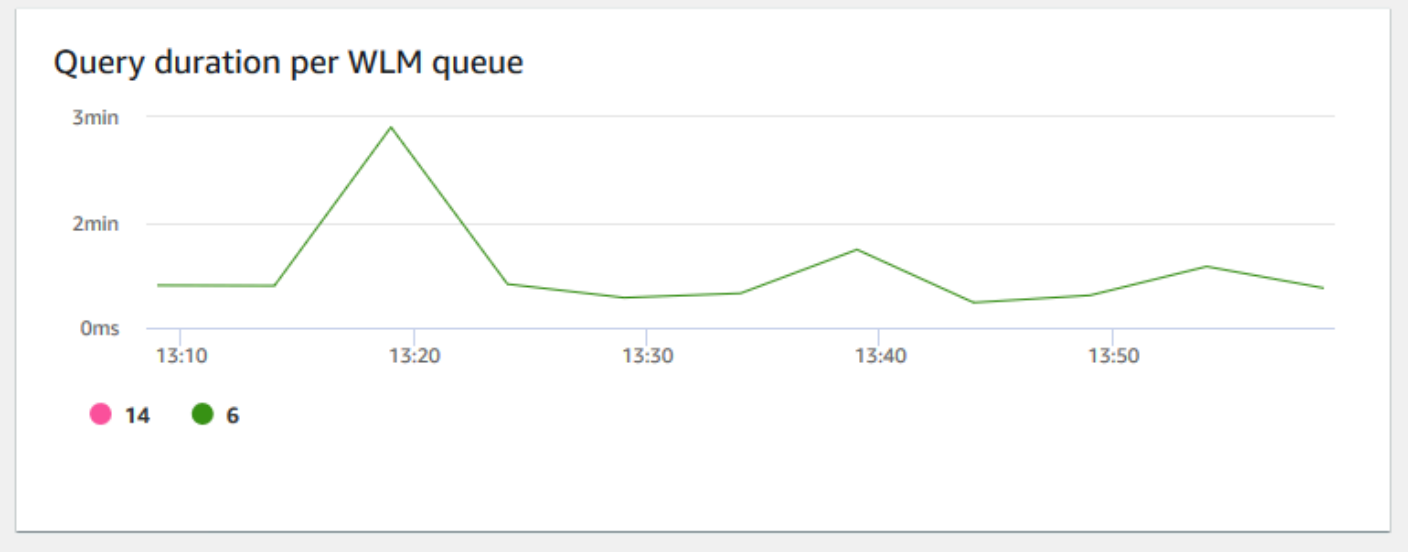
- クエリ期間 – クエリを完了するまでの平均時間 (マイクロ秒単位) を示します。このグラフのデータをベンチマークして、クラスター内の I/O パフォーマンスを測定し、最も時間のかかるクエリを必要に応じて調整できます。



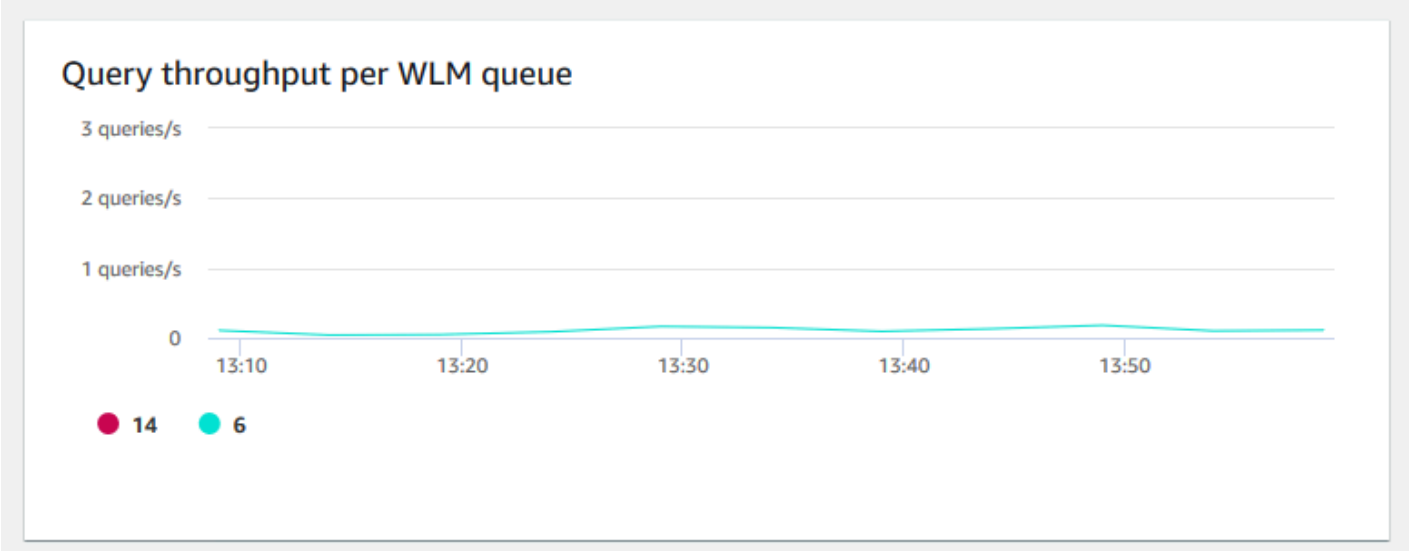
- クエリスループット – 1 秒あたりの完了済みクエリの平均数を示します。このグラフ上のデータを分析して、データベースのパフォーマンスを測定し、バランスのとれた方法でマルチユーザーのワークロードに対応するシステム能力を明らかにできます。



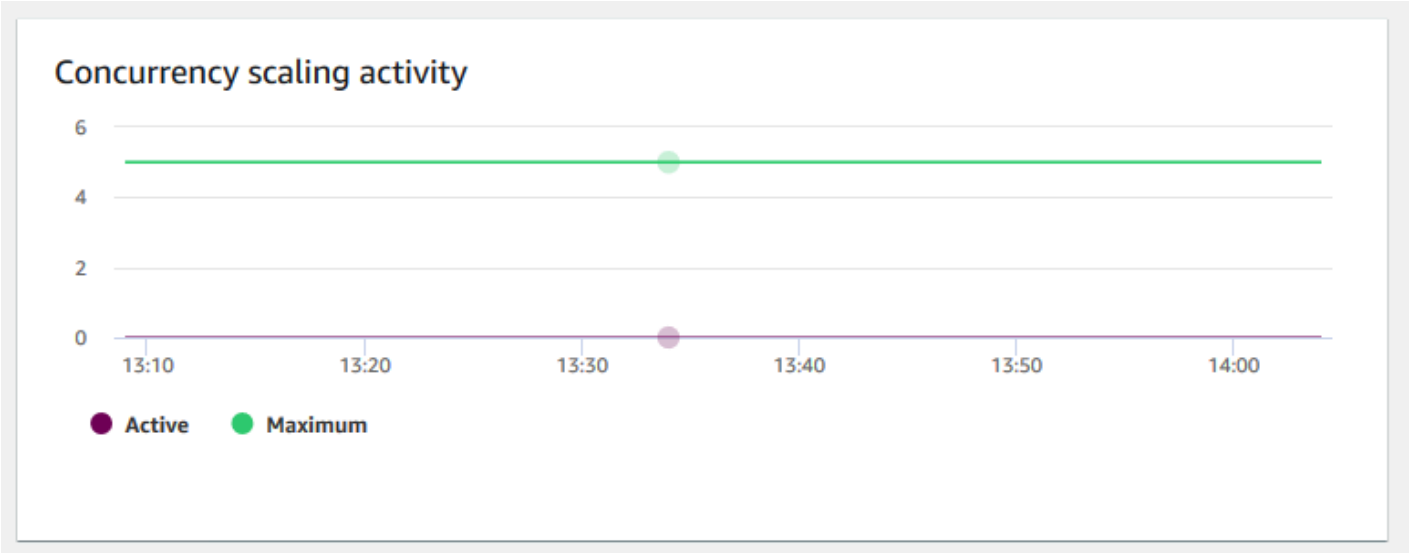
- WLM キューあたりのクエリの期間 – クエリを完了するまでの平均時間 (マイクロ秒単位) を示します。このグラフのデータをベンチマークして、WLM キューあたりの I/O パフォーマンスを測定し、最も時間のかかるクエリを必要に応じて調整できます。



- WLM キューあたりのクエリスループット – 1 秒あたりの完了済みクエリの平均数を示します。このグラフのデータを分析して、WLM キューごとのデータベースパフォーマンスを測定できます。



- 同時実行スケーリングアクティビティ – アクティブな同時実行スケーリングクラスターの数を表示します。同時実行スケーリングが有効になっていると、同時読み込みクエリの増加を処理する必要がある場合、Amazon Redshift は自動的に追加のクラスター容量を追加します。



クエリ履歴データの表示

Amazon Redshift のクエリ履歴メトリクスを使用して、次の操作を実行できます。

- クエリのパフォーマンスの問題を分離し、診断します。
- クエリのランタイムメトリクスとクラスターのパフォーマンスメトリクスを同じタイムラインで比較して、2 つの関連性を確認します。この操作は、パフォーマンスが低いクエリの特定、ボトル

ネックとなっているクエリの検索、およびワークロードに対してクラスターのサイズ変更が必要かどうかの判断に役立ちます。

- タイムラインで選択して、特定のクエリの詳細にドリルダウンします。クエリ ID およびその他のプロパティがグラフの下の行に表示されている場合、クエリを選択してクエリの詳細を表示できます。詳細として、クエリの SQL ステートメント、実行の詳細、クエリプランなどが表示されます。詳細については、クエリの詳細の表示と分析 を参照してください。
- ロードジョブが正常に完了し、サービスレベルアグリーメント (SLA) を満たしているかどうかを確認します。

クエリ履歴データを表示するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [クラスター] を選択し、リストからクラスター名を選択してその詳細を開きます。クラスターの詳細が、[クラスターのパフォーマンス] タブ、[クエリのモニタリング] タブ、[データベース] タブ、[データ共有] タブ、[スケジュール] タブ、[メンテナンス] タブ、[プロパティ] タブなどに表示されます。
3. クエリに関するメトリクスの [クエリのモニタリング] タブを選択します。
4. [Query monitoring (クエリのモニタリング)] セクションで、[Query history (クエリの履歴)] タブを選択します。

   ウィンドウのコントロールを使用して、[Query list (クエリリスト)] と [Cluster metrics (クラスターメトリクス)] を切り替えることができます。

   [Query list (クエリリスト)] を選択すると、タブには次のグラフが表示されます。

   - クエリランタイム – タイムライン上のクエリのアクティビティ。このグラフを使用して、どのクエリが同じ期間で実行されているかを確認します。クエリを選択すると、クエリ実行の詳細が表示されます。X 軸は、選択した期間を示します。実行、完了、ロードなどによって、グラフ化されたクエリをフィルタリングできます。各バーはクエリを表し、バーの長さはバーの開始から終了までのランタイムを表します。クエリには、SQL データ操作ステートメント (SELECT、INSERT、DELETE など) とロード (COPY など) を含めることができます。デフォルトでは、選択した期間に実行されている最長のクエリの上位 100 件が表示されます。
   - クエリとロード – クラスターで実行されたクエリとロードのリスト。このウィンドウには、クエリが現在実行されている場合は、[Terminate query (クエリを終了)] オプションが表示されます。

[Cluster metrics (クラスターメトリクス)] を選択すると、タブには次のグラフが表示されます。

- クエリランタイム – タイムライン上のクエリのアクティビティ。このグラフを使用して、どのクエリが同じ期間で実行されているかを確認します。クエリを選択すると、クエリ実行の詳細が表示されます。
- CPU 使用率 – リーダーノード別のクラスターの CPU 使用率、およびコンピュートノードの平均数。
- 使用済みストレージ容量 – ストレージ容量の割合。
- アクティブなデータベース接続 – クラスターへのアクティブなデータベース接続の数。

クエリ履歴グラフを操作する場合は、次の点を考慮してください。

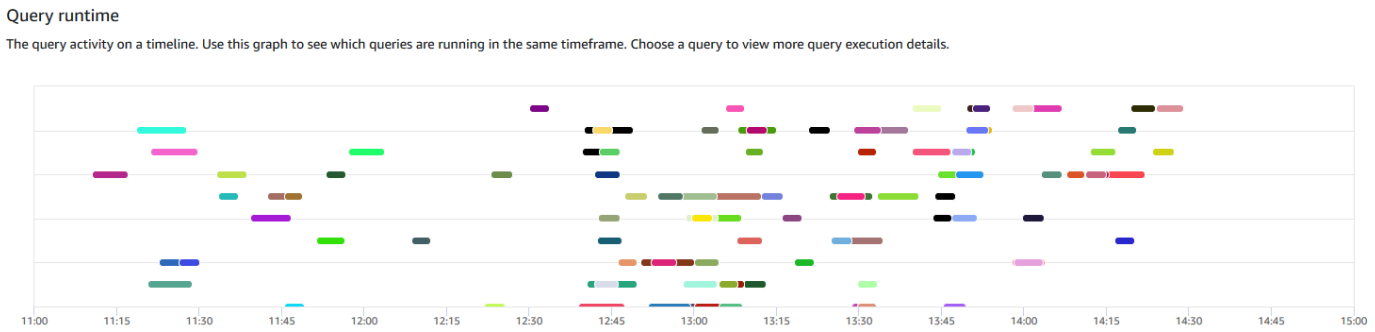
- [Query runtime (クエリランタイム)] グラフで特定のクエリを表すバーを選択すると、そのクエリの詳細が表示されます。また、[Queries and loads (クエリとロード)] リストでクエリ ID を選択して、その詳細を確認することもできます。
- スワイプして [Query runtime (クエリランタイム)] グラフのセクションを選択し、ズームインして特定の期間を表示できます。
- [Query runtime (クエリランタイム)] グラフで、選択したフィルタですべてのデータを対象にするには、[Queries and loads (クエリとロード)] リストに表示されているすべてのページを確認します。
- [Queries and loads (クエリとロード)] リストに表示される列と行数は、設定歯車アイコンによって表示される設定ウィンドウを使用して変更できます。
- [Queries and loads (クエリとロード)] リストは、左側のナビゲータの [クエリ] アイコン、[Queries and loads (クエリとロード)] から移動して表示することもできます。詳細については、クエリとロードの表示 を参照してください。

クエリ履歴グラフ

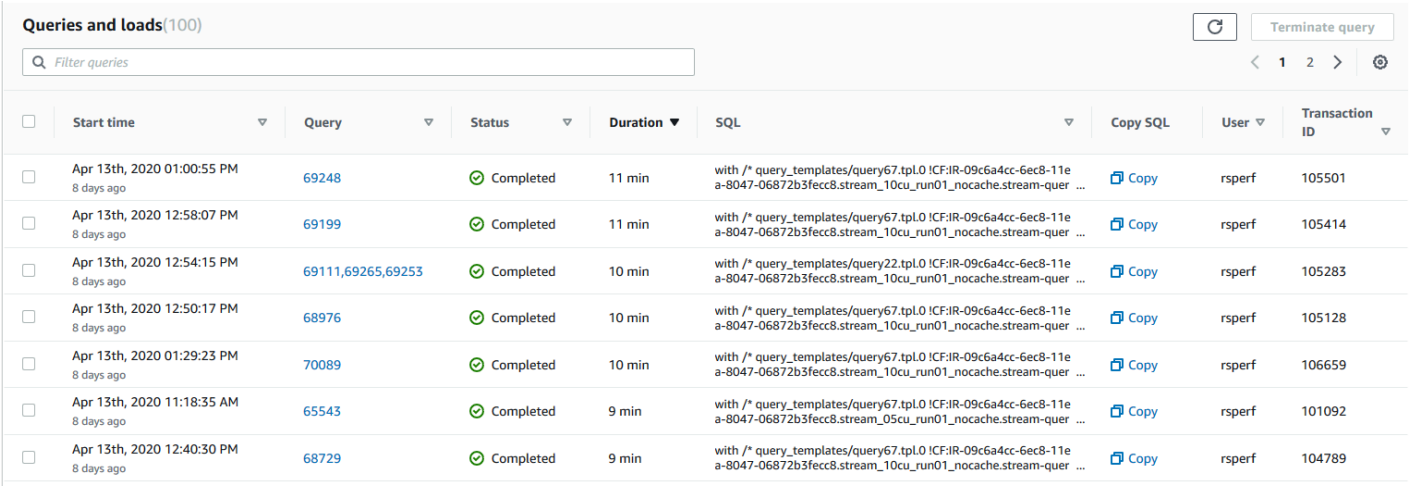
次の例は、新しい Amazon Redshift コンソールに表示されるグラフを示しています。

> **Note**
>
> Amazon Redshift コンソールグラフには、最新の 100,000 個のクエリのデータのみが含まれます。

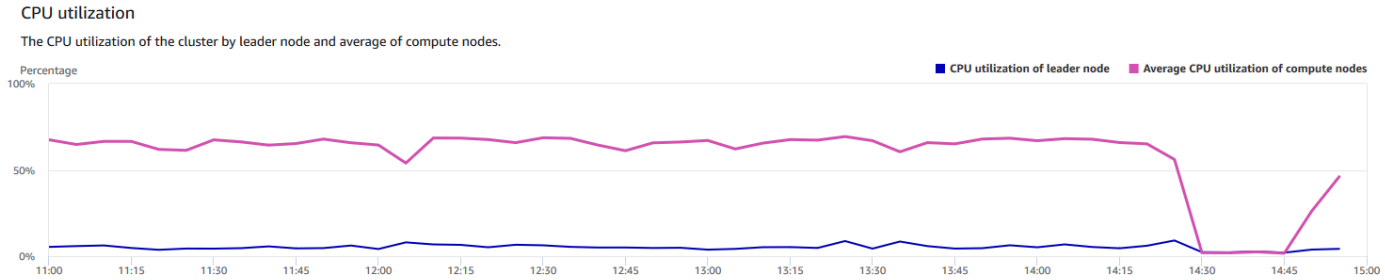
- Query runtime (クエリランタイム



- Queries and loads (クエリとロード



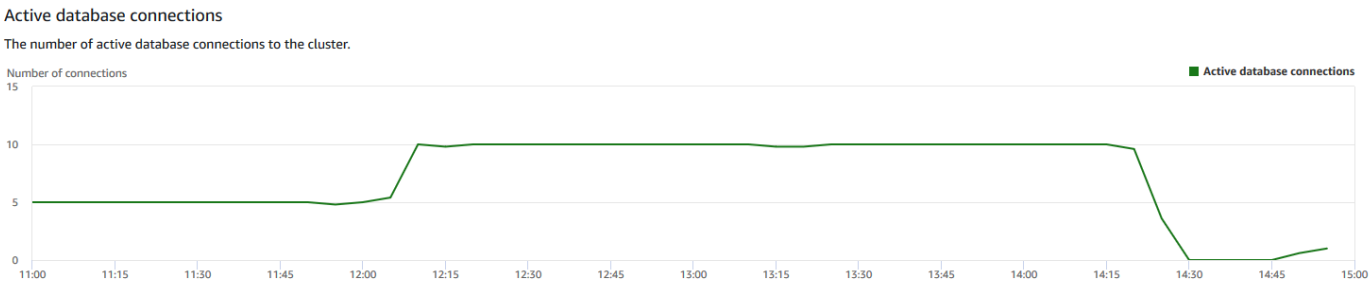
- CPU 使用率



- Storage capacity used (使用済みのストレージキャパシティー



- Active database connections (アクティブなデータベース接続



データベースパフォーマンスデータの表示

Amazon Redshift のデータベースパフォーマンスメトリクスを使用して、次の操作を実行できます。

- 処理ステージ別に、クエリにかかった時間を分析します。ステージでかかった時間の量の異常な傾向を探すことができます。
- クエリの数、期間、およびクエリのスループットを期間範囲 (短期、中期、長期) 別に分析します。
- クエリの優先度 (最低、低、標準、高、最高、重大) 別にクエリ待機時間の傾向を確認します。
- WLM キュー別のクエリ期間、スループット、または待機時間の傾向を調べます。

データベースパフォーマンスデータを表示するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [クラスター] を選択し、リストからクラスター名を選択してその詳細を開きます。クラスターの詳細が表示されます。これには、[クラスターのパフォーマンス]、[クエリのモニタリング]、[データベース]、[データ共有]、[スケジュール]、[メンテナンス]、および [プロパティ] タブなどがあります。

3. クエリに関するメトリクスの [Query monitoring (クエリのモニタリング)] タブを選択します。
4. [Query monitoring (クエリのモニタリング)] セクションで、[Database performance (データベースのパフォーマンス)] タブを選択します。

   ウィンドウのコントロールを使用して、クラスターメトリクスと WLM キューメトリクスを切り替えることができます。

   [Cluster metrics (クラスターメトリクス)] を選択すると、タブには次のグラフが表示されます。

   - ワークロードの実行内訳 – クエリ処理ステージでかかった時間。
   - 期間範囲別のクエリ – 短期、中期、長期のクエリの数。
   - クエリスループット – 1 秒あたりに実行されるクエリの平均数。
   - クエリの期間 – クエリを完了するまでの平均時間。
   - 優先度別の平均キュー待機時間クエリ – クエリ優先度別に WLM キューでの待機にかかったクエリの合計時間。

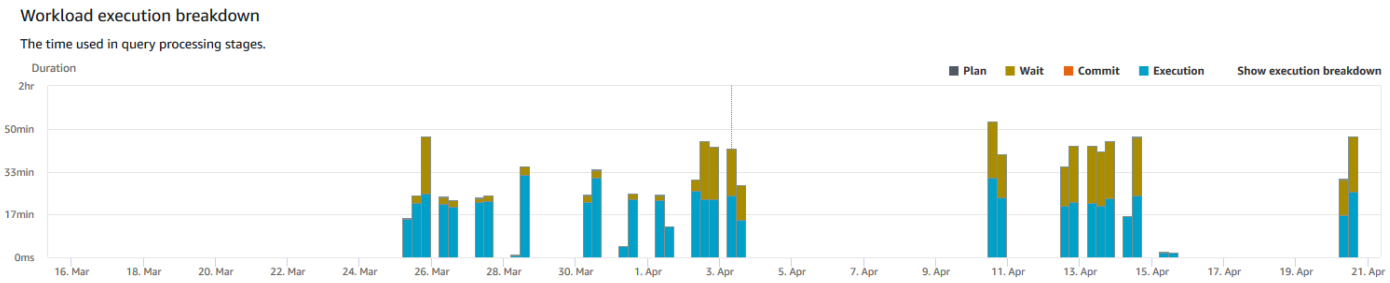
   [WLM queue metrics (WLM キューメトリクス)] を選択すると、タブには次のグラフが表示されます。

   - キュー別のクエリ期間 – WLM キューごとの平均クエリ期間。
   - キュー別のクエリスループット – WLM キューが 1 秒あたりに完了したクエリの平均数。
   - キュー別のクエリ待機時間 – WLM キューの待機にかかったクエリの平均期間。

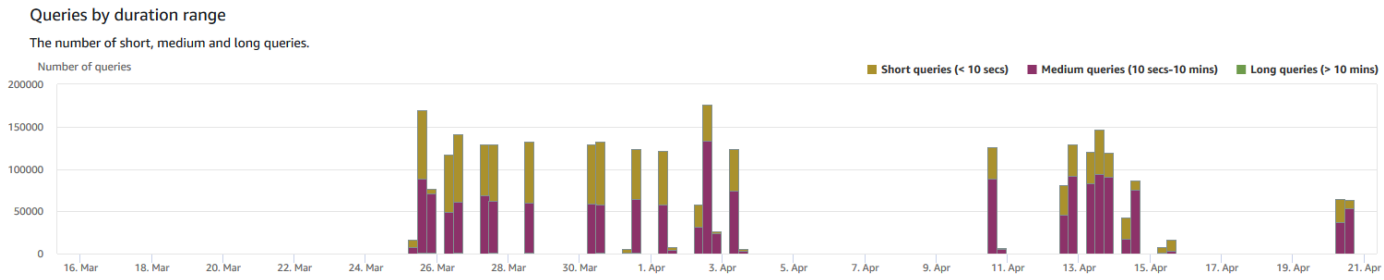
データベースパフォーマンスグラフ

次の例は、新しい Amazon Redshift コンソールに表示されるグラフを示しています。

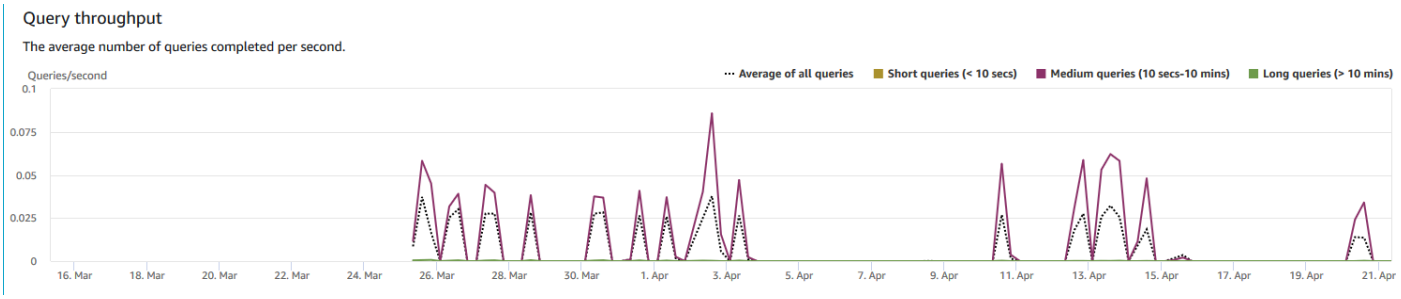
- Workload execution breakdown (ワークロードの実行内訳



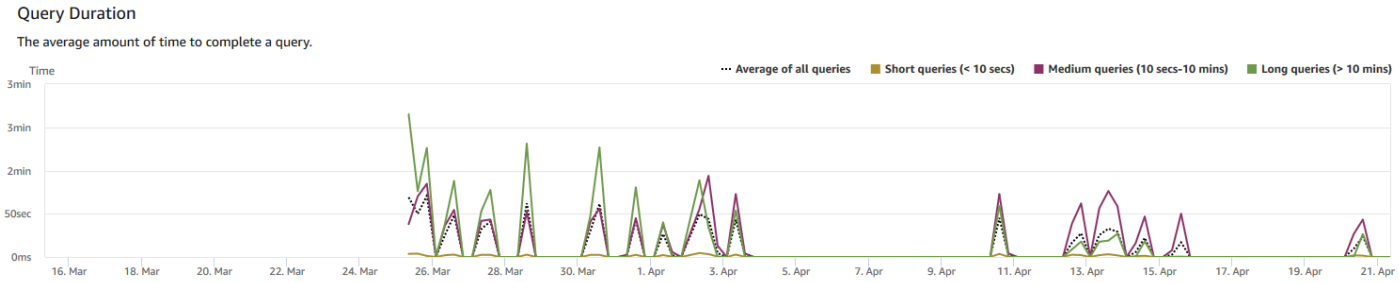
- Queries by duration range (期間範囲別のクエリ



- クエリのスループット



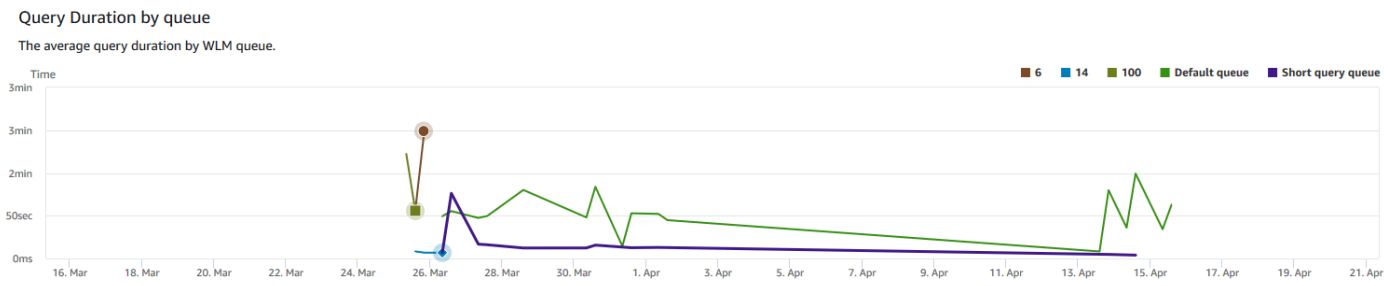
- クエリの期間



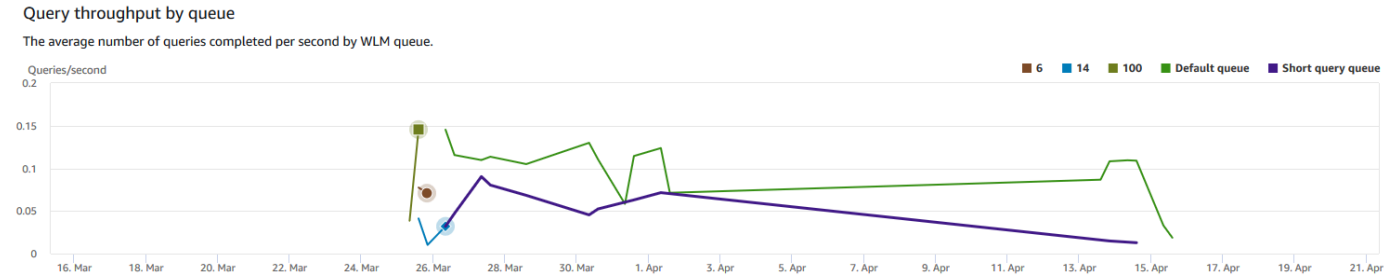
- Average queue wait time by priority (優先度別の平均キュー待機時間



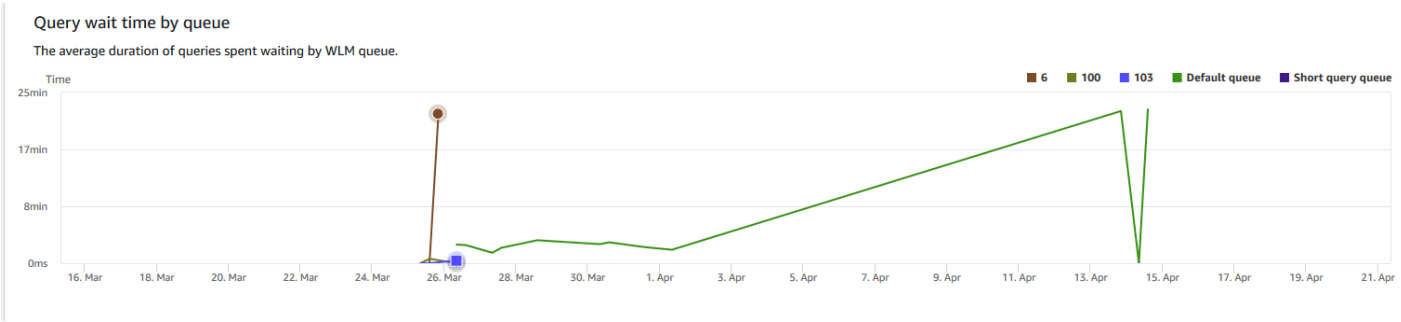
- Query duration by queue (キュー別のクエリ期間



- Query throughput by queue (キュー別のクエリスループット



- Query wait time by queue (キュー別のクエリ待機時間



ワークロードの同時実行と同時実行スケーリングデータの表示

Amazon Redshift で同時実行スケーリングメトリクスを使用すると、次のことを行うことができます。

- 同時実行スケーリングを有効にして、キューに入れられたクエリの数を減らすことができるかどうかを分析します。WLM キュー別、またはすべての WLM キューに対して比較できます。
- 同時実行スケーリングクラスターで同時実行スケーリングアクティビティを表示します。これにより `max_concurrency_scaling_clusters` で同時実行スケーリングが制限されているかどうかがわかります。制限されている場合、DB パラメータの `max_concurrency_scaling_clusters` を増やすことを選択できます。

- すべての同時実行スケーリングクラスターについて合計された同時実行スケーリングの合計使用量を表示します。

同時実行スケーリングデータを表示するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [クラスター] を選択し、リストからクラスター名を選択してその詳細を開きます。クラスターの詳細が、[クラスターのパフォーマンス] タブ、[クエリのモニタリング] タブ、[データベース] タブ、[データ共有] タブ、[スケジュール] タブ、[メンテナンス] タブ、[プロパティ] タブなどに表示されます。
3. クエリに関するメトリクスの [Query monitoring (クエリのモニタリング)] タブを選択します。
4. [Query monitoring (クエリのモニタリング)] セクションで、[Workload concurrency (ワークロードの同時実行)] タブを選択します。

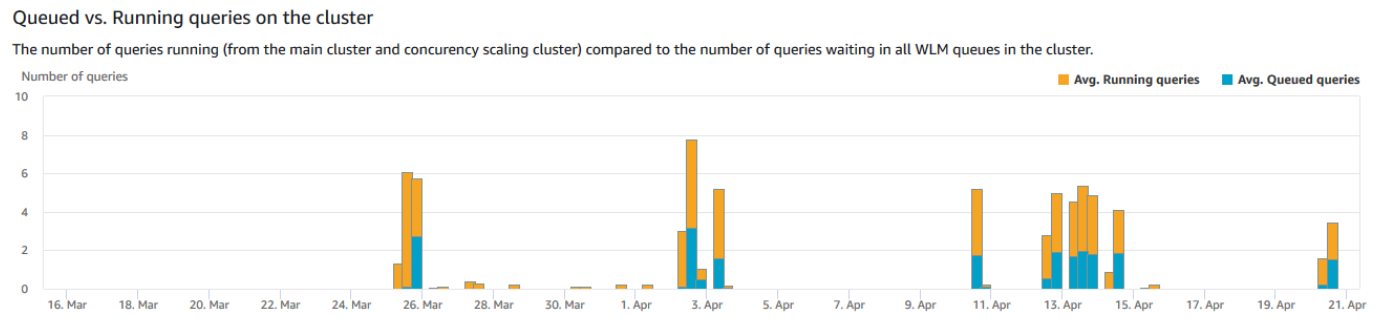
   このタブには、次のグラフがあります。

   - クラスター上のキューと実行中のクエリ – クラスターのすべての WLM キューで待機しているクエリの数と比較した、(メインクラスターおよび同時実行スケーリングクラスターから) 実行中のクエリの数。
   - キューごとのキューイングと実行中のクエリ – 各 WLM キューで待機しているクエリの数と比較した、(メインクラスターおよび同時実行スケーリングクラスターから) 実行中のクエリの数。
   - 同時実行スケーリングアクティビティ – クエリをアクティブに処理している同時実行スケーリングクラスターの数。
   - 同時実行スケーリングの使用 – アクティブなクエリ処理アクティビティを実行中の同時実行スケーリングクラスターの使用量。

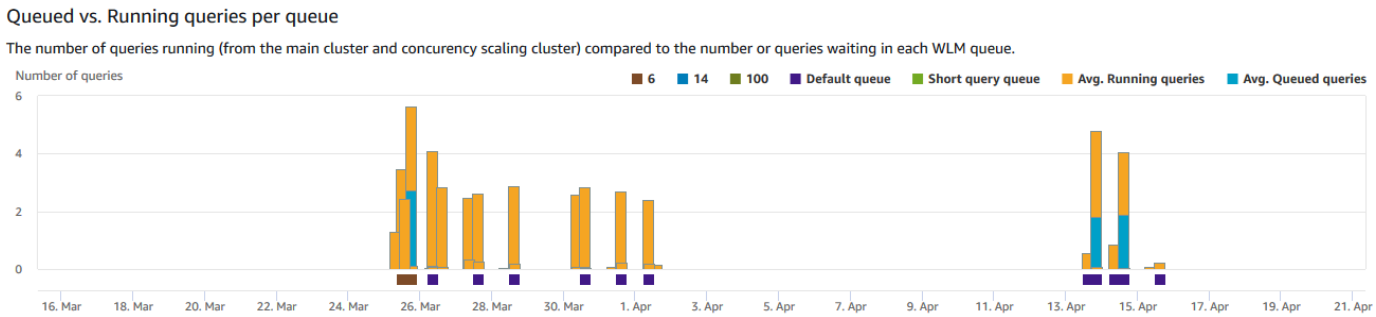
ワークロードの同時実行グラフ

次の例は、新しい Amazon Redshift コンソールに表示されるグラフを示しています。Amazon CloudWatch で同様のグラフを作成するには、同時実行スケーリングと WLM CloudWatch メトリクスを使用できます。Amazon Redshift の CloudWatch メトリクスの詳細については、「Amazon Redshift のパフォーマンスデータ」を参照してください。

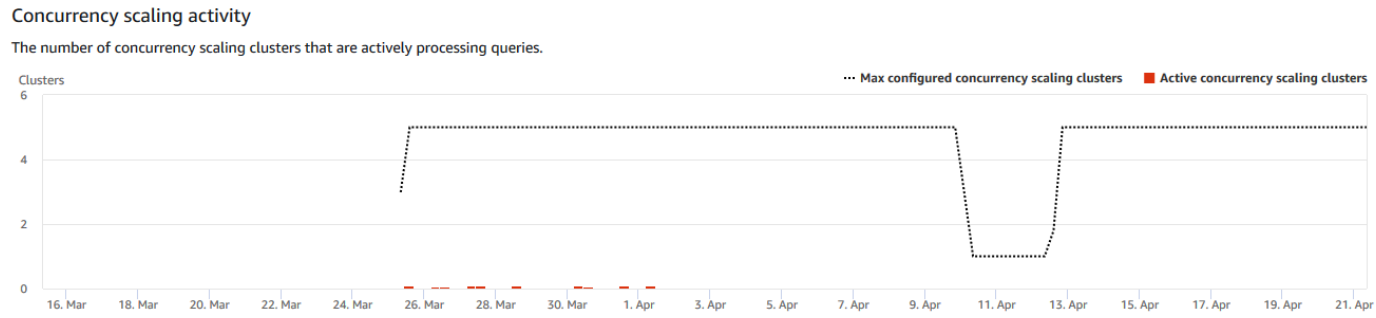
- クラスター上のキューと実行中のクエリ



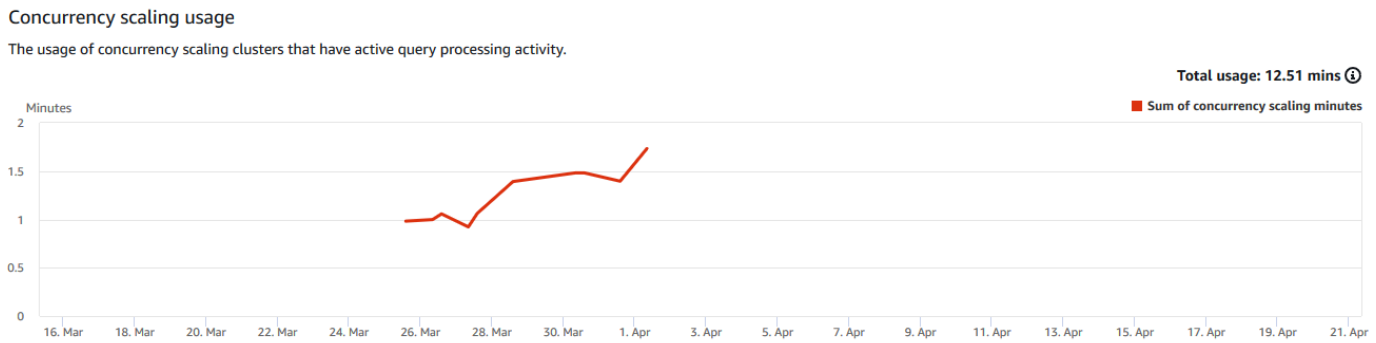
- キューごとのキューイングと実行中のクエリ



- 同時実行スケーリングアクティビティ



- Concurrency scaling usage (同時実行スケーリング使用率



自動最適化データの表示

Amazon Redshift コンソールは、追加のコンピューティングリソースを使用して実行される自動最適化または自律管理に関する情報を提供します。この情報を使用して使用状況を追跡し、使用制限に達したかどうかをモニタリングできます。Amazon Redshift は、プロビジョニングされたクラスター自体で実行される自律管理に対して課金しませんが、追加のコンピューティングリソースを使用して実行される自律管理に対しては課金します。詳細については、「Amazon Redshift データベースデベロッパーガイド」の「データベースの自動最適化のための追加のコンピューティングリソースの割り当て」を参照してください。

追加のコンピューティングの自律管理データを表示するには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [Clusters] (クラスター) を選択し、リストからクラスターの名前を選択してその詳細を開きます。
3. クラスターの詳細ページで、[アクション] ドロップダウンメニューから [使用制限を管理] を選択します。クラスターの [メンテナンス] タブを選択し、下にスクロールして [使用制限を作成] を選択することもできます。
4. 追加のコンピューティングの自律管理データを示すグラフは、[自動最適化のための追加のコンピューティングの使用制限] というセクションの下に表示されます。グラフには、Amazon Redshift が特定の期間に追加のコンピューティングリソースを使用して自律管理を実行する時間が表示されます。

クエリとロードの表示

Amazon Redshift コンソールには、データベースで実行されるクエリとロードに関する情報が表示されます。この情報を使用して、処理に長い時間がかかるクエリや、他のクエリの効率的な処理を妨げるようなボトルネックを作成するクエリを識別し、トラブルシューティングすることができます。Amazon Redshift コンソールのクエリ情報を使用して、クエリ処理をモニターすることができます。

クエリパフォーマンスデータを表示するには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。

2. ナビゲーションメニューで、[Queries and loads] (クエリおよびロード) を選択し、アカウントのクエリのリストを表示します。

   デフォルトでは、全クラスターの過去 24 時間のクエリがリストに表示されます。コンソールで表示日付の範囲を変更できます。

   > **Important**
   >
   > [Queries and loads (クエリおよびロード)] リストには、システムで最も長く実行されている最大 100 件のクエリが表示されます。

クエリの詳細の表示と分析

クエリ識別子を使用して、クエリの詳細を表示できます。詳細として、クエリの完了ステータス、期間、SQL ステートメントなどの他に、ユーザクエリであるか、Amazon Redshift によって書き換えられたクエリであるかが表示されます。ユーザークエリは、SQL クライアントから生成された、またはビジネスインテリジェンスツールによって生成された、Amazon Redshift に送信されるクエリです。Amazon Redshift はクエリを書き換えて最適化することがあり、これにより複数のクエリが書き換えられる可能性があります。この処理は Amazon Redshift によって実行されますが、クエリの詳細ページには、ユーザークエリとともに書き換えられたクエリが表示されます。

クエリを表示するには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Queries and loads] (クエリおよびロード) を選択し、アカウントのクエリのリストを表示します。クエリを見つけるため、このページの設定を変更する必要がある場合があります。
3. リストで [クエリ] 識別子を選択して、[クエリの詳細] を表示します。

   [クエリの詳細] ページには、[クエリの詳細] とクエリのメトリクスが表示された [クエリプラン] タブが含まれています。

   メトリクスには、クエリの開始時刻、クエリ ID、ステータス、期間などの詳細が表示されます。その他の詳細としては、クエリがメインクラスターと同時実行スケーリングクラスターのどちらで実行されたか、クエリが親クエリと書き換えられたクエリのどちらであるかが示されます。

クエリの実行中にクラスターパフォーマンスを表示するには

クエリの実行時にクラスターのパフォーマンスをモニタリングして、潜在的なボトルネックを特定し、クエリの実行を最適化できます。クエリの実行時にクラスターのパフォーマンスを表示すると、CPU 使用率、ディスク I/O、ネットワークトラフィックなどのシステムレベルのメトリクスと、実行時間、処理されたデータ、クエリステップなどのクエリレベルの詳細がリアルタイムで表示されます。次の手順では、パフォーマンスメトリクスにアクセスして解釈し、プロビジョニングされたクラスターを効果的に管理および最適化する方法について説明します。

クエリの実行中にクラスターパフォーマンスを表示するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [クラスター] を選択し、リストからクラスター名を選択してその詳細を開きます。クラスターの詳細が、[クラスターのパフォーマンス] タブ、[クエリのモニタリング] タブ、[データベース] タブ、[データ共有] タブ、[スケジュール] タブ、[メンテナンス] タブ、[プロパティ] タブなどに表示されます。
3. 詳細については、[Query monitoring (クエリのモニタリング)] タブを選択します。

   詳細については、「クエリ履歴データの表示」を参照してください。

ロード操作中にクラスターメトリクスを表示する

ロードオペレーション中にクラスターのパフォーマンスを表示すると、リソースを消費しているクエリを特定し、その影響を軽減するためのアクションを実行できます。ロードを最後まで実行しない場合はロードを終了できます。

> **Note**
>
> Amazon Redshift コンソールでクエリとロードを終了するには、特定のアクセス権限が必要です。ユーザーがクエリとロードを終了できるようにする場合は、必ず `redshift:CancelQuerySession` アクションを AWS Identity and Access Management (IAM) ポリシーに追加します。この要件は、Amazon Redshift 読み取り専用 AWS 管理ポリシーを選択するか、IAM でカスタムポリシーを作成するかにかかわらず適用されます。Amazon Redshift フルアクセスポリシーが適用されているユーザーには、クエリとロードを終了するためのアクセス許可がすでにあります。Amazon Redshift の IAM ポリシーにおけるアクションの詳細については、「リソースへのアクセスの管理」を参照してください。

ロード操作中にクラスターパフォーマンスを表示するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [クラスター] を選択し、リストからクラスター名を選択してその詳細を開きます。クラスターの詳細が、[クラスターのパフォーマンス] タブ、[クエリのモニタリング] タブ、[データベース] タブ、[データ共有] タブ、[スケジュール] タブ、[メンテナンス] タブ、[プロパティ] タブなどに表示されます。
3. 詳細については、[クエリのモニタリング] タブを選択します。
4. [Queries and loads (クエリとロード)] セクションで、[ロード] をセクションしてクラスターのロード操作を表示します。ロードが実行中の場合、[クエリの終了] を選択することにより終了させることができます。

クラスターワークロードの内訳表の表示

コンソールのワークロードの実行内訳表を確認して、ワークロードのパフォーマンスの詳細を表示できます。この表は、QueryRuntimeBreakdown メトリクスで提供されるデータを使用して構成されています。この表では、待機や計画などのさまざまな処理ステージで、クエリにどれだけの時間がかかっているかを見ることができます。

> **Note**
>
> ワークロードの実行内訳表は、単一ノードのクラスターには表示されません。

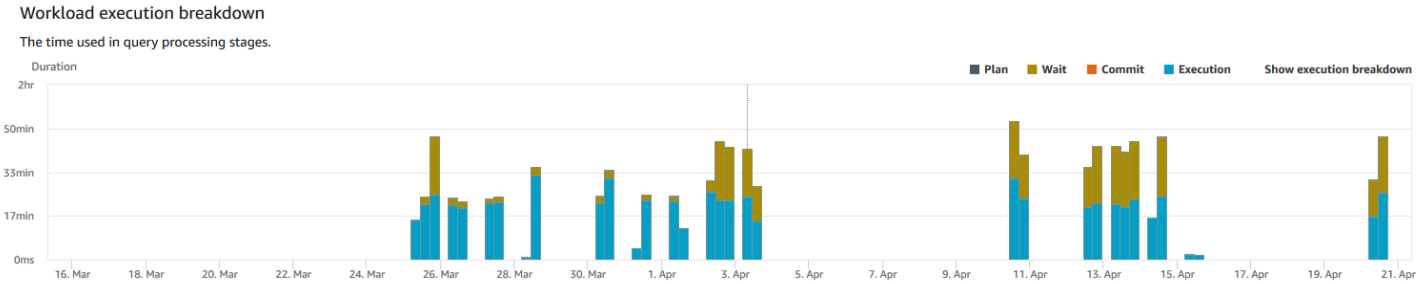
次のメトリクスのリストでは、さまざまな処理ステージを説明しています。

- `QueryPlanning`: SQL ステートメントの解析と最適化にかかった時間。
- `QueryWaiting`: ワークロード管理 (WLM) キューでの待機にかかった時間。
- `QueryExecutingRead`: 読み取りクエリの実行にかかった時間。
- `QueryExecutingInsert`: 挿入クエリの実行にかかった時間。
- `QueryExecutingDelete`: 削除クエリの実行にかかった時間。
- `QueryExecutingUpdate`: 更新クエリの実行にかかった時間。
- `QueryExecutingCtas`: CREATE TABLE AS クエリの実行にかかった時間。
- `QueryExecutingUnload`: アップロードクエリの実行にかかった時間。

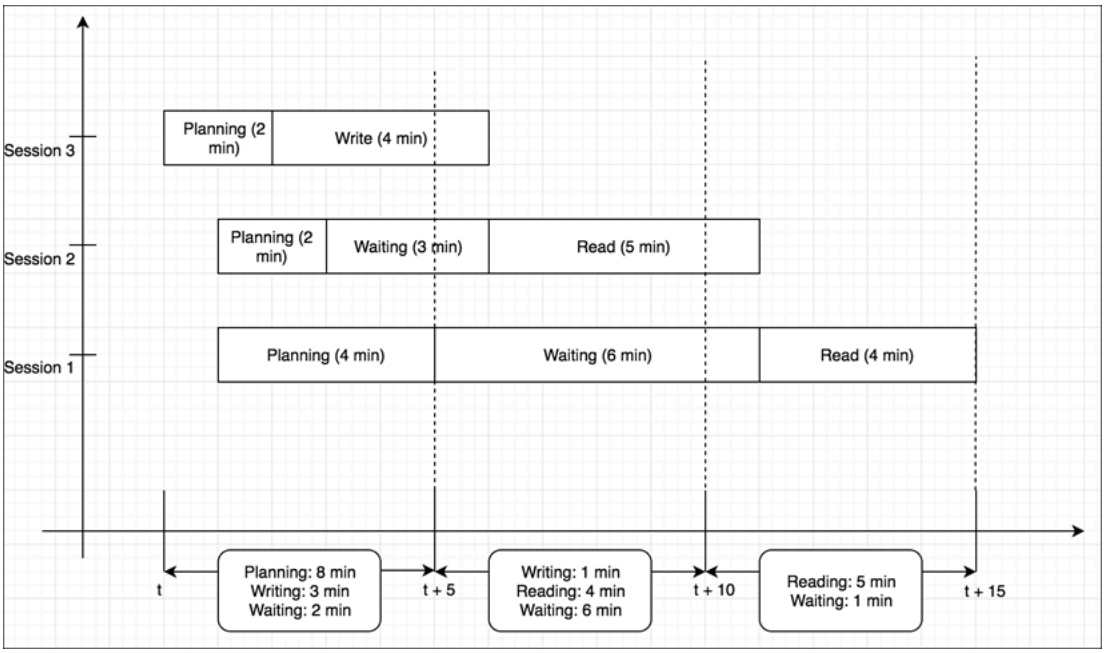
- QueryExecutingCopy: コピークエリの実行にかかった時間。

たとえば、Amazon Redshift コンソールの次のグラフには、計画、待機、読み取り、および書き込みの各段階でクエリにかかった時間が示されます。このグラフの結果を、この先の分析のために他のメトリクスと組み合わせることができます。一部のケースでは、短い期間のクエリ (QueryDuration メトリクスによって測定) が待機時間に多くの時間をかけていると表示されることがあります。このような場合には、特定のキューの WLM 同時実行率を上げることで、スループットを増大させることができます。

次に、ワークロードの実行内訳表の例を示します。チャートの y 軸の値は、積み上げ棒グラフとして示された、指定時刻における各ステージの平均期間です。



次の図は、Amazon Redshift がどのように同時セッションの集計クエリ処理を行うかを示しています。



クラスターワークロードの内訳表を表示するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [クラスター] を選択し、リストからクラスター名を選択してその詳細を開きます。クラスターの詳細が、[クラスターのパフォーマンス] タブ、[クエリのモニタリング] タブ、[データベース] タブ、[データ共有] タブ、[スケジュール] タブ、[メンテナンス] タブ、[プロパティ] タブなどに表示されます。
3. クエリに関するメトリクスの [Query monitoring (クエリのモニタリング)] タブを選択します。
4. [Query monitoring (クエリのモニタリング)] セクションで、[Database performance (データベースのパフォーマンス)] を選択し、[Cluster metrics (クラスターのメトリクス)] を選択します。

   次のメトリクスは、選択した時間範囲について積み上げ棒グラフで表示されています。

   - プラン時間
   - 待機時間
   - コミット時間
   - 実行時間

クエリ実行の分析

クエリの実行の詳細を分析して、クエリがどのように実行されたかを理解し、最適化の対象となる可能性のある領域を特定できます。クエリを分析すると、関連するステップ、各ステップにかかる時間、処理されるデータ量など、クエリプランに関するインサイトが得られます。一般的なユースケースには、実行速度の遅いクエリのトラブルシューティング、データ分散戦略の最適化、クエリの書き換えやインデックス作成の機会の特定などがあります。

クエリを分析するには

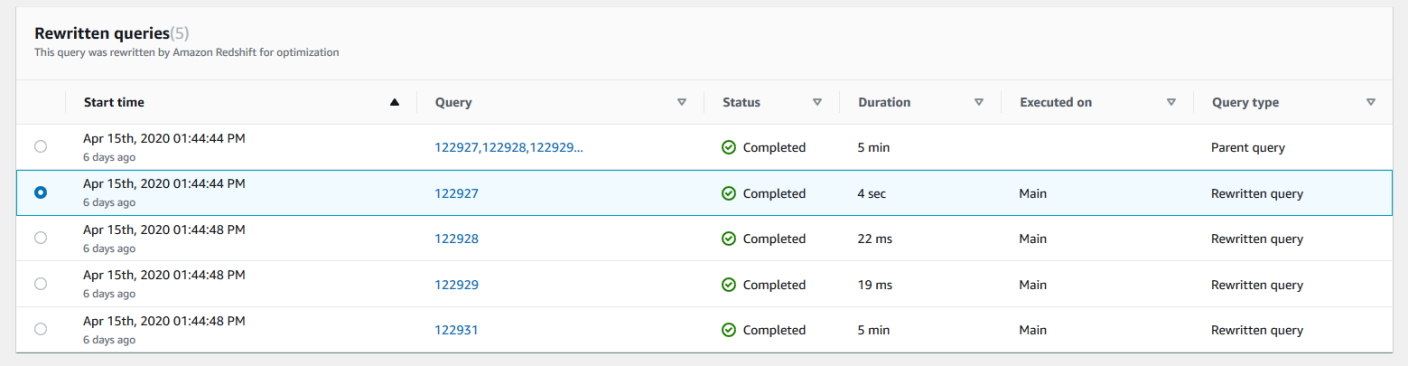
1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Queries and loads] (クエリおよびロード) を選択し、アカウントのクエリのリストを表示します。クエリを見つけるため、このページの設定を変更する必要がある場合があります。
3. リストで [クエリ] 識別子を選択して、[クエリの詳細] を表示します。

[クエリの詳細] ページには、[クエリの詳細] とクエリのメトリクスが表示された [クエリプラン] タブが含まれています。

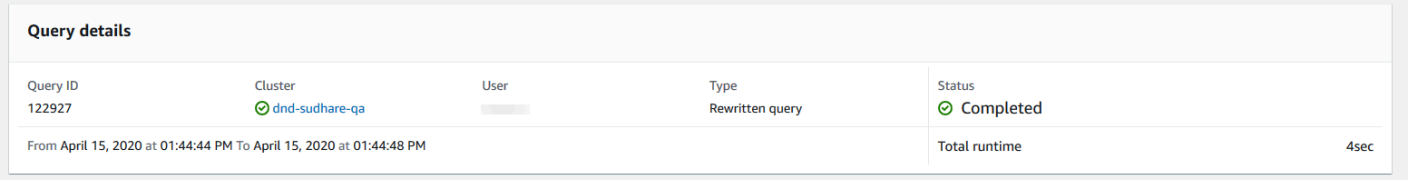
> **Note**
>
> [クエリランタイム] グラフでクエリにドリルダウンすると、[クラスターの詳細] ページの [クエリ履歴] タブから [クエリの詳細] ページに移動することもできます。

[Query details (クエリの詳細)] ページには、次のセクションがあります。

- [Rewritten queries (書き換えクエリ)] のリスト (次のスクリーンショットを参照)。



- [Query details (クエリの詳細)] セクション (次のスクリーンショットを参照)。



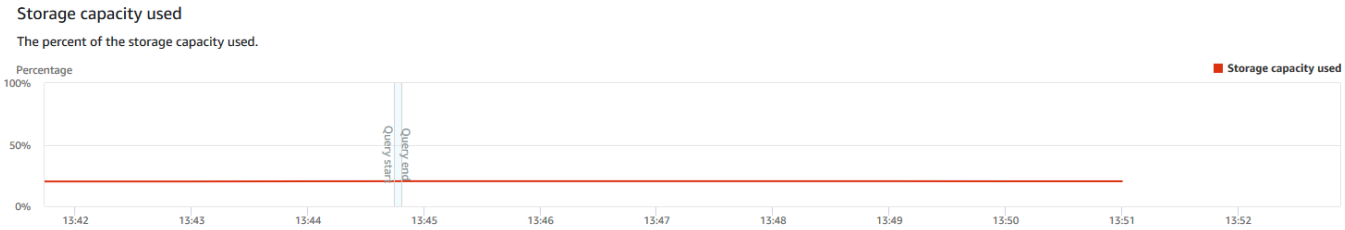
- 実行された SQL と実行に関する実行の詳細を含む [Query details] (クエリの詳細) タブ。
- [Query plan (クエリプラン)] のステップとクエリプランに関するその他の情報を含む [Query plan (クエリプラン)] タブ。このテーブルには、クエリが実行されたときのクラスターに関するグラフも含まれています。
  - Cluster health status (クラスターの健全性ステータス



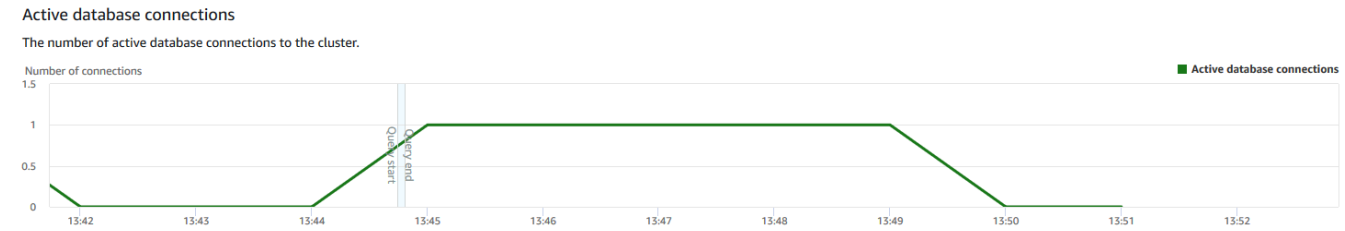
  - CPU 使用率



- Storage capacity used (使用済みのストレージキャパシティー



- Active database connections (アクティブなデータベース接続



アラームを作成する

Amazon Redshift コンソールで作成するアラームは、CloudWatch アラームです。これらのアラームは、クラスターとサーバーレスインスタンスに関する予防的な決定を行うために役立ちます。「Amazon Redshift のパフォーマンスデータ」にリストされているいずれのメトリクスにも、1 つ以上のアラームを設定できます。たとえば、クラスターノードの高い CPUUtilization に対してアラームを設定すると、ノードの使用率が高すぎるときに、そのことが示されます。高 DataStorage のアラームは、サーバーレス名前空間がデータに使用しているストレージ領域を追跡します。

[アクション] から、アラームを変更または削除できます。また、Slack や Amazon Chime のウェブフック URL を指定して、CloudWatch から Slack または Amazon Chime にアラートを送信する Chime アラームまたは Chime アラートを作成できます。

このセクションでは、Amazon Redshift コンソールを使用してアラームを作成する方法について説明します。CloudWatch コンソールを使用するか、AWS CLI や AWS SDK などのメトリクスを操作するその他の方法を使用して、アラームを作成できます。

Amazon Redshift コンソールで CloudWatch アラームを作成するには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。

   Amazon Redshift Serverless を使用している場合は、ダッシュボードの右上にある [Go to Serverless] (サーバーレスに移動) を選択します。
2. ナビゲーションメニューで、[Alarms] (アラーム)、[Create alarm] (アラームの作成) の順に選択します。
3. [Create alarm (アラームの作成)] ページで、プロパティを入力して CloudWatch アラームを作成します。
4. [アラームの作成] を選択します。

実行中クエリの終了

また、[クエリ] ページを使用して、現在進行中のクエリを終了することもできます。

> **Note**
>
> Amazon Redshift コンソールでクエリとロードを終了するには、特定のアクセス権限が必要です。ユーザーがクエリとロードを終了できるようにする場合は、必ず `redshift:CancelQuerySession` アクションを AWS Identity and Access Management (IAM) ポリシーに追加します。この要件は、Amazon Redshift 読み取り専用 AWS マネージドポリシーを選択するか、IAM でカスタムポリシーを作成するかにかかわらず適用されます。Amazon Redshift フルアクセスポリシーが適用されているユーザーには、クエリとロードを終了するためのアクセス許可がすでにあります。Amazon Redshift の IAM ポリシーにおけるアクションの詳細については、「リソースへのアクセスの管理」を参照してください。

実行中クエリを終了するには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Queries and loads] (クエリおよびロード) を選択し、アカウントのクエリのリストを表示します。
3. リストで終了する実行中クエリを選択してから、[クエリの終了] を選択します。

CloudWatch コンソールでのパフォーマンスメトリクス

CloudWatch コンソールで Amazon Redshift メトリクスを使用するときは、次のいくつかの点に留意してください。

- クエリ/ロードパフォーマンスデータを使用できるのは、Amazon Redshift コンソール内のみです。
- CloudWatch の一部のメトリクスには、Amazon Redshift コンソールで使用されるものとは異なる単位があります。たとえば、`WriteThroughput` は GB/秒で表示されますが (CloudWatch ではバイト/秒)、これはノードのストレージ容量としてより一般的な単位です。

CloudWatch コンソール、コマンドラインツール、または Amazon SDK で Amazon Redshift メトリクスを使用するときは、2 つの概念を覚えておいてください。

1. 最初に、対象のメトリクスディメンションを指定します。ディメンションは、メトリクスを一意に識別できるようにする名前と値のペアです。Amazon Redshift のディメンションは `ClusterIdentifier` および `NodeID` です。CloudWatch コンソールには、クラスターおよびノード固有のディメンションを簡単に選択できるように、`Redshift Cluster` および `Redshift Node` ビューが用意されています。ディメンションの詳細については、CloudWatch デベロッパーガイド のディメンションを参照してください。
2. 次に、`ReadIOPS` などのメトリクス名を指定します。

次の表は、使用できる Amazon Redshift メトリックディメンションのタイプをまとめたものです。メトリクスに応じて、データは 1 分または 5 分間隔で無料で利用できます。詳細については、「Amazon Redshift メトリクス」を参照してください。

CloudWatch 名前空間	ディメンション	説明
`AWS/Redshift`	`NodeID`	フィルターにより、クラスターのノードに固有のデータが要求されました。`NodeID` は、「Leader」、「Shared」、または「Compute-N」のいずれかになります。ここで、N はクラスターのノード数を表す 0、1、... です。"Shared" は、クラスターには 1 つのノードのみがあることを意味します。つまり、リーダーノードとコンピューティングノードが組み合わされています。

CloudWatch 名前空間	ディメンション	説明
AWS/Redshift	ClusterIdentifier	フィルターにより、クラスターに固有のデータが要求されました。クラスターに固有のメトリクスには、HealthStatus 、MaintenanceMode 、DatabaseConnections がありま す。ノードのメトリクスでもある、このディメンションの全般的なメトリクス (ReadIOPS など) は、ノードメトリクスデータの集計を表します。これらのメトリクスは、リーダーノードおよびコンピューティングノードの動作を集計するため、その解釈には注意してください。

ゲートウェイおよびボリュームメトリクスの使用は、その他のサービスメトリクスの使用と似ています。次を含む一般的なタスクの多くが、CloudWatch ドキュメントで説明されています。

- 利用可能なメトリクスを表示する
- メトリクスの統計を取得する
- CloudWatch アラームの作成

Query profiler

このドキュメントでは、クエリのコンポーネントとパフォーマンスを分析するためのグラフィカルツールである Query profiler について説明します。

Query profiler は、Amazon Redshift コンソールから表示できる、クエリのモニタリングおよびトラブルシューティング機能です。クエリのパフォーマンスを分析するのに役立ちます。主に、クエリに関する実行順序、実行計画、統計を視覚とグラフィックで表示し、それらを理解して、トラブルシューティングを行いやすくすることを目的としています。Query profiler は、次のタイプのクエリコンポーネントを分析するのに役立ちます。

- 子クエリ - 子クエリは、クエリからの作業の一部です。Amazon Redshift では、効率的であれば、1 つの大きなクエリとして処理する代わりにクエリを複数の子クエリに分割できます。プロファイラーには、それぞれの子クエリのプロパティが表示されます。子クエリは、ストリームと追加のサブコンポーネントで構成されます。

  Query profiler が表示する子クエリのタイプには、通常、以下のものが含まれます。

- 一時テーブルクエリ: この子クエリのテキストは CREATE TEMP TABLE コマンドで始まります。この子クエリは、処理する他の子クエリの一時テーブルを作成します。
- 統計クエリ: Query profiler では、この子クエリの先頭に次のコメントを追加して、識別しやすくします。

```
-- collect statistics of child query queryID
```

この子クエリは、Amazon Redshift クエリエンジンがパフォーマンスを最適化するために使用する情報を収集します。

> **Note**
>
> Query Profiler は、Amazon Redshift が実行する最後の子クエリとしてユーザーが指定するクエリを表示します。

- ストリーム - ストリームは、利用可能なコンピューティングノードのスライスにパッケージ化されたセグメントのコレクションです。それぞれの子クエリは、1 つ以上のセグメントで構成されます。Query profiler では、実行時間など、各ストリームのプロパティを確認できます。ストリームのリストを確認すれば、パフォーマンスのボトルネックをすばやく発見できる可能性が高くなります。
- セグメント - セグメントは、1 つのプロセスで実行できる複数のステップの組み合わせです。また、セグメントは、コンピューティングノードのスライスによって実行可能な最小コンパイルユニットです。スライスは、Amazon Redshift の並列処理単位です。並行して実行されるストリーム内のセグメント。Query profiler はセグメントをグラフィカルには表示しませんが、そのステップの詳細ペインにあるステップのセグメント情報にアクセスできます。
- ステップ - 各セグメントは、ステップのコレクションで構成されます。ステップは、クエリ内の作業の一部です。ステップには、例えば hashjoin や、テーブルからのレコードの読み取りである scan を含めることができます。

ストリーム、セグメント、ステップの詳細については、「Amazon Redshift データベースデベロッパーガイド」の「クエリプランと実行ワークフロー」を参照してください。

Query Profiler には、SYS_QUERY_HISTORY、SYS_QUERY_DETAIL、SYS_QUERY_EXPLAIN、SYS_CHILD_QUERY_TEXT ビューから返される情報が表示されます。これらのビューの詳細については、「Amazon Redshift

データベースデベロッパーガイド」の「SYS_QUERY_HISTORY」、「SYS_QUERY_DETAIL」、「SYS_QUERY_EXPLAIN」、「SYS_CHILD_QUERY_TEXT」を参照してください。

Query profiler は、データベースで最近実行されたクエリのクエリ情報のみを表示します。データベースに対して実行するのではなく、事前に読み込まれたキャッシュデータを使用して完了するクエリには、事前に情報が利用できなかった場合、クエリプロファイルは作成されません。これは、Amazon Redshift が上記に対してクエリプランを生成しないためです。

Query Profiler を使用するための前提条件

SYS モニタリングビューは、使いやすく、複雑さを軽減するように設計されており、効果的なモニタリングとトラブルシューティングに役立つさまざまなメトリクスを提供します。SYS モニタリングビューでは、クラスターのサイズやアクティビティに関係なく、過去 7 日分のクエリ履歴も確実に表示されます。ユーザーは自分が実行したクエリしか表示できませんが、スーパーユーザーはすべてのユーザーが実行したクエリを表示できます。

IAM ユーザーアカウントまたはロールには、コンソールの [クエリとデータベースのモニタリング] セクションにアクセスするためのアクセス許可が必要です。このセクションでは、ユーザーアカウントまたはロールにアクセス許可を追加する方法について説明します。

次のポリシーを使用して、IAM ユーザーアカウントまたはロールに最小限のアクセス許可を追加します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "redshift:DescribeClusters",
                "redshift-serverless:ListNamespaces",
                "redshift-serverless:ListWorkgroups",
                "redshift-data:ExecuteStatement",
                "redshift-data:DescribeStatement",
                "redshift-data:GetStatementResult"
            ],
            "Resource": [
                "arn:aws:redshift-serverless:us-east-1:111122223333:*",
```

```
                "arn:aws:redshift:us-east-1:111122223333:*"
            ]
        }
    ]
}
```

トピック

- ロールにクエリモニタリング許可を付与する
- ユーザーにクエリモニタリング許可を付与する
- IAM ID を使用した一時的な認証情報

ロールにクエリモニタリング許可を付与する

sys:monitor アクセス許可を付与されたロールを割り当てられたユーザーは、すべてのクエリを表示できます。sys:operator アクセス許可を付与されたロールを割り当てられたユーザーは、クエリのキャンセル、クエリ履歴の分析、VACUUM オペレーションを実行できます。

ロールにクエリモニタリング許可を付与する

1. 次のコマンドを使用して、システムモニタリングのアクセス権を付与します。*role-name* は、アクセス権を付与する対象ロールの名前です。

   ```
   grant role sys:monitor to "IAMR:role-name";
   ```

2. (オプション) 次のコマンドを使用して、システムオペレーターのアクセス権を付与します。*role-name* は、アクセス権を付与する対象ロールの名前です。

   ```
   grant role sys:operator to "IAMR:role-name";
   ```

ユーザーにクエリモニタリング許可を付与する

sys:monitor アクセス許可を割り当てられたユーザーは、すべてのクエリを表示できます。sys:operator アクセス許可を割り当てられたユーザーは、クエリのキャンセル、クエリ履歴の分析、VACUUM オペレーションを実行できます。

ユーザーにクエリモニタリング許可を付与する

1. 次のコマンドを使用して、システムモニタリングのアクセス権を付与します。*user-name* は、アクセス権を付与する対象ユーザーの名前です。

```
grant role sys:monitor to "IAMR:user-name";
```

2. (オプション) 次のコマンドを使用して、システムオペレーターのアクセス権を付与します。*user-name* は、アクセス権を付与する対象ユーザーの名前です。

```
grant role sys:operator to "IAMR:user-name";
```

IAM ID を使用した一時的な認証情報

このオプションは、クラスターに接続している場合のみ使用できます。この方法では、クエリプロファイラーは、ユーザー名を IAM アイデンティティにマッピングして、IAM アイデンティティとしてデータベースに接続するための一時パスワードを生成します。この方法を使用して接続するユーザーには、`redshift:GetClusterCredentialsWithIAM` への IAM アクセス許可を付与する必要があります。ユーザーがこの方法を使用できないようにするには、IAM ユーザーまたはロールを変更して、このアクセス許可を拒否します。

Amazon Redshift コンソールで Query profiler にアクセスしてクエリを分析する

Amazon Redshift Serverless またはプロビジョニングされた Amazon Redshift のいずれかの Query Profiler にアクセスできます。詳細については、次のセクションを参照してください。

トピック

- Amazon Redshift Serverless の Amazon Redshift コンソールで Query profiler にアクセスする
- Amazon Redshift Provisioned の Amazon Redshift コンソールで Query profiler にアクセスする

Amazon Redshift Serverless の Amazon Redshift コンソールで Query profiler にアクセスする

Amazon Redshift Serverless の Query profiler にアクセスするには、以下を実行します。

- Amazon Redshift Serverless コンソールを開きます。
- ナビゲーションペインの [モニタリング] で、[クエリとデータベースのモニタリング] を選択します。

- [ワークグループ] を選択します。
- [クエリとデータベースのモニタリング] を選択します。
- [クエリ] を選択します。
- [クエリの詳細] ページの [クエリプラン] タブを選択します。

クエリプランが利用可能な場合は、子クエリのリストが表示されます。[クエリ] を選択して、Query profiler で表示します。

Amazon Redshift Provisioned の Amazon Redshift コンソールで Query profiler にアクセスする

Amazon Redshift Provisioned の Query profiler にアクセスするには、以下を実行します。

- Amazon Redshift Provisioned のクラスターダッシュボードを開きます。
- [クラスター] を選択します。
- [Query monitoring] (クエリのモニタリング) を選択します。
- データベースに接続します。
- [クエリとデータベースのモニタリング] を選択します。
- [クエリ] を選択します。

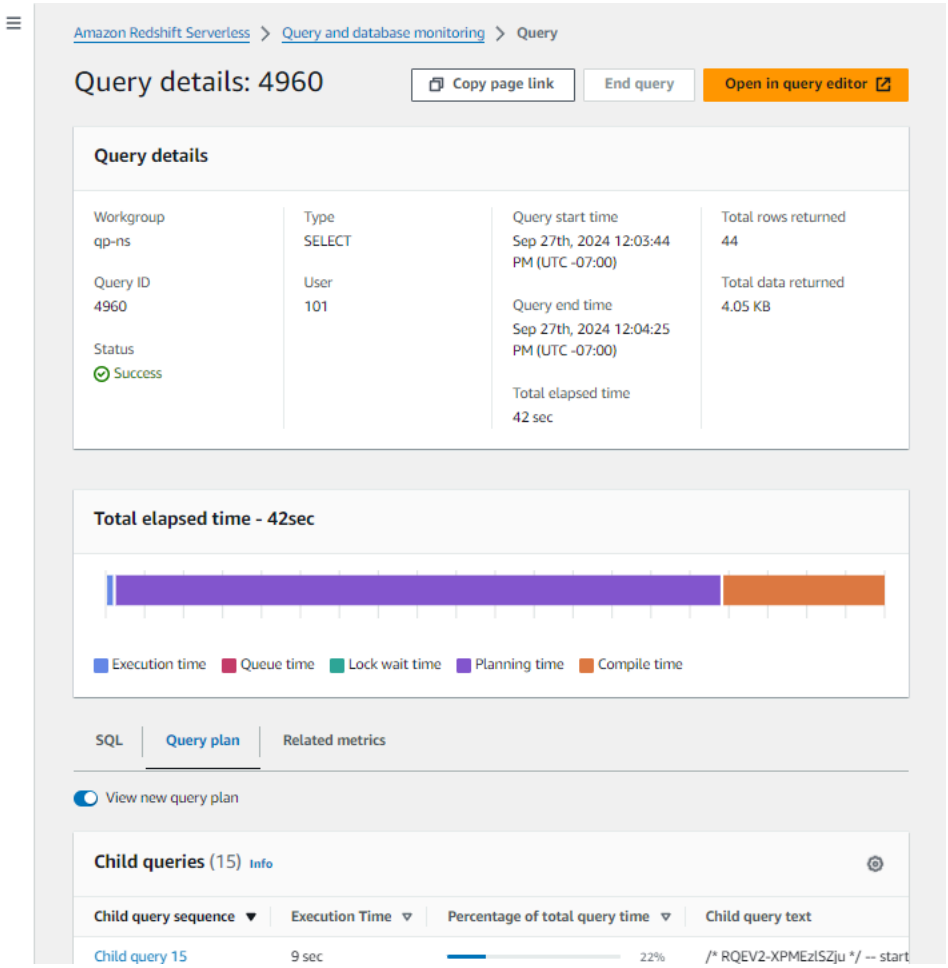
クエリプランが利用可能な場合は、子クエリのリストが表示されます。[クエリ] を選択して、Query profiler で表示します。

Query profiler のユーザーインターフェイス

Query profiler は、次のページを使用してクエリに関する情報を表示します。

- [クエリの詳細] ページ -このページには、クエリの統計と子クエリが表示されます。
- [子クエリ] ページ - このページには、子クエリの実行計画の統計、ストリーム、ビジュアル表現が表示されます。このページは、[クエリとデータベースのモニタリング] ページの [子クエリ] リストから [子クエリ] を選択するとコンソールに表示されます。

[クエリの詳細] ページ



[クエリの詳細] ページには、次のコンポーネントがあります。

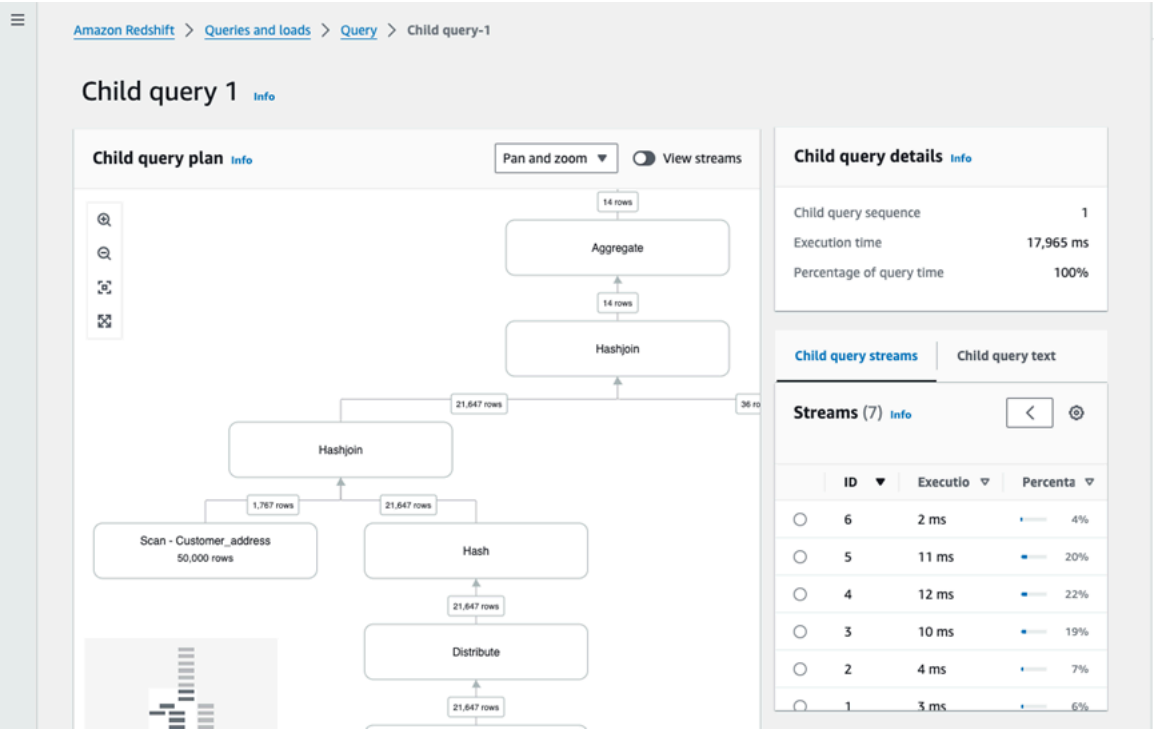
- 上部ペイン - ページの上部ペインには、ステータスやタイプなど、クエリの詳細が表示されます。上部ペインに表示される情報のソースについては、「Amazon Redshift データベースデベロッパーガイド」の「SYS_QUERY_HISTORY」を参照してください。
- [SQL] タブ - 下部パネルのこのタブには、元のユーザークエリの SQL テキストが表示されます。
- [クエリプラン] タブ - 下部パネルのこのタブには、Amazon Redshift がユーザークエリのデータと統計を準備するために使用される子クエリのリストが表示されます。デフォルトでは、[子クエリ] リストには、それぞれの子クエリに関する情報と集計された統計が表示されます。このページに表示される情報のソースについては、「Amazon Redshift データベースデベロッパーガイド」の「SYS_QUERY_DETAIL」を参照してください。

  [子クエリ] リストの列を追加または削除するには、[設定] メニューを使用します。

- [関連メトリクス] - 下部パネルのこのタブには、クエリの次の CloudWatch メトリクスが表示されます。
  - [使用した RPU キャパシティー] (サーバーレスワークグループの場合): クエリで使用したコンピューティングキャパシティーで、Redshift 処理ユニット (RPU) で測定されます。詳細については、「Amazon Redshift Serverless 容量を計算する」を参照してください。
  - [クラスターの健全性ステータス]、[CPU 使用率]、[使用済みのストレージキャパシティー] (プロビジョニングされたクラスターの場合): クエリが使用したステータスとシステムリソースです。
  - [アクティブなデータベース接続]: クエリの `DatabaseConnections` メトリクス。

  CloudWatch のメトリクスの詳細については、「Amazon Redshift のパフォーマンスデータ」を参照してください。

[子クエリ] ページ



[子クエリ] ページには、次のコンポーネントがあります。

- 子クエリのプルダウン - このコントロールは、子クエリごとのシーケンス名と実行時間を表示します。このコントロールで子クエリを選択すると、他の子クエリに移動できます。
- サイドパネル - このパネルには、子クエリストリームと子クエリテキストを表示するためのタブが含まれています。
- [子クエリストリーム] タブ - 上部パネルのこのタブには、以下のものが表示されます。

- [ストリーム] - このペインには、子クエリ内のストリームのリストが表示されます。このペインには、Amazon Redshift がクエリの最適化に使用するストリームに関する情報と集計データが表示されます。このペインの詳細については、「Amazon Redshift データベースデベロッパーガイド」の「SYS_QUERY_DETAIL」を参照してください。

  [設定] メニューを使用して、[ストリーム] ペインから列を追加または削除できます。[ストリーム] ペインの歯車アイコンを使用して、[設定] メニューにアクセスします。
- [子クエリ] ペイン - 子クエリのステップをグラフィカルに表現したものです。[子クエリ] ペインの詳細については、以下の「[子クエリ] ペイン」を参照してください。

- [子クエリテキスト] タブ - 上部パネルのこのタブには、子クエリの SQL コードが表示されます。
- [子クエリの詳細] ペイン - 右側のパネルのこのペインには、子クエリの詳細が表示されます。このペインの詳細については、「Amazon Redshift データベースデベロッパーガイド」の「SYS_QUERY_DETAIL」を参照してください。
- [ストリーム詳細] ペイン - [ストリーム] ペインで [ストリーム] を選択すると、[ストリーム詳細] ペインにストリームに関する情報が表示されます。このパネルの詳細については、「Amazon Redshift データベースデベロッパーガイド」の「SYS_QUERY_DETAIL」を参照してください。
- [ステップの詳細] ペイン - [ストリーム] ペインまたは [子クエリプラン] のグラフで [ステップ] を選択すると、[ステップの詳細] ペインにステップに関する情報が表示されます。このペインの詳細については、「Amazon Redshift データベースデベロッパーガイド」の「SYS_QUERY_DETAIL」を参照してください。

[子クエリ] ペイン

Query profiler は、選択した子クエリのステップのグラフィカル表現として、[子クエリ] ペインに子クエリを表示します。

[子クエリ] ペインには、実行順序とステップ間の関係が表示されます。例えば、あるステップが他の 2 つのステップからの出力を結合する場合、[子クエリ] ペインには、そのステップが 2 つのノードから入力されるツリーノードとして表示されます。



デフォルトでは、[子クエリ] ペインにはステップを含むストリームは表示されません。Amazon Redshift が子クエリのステップを論理的にパーティション化するために使用するストリームを表示す

るには、[ストリームの表示] を選択します。[ストリームの表示] を選択すると、[子クエリ] ペインにクエリのストリームに含まれるステップが表示されます。

[子クエリ] ペインには、セグメント情報は表示されません。ステップのセグメントを表示するには、[ステップ] を選択します。[ステップの詳細] ペインには、ステップのセグメントが表示されます。

[子クエリ] ペインのナビゲーション

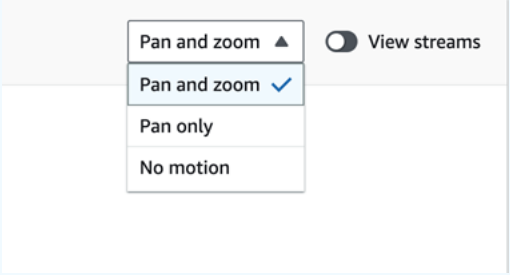
[子クエリ] ペインで [ステップ] を選択すると、それらに関する詳細情報を表示できます。また、ワークスペースでパンやズームを行い、クエリプランのステップをより視覚化的に確認できます。

次の方法を使用して、[子クエリ] プランでノード、パン、ズームを選択できます。

- マウスの使用 - [ノード]を選択して、ワークスペースをクリックしてドラッグしてパンし、**Ctrl** (Windows) または **CMD** (Mac) を長押ししてマウスホイールでズームできます。[ノード] を選択すると、ワークスペースをパンおよびズームすることにより、そのノードが強調表示されます。ワークスペースで [ストリーム] を選択すると、そのストリームは[ストリーム] リストで強調表示されます。ワークスペースでステップを選択すると、[ステップの詳細] ペインにそのステップに関する情報が表示されます。
- ワークスペースの左上にあるズームと画面合わせのコントロールを使用する - これらのコントロールを使用すると、ワークスペース全体に合わせてズームイン、ズームアウト、画面合わせ、全画面表示モードに移行できます。クエリプラン全体に合わせてズームすると、ワークスペースはクエリプランを水平方向と垂直方向の両方にセンタリングします。
- ワークスペースの右下にあるミニマップの使用 - ワークスペースの左下隅にあるミニマップコントロールを使用して、ワークスペースのパンまたはズームを行うことができます。
- [ストリーム] ペインでストリームを選択する - [ストリーム] ペインで [ストリーム] を選択すると、ワークスペースがパンおよびズームして、選択したストリームを表示し、[ストリームの詳細] ペインにストリームに関する情報を表示します。
- [ストリーム詳細] ペインで [ステップ] を選択する - [ストリーム詳細] ペインで [ステップ] を選択すると、ワークスペースをパンおよびズームして選択したステップを表示し、[ステップ詳細] ペインにステップに関する情報を表示します。

> **Note**
>
> ワークスペースまたは別のペインで [ステップ] を選択すると、ワークスペースはズームとパンを試行し、選択したステップが最適に表示されます。
> ワークスペースまたは別のペインからストリームまたはステップを選択すると、ワークスペースの右上のコントロールで [パンおよびズーム] を選択した場合のみ、ワークスペースは

そのストリームまたはステップにズームおよびパンします。この動作は、プルダウンで適切な設定を選択することにより、パンとズーム、パンのみ、または動作なしに制限できます。



Query profiler を使用したクエリのトラブルシューティング

クエリのトラブルシューティングを行う場合は、[子クエリ] を選択して、[合計クエリ時間の割合] が最も大きいストリームを判断できます。これは、クエリのどの部分を詳細に分析する必要があるかを判断するための簡単な方法です。

最も時間がかかっている子クエリが判明したら、そのステップを表示して、パフォーマンスを低下させている結合やスキャンを確認します。

クエリとデータベースのモニタリング

このドキュメントでは、Amazon Redshift のプロビジョン済みのクラスターまたはサーバーレスワークグループのパフォーマンスとこれらに対して実行されるクエリを分析するための AWS マネジメントコンソール 機能である、クエリとデータベースモニタリングページについて説明します。

クエリとデータベースモニタリングページを使用すると、次のシナリオを調査できます。

- 指定した期間内のデータウェアハウスメトリクスのモニタリング
- クエリがデータウェアハウス全体にわたるパフォーマンスに与える影響
- ロック待機時間、コンパイル時間、実行時間などのライフサイクルイベントへのクエリ実行時間の詳細の表示
- 指定した期間内でリソースを最も多く使用するクエリを実行するユーザー
- パッチイベントによるクエリのパフォーマンスへの影響のモニタリング

トピック

- アクセス許可
- クエリとデータベースモニタリングコンソール

アクセス許可

コンソールへのアクセスに使用する AWS アカウント の権限は、[クエリとデータベースモニタリング] ページに表示されるクエリに影響を及ぼします。デフォルトでは、自分のクエリのみを表示できます。他のユーザーが所有するクエリを表示するには、自分のアカウントに SYS:MONITOR ロールを付与してください。[クエリとデータベースモニタリング] ページからユーザーが実行中のクエリを終了できるようにするには、ユーザーに SYS:OPERATOR 権限を付与します。

Amazon Redshift Serverless またはプロビジョニング済みクラスターの IAM ユーザーまたはロールに sys:monitor ロールを自動的に割り当てるには、次のコマンドを実行します。

```
create role monitor;
grant role sys:monitor to role monitor;
```

クエリのモニタリングに使用される IAM ロールを更新するには、次の手順を実行してください。

1. [タグ] タブを選択します。
2. [タグを管理] を選択します。
3. キーが **RedshiftDbRoles**、値が **monitor** のタグを追加します。
4. 変更を保存する

データベース認証情報をユーザーに追加するには、次のコマンドを実行します。

```
grant role sys:monitor to <username>
```

GRANT コマンドの使用方法については、「Amazon Redshift データベースデベロッパーガイド」の「GRANT」を参照してください。

クエリのモニタリングを使用するには、IAM ユーザーに Amazon Redshift データプレーンへのアクセス許可が必要です。IAM ユーザーのアクセス許可ポリシーに、次のアクセス許可が含まれていることを確認してください。

```
{
    "Sid": "DataAPIPermissions",
    "Action": [
        "redshift-data:ExecuteStatement",
        "redshift-data:CancelStatement",
        "redshift-data:GetStatementResult",
```

```
        "redshift-data:DescribeStatement",
        "redshift-data:ListDatabases"
    ],
    "Effect": "Allow",
    "Resource": "arn:aws:redshift-serverless:us-
west-2:123456789012:workgroup/01234567-89ab-cdef-0123-456789abcdef"
},
```

IAM ID を使用した一時的な認証情報

このオプションは、クラスターに接続している場合のみ使用できます。この方法では、クエリとデータベースモニタリングは、ユーザー名を IAM アイデンティティにマッピングして、IAM アイデンティティとしてデータベースに接続するための一時パスワードを生成します。この方法を使用して接続するユーザーには、redshift:GetClusterCredentialsWithIAM への IAM アクセス許可を付与する必要があります。ユーザーがこの方法を使用できないようにするには、IAM ユーザーまたはロールを変更して、このアクセス許可を拒否します。

クエリとデータベースモニタリングコンソール

このセクションでは、クエリとデータベースモニタリングコンソールページの使用について説明します。

クエリとデータベースモニタリングコンソールを使用すると、データウェアハウスのパフォーマンスの概要を迅速に把握できます。データウェアハウスのパフォーマンスを経時的にモニタリングして、データウェアハウスのプロビジョン済みクラスターまたは個別のクエリのパフォーマンスを調べて、ボトルネックやその他の改善すべき領域を最適な方法で特定できます。

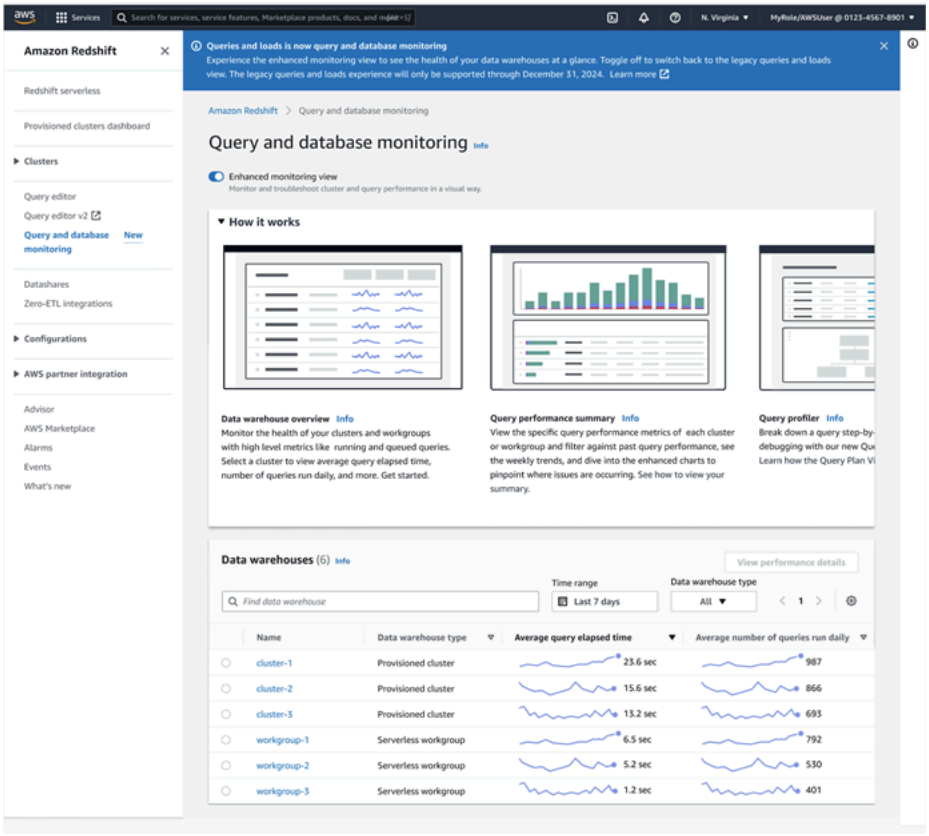
[クエリとデータベースモニタリング] ページは、次の機能を提供します。

- セキュリティの向上 — 他のユーザーのクエリをモニタリングするには、昇格された権限が必要です。詳細については、「アクセス許可」を参照してください。
- 7 日間のクエリ履歴 - 7 日間のクエリ履歴へのアクセスが保証されます。
- クエリのモニタリング — プロビジョン済みのクラスターとサーバーレスワークグループのクエリをユーザークエリレベルでモニタリングできます。
- クエリのトレンド分析 — 指定した条件に一致する類似のクエリのパフォーマンスを比較できます。

[クエリとデータベースモニタリング] ページにアクセスするには、以下を実行します。

1. https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールにサインインします。
2. ナビゲーションペインから [クエリとデータベースモニタリング] を選択します。

[クエリとデータベースモニタリング] コンソールページが次のとおり表示されます。



[クエリとデータベースモニタリング] ページには、次のコンポーネントがあります。

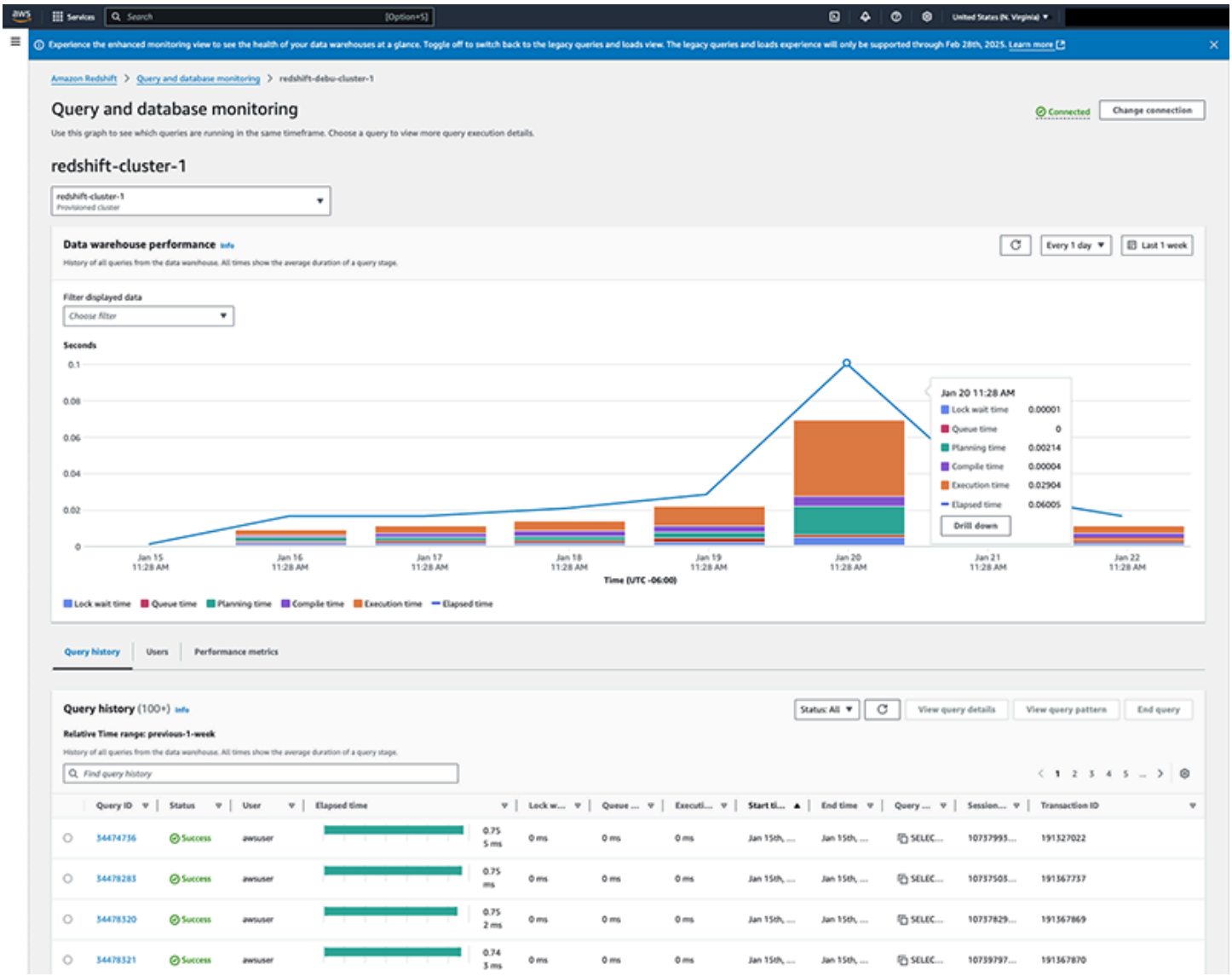
- データウェアハウスの概要 — プロビジョン済みのクラスターとサーバーレスワークグループの平均クエリパフォーマンスをモニタリングします。このページの統計でスパイクやアクティビティが高まる期間を調べると、特定のクラスターやワークグループのパフォーマンスの問題を迅速に確認できます。
- クエリパフォーマンスの要約 — プロビジョン済みの特定のクラスターとサーバーレスワークグループの平均クエリパフォーマンスをモニタリングします。[データウェアハウスの概要] リストでクラスターまたはワークグループのいずれかをクリックすると、[クエリパフォーマンスの要約] ページにアクセスすることもできます。

トピック

- クエリパフォーマンスの要約
- クエリの詳細
- クエリパターン

クエリパフォーマンスの要約

[クエリとデータベースモニタリング] ページから [クエリパフォーマンスの要約] をクリックするか、[データウェアハウスの概要] ページからクラスターまたはワークグループをクリックすると、コンソールに個別のプロビジョン済みクラスターまたはサーバーレスワークグループのパフォーマンスの概要が表示されます。



このページには次のコンポーネントが提供されています。

- クラスターまたはワークグループのドロップダウン — 分析するクラスターまたはワークグループを選択します。
- データウェアハウスのパフォーマンス — このペインには、指定された期間内のクラスターまたはワークグループの履歴が表示され、クエリの各フェーズに費やされた時間が表示されます。上記の画像で 1 月 20 日の計画や実行時間の増加など、特定のクエリフェーズでスパイクが見られる場合は、この情報を使用してクエリのパフォーマンスに関する問題を特定できます。表示されるデフォルトの期間は過去 7 日間です。ただし、分析のニーズに応じて期間は調整できます。
- クエリ履歴 — このペインには、指定したフィルター期間内で実行されたすべてのクエリのパフォーマンス履歴が表示されます。この情報を使用して、個別のクエリのパフォーマンスをトラブルシューティングできます。特定のクエリをさらに詳細に分析するには、このグラフまたは [データウェアハウスのパフォーマンス] グラフから該当のクエリを選択できます。

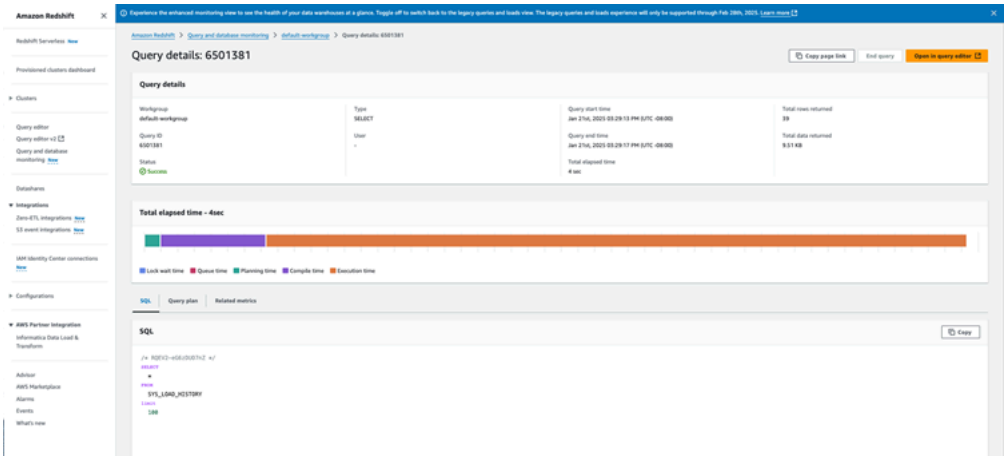
  グラフのデータポイントにカーソルを合わせると、ポップアップが表示されます。このポップアップには、そのデータポイントの時間メトリクスの詳細が表示されます。データポイントにクエリデータが含まれている場合は、[ドリルダウン] をクリックすると、グラフの時間範囲をデータポイントの次に小さい時間単位に更新できます。これらの時間単位は次のとおりです。

  - 日
  - 時間
  - 15 分
  - 5 分
  - 1 分

  例えば、データポイントで [ドリルダウン] をクリックすると、グラフの範囲が 1 日間に変わります。もう一度 [ドリルダウン] をクリックすると、グラフの範囲は 1 時間に変わります。
- Query Profiler - クエリのパフォーマンスをモニタリングするためのグラフィカルツール。詳細については、「Query profiler」を参照してください。

クエリの詳細

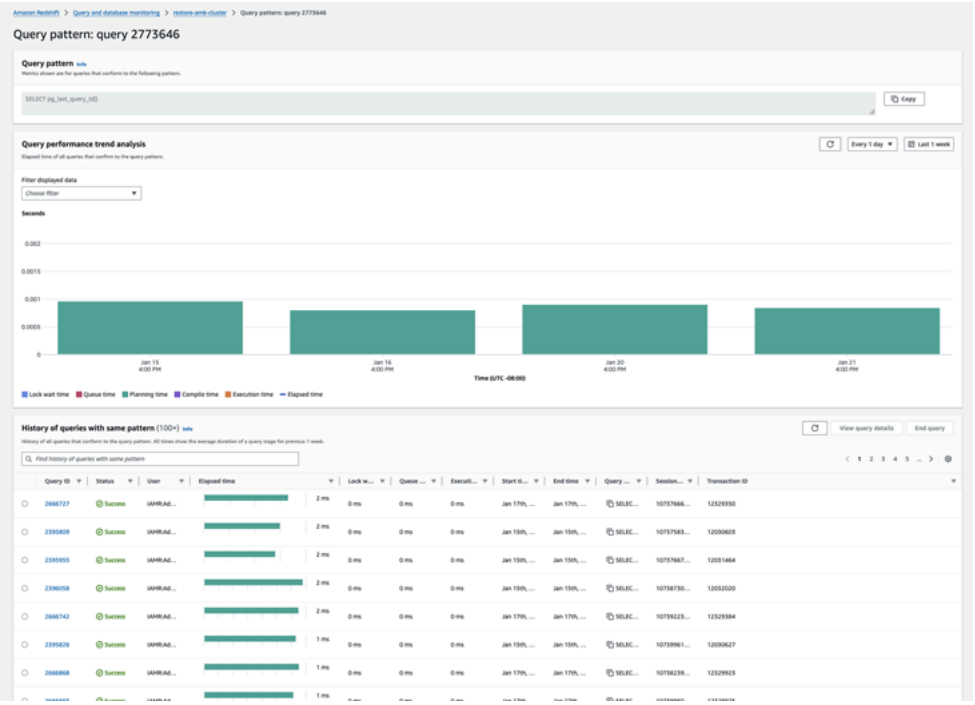
クラスターページまたはワークグループの詳細ページの [データウェアハウスのパフォーマンス] ペインまたは [クエリ履歴] ペインからクエリを選択すると、[クエリの詳細] ページが開きます。



[クエリの詳細] ページの詳細については、「[クエリの詳細] ページ」を参照してください。

クエリパターン

[クエリパフォーマンスの要約] ページの [クエリ履歴] ペインで [クエリパターンを表示] ボタンをクリックすると、同じパターンのクエリの履歴を表示できます。[クエリパターン] ページには、指定した SQL ステートメントが取得した過去 1 週間のすべてのクエリが表示されます。



[クエリパターン] ページに記載されるコンポーネントは次のとおりです。

- クエリパターン — 分析中のクエリを取得する SQL ステートメント

- クエリのパフォーマンストレンド分析 — クエリパターンが選択したすべてのクエリの経過時間を示す棒グラフ。結果は日別にグループ化されます。
- History of queries with the same pattern (同じパターンのクエリの履歴) — クエリパターンが選択したクエリの各フェーズにかかった時間

[クエリパターン] ページを使用すると、次のインサイトを取得できます。

- 毎日特定の時間に実行されるクエリのトレンド
- データウェアハウスに対して定期的に実行するクエリの実行時間のスパイク

SYS ビューベースのクエリとデータベースのモニタリング

このドキュメントでは、Amazon Redshift コンソールの [クエリとデータベースのモニタリング] ページにデータを提供する SYS ビューについて説明します。これは、クエリのコンポーネントとパフォーマンスを分析するためのツールです。クエリとデータベースのモニタリングページの詳細については、「クエリとデータベースのモニタリング」を参照してください。

[クエリとデータベースのモニタリング] ページには、SYS ビューで提供される情報の表示機能があります。コンソールビューには Query Profiler があり、クエリの実行プランがグラフィカルに表示されます。SYS ベースのビューに切り替えるには、以下の手順に従って、新しい [クエリとデータベースのモニタリング] ページに必要なアクセス権と権限を付与してください。

[クエリとデータベースのモニタリング] ページの SYS ベースのビュー機能には、次の機能があります。

- セキュリティの向上 - 他のユーザーのクエリをモニタリングするには、昇格された権限が必要です。
- 7 日間のクエリ履歴 - 7 日間のクエリ履歴へのアクセスが保証されます。
- Query Profiler - クエリのパフォーマンスをモニタリングするためのグラフィカルツール。詳細については、Query profilerを参照してください。

デフォルトでは、自分のクエリのみを表示できます。他のユーザーが所有するクエリを表示するには、自分のアカウントに SYS:MONITOR ロールを付与してください。実行中のクエリをユーザーが終了できるようにするには、そのユーザーに SYS:OPERATOR 権限を付与します。

すべてのユーザーが所有するクエリの表示権限をデータベースユーザーまたはロールに付与するには、次のコマンドを実行します。

```
grant role sys:monitor to "IAM:role-name";
grant role sys:monitor to "IAM:user-name";
```

Amazon Redshift Serverless またはプロビジョニング済みクラスターの IAM ユーザーまたはロールに sys:monitor ロールを自動的に割り当てるには、次のコマンドを実行します。

```
 create role monitor;
grant role sys:monitor to role monitor;
```

クエリのモニタリングに使用される IAM ロールを更新するには、次の手順を実行してください。

1. [タグ] タブを選択します。
2. [タグを管理] を選択します。
3. キーが **RedshiftDbRoles**、値が **monitor** のタグを追加します。
4. 変更を保存する

データベース認証情報をユーザーに追加するには、次のコマンドを実行します。

```
grant role sys:monitor to <username>
```

アクセス許可

クエリのモニタリングを使用するには、IAM ユーザーに Amazon Redshift データプレーンへのアクセス許可が必要です。IAM ユーザーのアクセス許可ポリシーに、次のアクセス許可が含まれていることを確認してください。

```
{
    "Sid": "DataAPIPermissions",
    "Action": [
        "redshift-data:ExecuteStatement",
        "redshift-data:CancelStatement",
        "redshift-data:GetStatementResult",
        "redshift-data:DescribeStatement",
        "redshift-data:ListDatabases"
    ],
    "Effect": "Allow",
    "Resource": "arn:aws:redshift-serverless:us-
west-2:123456789012:workgroup/01234567-89ab-cdef-0123-456789abcdef"
```

```
},
```

データベースに接続する

拡張クエリモニタリング機能を使用する前に、SYS ビューベースの情報にアクセスするために、まずデータベースに接続する必要があります。データベースに接続するには、次のいずれかの認証情報を使用します。

- ユーザー名とパスワード
- IAM ロールに関連付けられた一時的な認証情報
- データベースユーザー

拡張クエリモニタリングの使用については、次の点に注意してください。

- プロビジョニング済みクラスターの場合、拡張クエリモニタリングは SYS ビューを使用するため、データベースに接続する必要があります。これらのビューではセキュリティが強化されており、他のユーザーが所有するクエリに関するデータにアクセスするには、昇格された権限が必要です。
- SYS ビューベースのクエリとデータベースのモニタリングページを使用する場合、ユーザーアカウントにデータベースのスーパーユーザーロールが割り当てられていない場合は、自分の `user_id` のみが表示されます。ユーザー名は、スーパーユーザー以外のユーザーには表示されません。
- SYS ビューベースのクエリとデータベースのモニタリングページのエクスペリエンスの一部として、クエリ実行のプロセス ID (`p_id`) が列見出し `session_id` の下に表示されます。

Amazon Redshift でプロビジョニングされたクラスターとサーバーレスワークグループのトラック

Amazon Redshift が新しいバージョンをリリースすると、Amazon Redshift データウェアハウス (サーバーレスワークグループまたはプロビジョニングされたクラスター) のバージョンが更新されます。この際、データウェアハウスを最新のリリースに更新するか、認定済みの前のリリースに更新するかを選択することができます。

サーバーレスワークグループまたはプロビジョニングされたクラスターのトラックは、バージョン更新中に適用されるリリースバージョンを決定します。Amazon Redshift は、プロビジョニングされたクラスターを指定されたメンテナンスウィンドウ中に更新し、通常はサーバーレスワークグループをアイドル期間中に更新します。Redshift Serverless がワークグループを更新するタイミングの詳細については、「サーバーレスワークグループの更新」を参照してください。

Amazon Redshift が新しいバージョンをリリースすると、そのバージョンは最新のトラックに割り当てられ、前のバージョンは直近のトラックに割り当てられます。データウェアハウスのトラックを設定するには、次のいずれかの値を指定します。

- 最新 – 最新のトラックでは、最新の機能、セキュリティアップデート、パフォーマンス強化を備えた最新の認定リリースバージョンを使用します。
- 直近 – 直近のトラックでは、前の認定リリースを使用します。

例えば、サーバーレスワークグループが現在バージョン 1.0.2762 を実行し、Amazon Redshift が Redshift Serverless バージョン 1.0.3072 をリリースしているとします。トラック値が最新の場合、ワークグループはバージョン 1.0.3072 (最新リリース) に更新されます。トラック値を直近に設定した場合、次の直近トラックバージョンがリリースされるときにワークグループが更新されます。

直近トラック機能を使用すると、Amazon Redshift データウェアハウスのサブセットを直近トラックで実行できます。これにより、最新トラックに設定されているデータウェアハウスで 1~6 週間のテストと統合の検証を行ってから、直近トラックのデータウェアハウスにリリースを適用できます。デフォルトでは、Amazon Redshift はすべてのクラスターとワークグループを最新トラックに作成し、最新の認定リリースを利用します。ただし、Amazon Redshift の直近トラックを本番環境で使用し、最新トラックをテストおよび開発環境で使用すると、追加の労力と時間をかけて最新リリースを評価できます。直近のトラックは最大の安定性を保証し、本番環境のミッションクリティカルなワークロードに最適です。

> **Note**
>
> 直近のトラックバージョンは、短期間、最新のトラックバージョンと同じである場合があります。これは、最新のトラックが次のバージョンに進まない場合に起きます。通常、最新のトラックバージョンは直近のトラックバージョンより新しいものになります。

トラックの切り替え

Amazon Redshift リソースのトラックの変更は、通常 1 回限りの決定です。トラックを変更するときは注意が必要です。どの機能がどのデータウェアハウスバージョンにあるかについては、「Amazon Redshift のクラスターバージョン」を参照してください。

トラックを [直近] から [最新] に変更すると、データウェアハウスは [最新] のトラックリリースバージョンに更新されます。データウェアハウスのトラックを [直近] に変更すると、次のようにデータウェアハウスが更新されます。

- サーバーレスワークグループの場合、データウェアハウスのバージョンはアイドル期間中に更新されます。Redshift Serverless がワークグループのバージョンを更新する方法の詳細については、「サーバーレスワークグループの更新」を参照してください。
- プロビジョニングされたクラスターの場合、最新のトラックリリースバージョンよりも新しいバージョンがリリースされるまで、データウェアハウスは更新されません。

トラックと復元

サーバーレスワークグループの場合、スナップショットはターゲットの Amazon Redshift データウェアハウスのトラックを継承します。例えば、直近のトラックに設定されたワークグループのスナップショットを作成し、そのスナップショットを最新のトラックに設定されたワークグループに適用すると、ワークグループのトラック設定は最新になります。

プロビジョニングされたクラスターの場合、スナップショットはソースの Amazon Redshift データウェアハウスのトラックを継承します。スナップショットの作成後にソースデータウェアハウスのトラックを変更した場合、スナップショットとソースデータウェアハウスは別のトラックに設定されます。スナップショットから復元すると、新しいデータウェアハウスは、スナップショットソースから継承されたトラックに設定されます。トラックは、復元オペレーションが完了した後で変更できます。

データウェアハウスのサイズを変更しても、そのトラックには影響しません。

サーバーレスワークグループの更新

ワークグループが選択したトラックで新しいバージョンが使用可能になると、Amazon Redshift Serverless は通常、保留中のトラック更新リクエストがない限り、アイドル期間中に更新を適用します。ワークグループで 14 日以内にアイドル期間が発生しない場合、Redshift Serverless はバージョンの更新を強制します。

Redshift Serverless は、ワークグループを次の上位バージョンにのみ更新します。Redshift Serverless は、選択したワークグループのトラックのバージョンがワークグループの現在のバージョンより小さい場合でも、中間のバージョンをスキップしたりワークグループをダウングレードしたりしません。`Trailing` トラックが追いつかないかぎり、ワークグループはメジャーバージョンアップグレードを受け取りません。

例えば、`Current` トラックがバージョン 186 で、`Trailing` トラックのバージョンが 185 であるとします。ワークグループの `Track` 値が `Current` で、バージョンが 186 である場合、`Track` 値を `Trailing` に変更しても、Redshift Serverless はワークグループのバージョンを 185 にダウングレードしません。このシナリオでは、Redshift Serverless は `Trailing` トラックのバージョンが 186 以上になるまで、ワークグループをバージョン 186 に保持します。

トラックの変更が保留中の場合、Redshift Serverless は、トラックの変更を適用するまで、ワークグループを既存のトラックの次のメジャーバージョンに更新しません。トラックの変更が完了すると、Redshift Serverless はワークグループを新しいトラックの適切なバージョンに更新するための条件を評価します。

例えば、ワークグループが `Current` トラックに設定されていて、現在のトラックが 186 であり、ワークグループを `Trailing` トラックに変更した場合、Redshift Serverless はトラックの変更を適用した後、`Trailing` バージョンが 186 以上のバージョンに更新されるまでワークグループを更新しません。

> **Note**
>
> スナップショットからの復元、KMS キーの変更、サイズ変更など、ワークグループに対する既存の操作は、既存のトラックでのみ発生します。Redshift Serverless は、サーバーレスオペレーションに保留中のトラックを使用しません。

保留中のトラックの切り替えリクエストがある場合、UpdateWorkgroup を使用して `track` パラメータを元の値に戻すことで、リクエストをキャンセルできます。

バージョンの管理

トラックは一連のリリースを指します。Amazon Redshift データウェアハウスで最新のトラックか直近のトラックを使用するかを設定できます。データウェアハウスで最新のトラックを使用する場合、常に最新のリリースバージョンにアップグレードされます。リソースで直近のトラックを使用する場合、最後にリリースされたバージョンの直前にリリースされたリリースバージョンが常に実行されます。

プロビジョニングされたクラスターの場合、Amazon Redshift データウェアハウス の Amazon Redshift コンソールリストの [リリースステータス] 列は、リソースの 1 つがアップグレードに使用できるかどうかを示します。

ワークグループバージョンまたはクラスターのバージョンの確認

Amazon Redshift コンソールを使用して、Amazon Redshift Serverless ワークグループのバージョンまたはプロビジョニングされたクラスターのバージョンエンジンを確認できます。

AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。

Serverless workgroups

サーバーレスワークグループの場合、ナビゲーションメニューで、[ワークグループ] を選択し、リストからワークグループ名を選択して詳細を開きます。ワークグループの詳細が表示されます。

Provisioned clusters

プロビジョニングされたクラスターの場合、ナビゲーションメニューで [クラスター] を選択し、リストからクラスター名を選択して詳細を開きます。

クラスターの詳細が表示されます。これには、[クラスターのパフォーマンス]、[クエリのモニタリング]、[データベース]、[データ共有]、[スケジュール]、[メンテナンス]、[プロパティ] タブなどがあります。[Maintenance] (メンテナンス) タブを選択し、詳細を確認します。

[Maintenance (メンテナンス)] セクションで、[Current cluster version (現行のクラスターバージョン)] を見つけます。

> **Note**
>
> プロビジョニングされたクラスターの場合、コンソールには 1 つのフィールドにバージョン情報が表示されますが、Amazon Redshift API には 2 つのパラメータが表示されます。これらは、`ClusterVersion` パラメータと `ClusterRevisionNumber` パラメータです。詳細については、Amazon Redshift API リファレンスの「クラスター」を参照してください。

ゼロ ETL 統合

ゼロ ETL 統合はフルマネージドソリューションで、複数の運用ソースとトランザクションソースから Amazon Redshift でトランザクションデータと運用データが利用できます。このソリューションを使用して、ソースから Amazon Redshift データウェアハウスへの統合を設定できます。抽出、変換、ロード (ETL) パイプラインを管理する必要はありません。データソースから Amazon Redshift クラスターまたは Redshift Serverless 名前空間にデータレプリケーションの作成と管理を自動化することにより、お客様に代わって ETL が処理されます。レポートやダッシュボードなどの分析ワークロードには Amazon Redshift を使用して、ソースデータの更新とクエリを継続できます。

ゼロ ETL 統合では、分析、AI/ML、レポート作成用の新しいデータが得られます。ビジネスダッシュボード、最適化されたゲームエクスペリエンス、データ品質のモニタリング、顧客行動分析などのユースケース向けに、より正確でタイムリーなインサイトが得られます。ビジネス全体で、データ駆動型の予測をより自信を持って実行し、カスタマーエクスペリエンスを向上させ、データ駆動型のインサイトを促進できます。

ゼロ ETL 統合では現時点で、次のソースがサポートされています。

- Amazon Aurora MySQL (AMS)
- Amazon Aurora PostgreSQL (APG)
- Amazon DynamoDB
- Amazon RDS for MySQL
- Amazon RDS for Oracle
- Amazon RDS for PostgreSQL
- Oracle Database@AWS
- Salesforce、Salesforce Marketing Cloud Account Engagement、SAP、ServiceNow、Instagram ads、Meta ads、Zendesk などのアプリケーション
- 自己管理型 MySQL、PostgreSQL、SQL Server、Oracle

ゼロ ETL 統合を作成するには、統合ソースと、ターゲットとして Amazon Redshift データウェアハウスを指定します。初回のデータロード後、統合によって、ソースデータベースからターゲットデータウェアハウスにデータがレプリケートされます。データが Amazon Redshift で利用できるようになります。統合ソース、ゼロ ETL 統合、Amazon Redshift データウェアハウスのそれぞれの作成時に、データの暗号化を制御します。統合では、データパイプラインの状態をモニタリングし、可能な

場合は問題から回復します。同種ソースから単一の Amazon Redshift データウェアハウスに統合を作成して、複数のアプリケーションにわたって総合的なインサイトを引き出すことができます。

Amazon Redshift のデータを使用して、Amazon Redshift が提供する分析を利用できます。例えば、組み込みの機械学習 (ML)、マテリアライズドビュー、データ共有、複数のデータストアやデータレイクへの直接アクセスなどが利用できます。データエンジニアの場合、ゼロ ETL 統合を使用して、複雑なデータパイプラインでの断続的なエラーを原因とする遅延が発生する可能性のある、時間的制約のあるデータにアクセスできます。トランザクションデータに対して分析クエリや機械学習モデルを実行して、時間的制約のあるイベントやビジネス上の意思決定について、タイムリーなインサイトを引き出すことができます。

Amazon Redshift イベント通知サブスクリプションを作成すると、特定のゼロ ETL 統合でのイベント発生時に通知を受け取ることができます。統合関連のイベント通知のリストを確認するには、「Amazon EventBridge によるイベント通知のゼロ ETL 統合」を参照してください。サブスクリプションを作成する最も簡単な方法は、Amazon SNS コンソールを使用する方法です。Amazon SNS トピックの作成とサブスクリプションについては、「Amazon Simple Notification Service デベロッパーガイド」の「Amazon SNS の開始方法」を参照してください。

ゼロ ETL 統合を始める際には、以下の概念を検討してください。

- ソースデータベースは、データを Amazon Redshift にレプリケートする元になるデータベースです。
- ターゲットデータウェアハウスとは、データがレプリケートされる Amazon Redshift でプロビジョンされたクラスターまたは Redshift Serverless ワークグループです。
- 送信先データベースとは、ターゲットデータウェアハウスのゼロ ETL 統合から作成するデータベースです。

ゼロ ETL 統合のモニタリングに使用できるシステムテーブルとビューについては、「Amazon Redshift システムビューとのゼロ ETL 統合のモニタリング」を参照してください。

ゼロ ETL 統合の各ソースがサポートする AWS リージョン のリストについては、「ゼロ ETL 統合をサポートするリージョン」を参照してください。

ゼロ ETL 統合の価格の情報については、次の該当する料金ページを参照してください。

- Amazon Redshift の料金表
- Amazon Aurora の料金
- Amazon RDS の価格設定

- Amazon DynamoDB の価格設定
- AWS Glue の料金

ゼロ ETL 統合のソースの詳細については、次のトピックを参照してください。

- Aurora Zero の-ETL 統合については、「Amazon Aurora ユーザーガイド」のゼロ ETL 統合についての「利点」、「主な概念」、「制限事項」、「クォータ」、「サポートされているリージョン」を参照してください。
- RDS の-ETL 統合については、「Amazon RDS ユーザーガイド」のゼロ ETL 統合についての「利点」、「主な概念」、「制限事項」、「クォータ」、「サポートされているリージョン」を参照してください。
- DynamoDB ゼロ ETL 統合については、「Amazon DynamoDB デベロッパーガイド」の「DynamoDB の Amazon Redshift とのゼロ ETL 統合」を参照してください。
- アプリケーションとのゼロ ETL 統合については、「AWS Glue デベロッパーガイド」の「ゼロ ETL 統合」を参照してください。

トピック

- でゼロ ETL 統合を使用する場合の考慮事項
- ゼロ ETL 統合の開始方法
- Amazon Redshift でのデスティネーションデータベースの作成
- Amazon Redshift でのレプリケートしたデータのクエリ
- ゼロ ETL 統合の表示
- 履歴モード
- Amazon Redshift でのデータの共有
- ゼロ ETL 統合のモニタリング
- ゼロ ETL 統合のメトリクス
- DynamoDB のゼロ ETL 統合を変更する
- DynamoDB のゼロ ETL 統合を削除する
- ゼロ ETL 統合をサポートするリージョン
- ゼロ ETL 統合のトラブルシューティング

でゼロ ETL 統合を使用する場合の考慮事項

Amazon Redshift との ゼロ ETL 統合には、以下の制限が適用されます。

- ターゲット Amazon Redshift データウェアハウスは、次の前提条件を満たす必要があります。
  - Amazon Redshift Serverless または RG や RA3 ノードタイプのプロビジョン済みクラスターを実行している。
  - 暗号化されている (プロビジョニングされたクラスターを使用している場合)
  - 大文字と小文字の区別が有効になっている。
- Amazon Redshift データウェアハウスの承認済み統合ソースであるソースを削除すると、関連するすべての統合が FAILED 状態になります。以前にレプリケートされたデータは Amazon Redshift データベースに残っており、クエリを実行できます。
- デスティネーションデータベースは読み取り専用です。デスティネーションデータベースにテーブル、ビュー、またはマテリアライズドビューを作成することはできません。ただし、ターゲットデータウェアハウスのその他のテーブルでもマテリアライズドビューを使用できます。
- マテリアライズドビューは、クロスデータベースクエリで使用される場合にサポートされます。ゼロ ETL 統合を介してレプリケートされたデータを使用してマテリアライズドビューを作成する方法の詳細については、「レプリケートしたデータのマテリアライズドビューでのクエリ」を参照してください。
- デフォルトでは、クエリできるのは、Synced 状態のターゲットデータウェアハウスのテーブルのみです。別の状態のテーブルをクエリするには、データベースパラメータ QUERY_ALL_STATES を TRUE に設定します。QUERY_ALL_STATES の設定の詳細については、「Amazon Redshift データベースデベロッパーガイド」の「CREATE DATABASE」と「ALTER DATABASE」を参照してください。データベースの状態の詳細については、「Amazon Redshift データベースデベロッパーガイド」の「SVV_INTEGRATION_TABLE_STATE」を参照してください。
- Amazon Redshift では UTF-8 文字のみを使用できるため、ソースで定義された照合順序に従わない場合があります。ソートルールと比較ルールは異なる場合があり、それによって最終的にクエリ結果が変わる可能性があります。
- ゼロ ETL 統合は、Amazon Redshift データウェアハウスのターゲットごとに 50 に制限されています。
- 統合ソースのテーブルにはプライマリキーが必要です。それ以外の場合は、Amazon Redshift でターゲットのデータウェアハウスにテーブルが複製されません。

  Amazon Aurora PostgreSQL にプライマリキーを追加する方法については、AWS データベースブログの「Handle tables without primary keys while creating Amazon Aurora PostgreSQL zero-ETL

integrations with Amazon Redshift」を参照してください。Amazon Aurora MySQL または RDS for MySQL にプライマリキーを追加する方法については、AWS データベースブログの「Amazon Aurora MySQL または Amazon RDS for MySQL と Amazon Redshift とのゼロ ETL 統合を作成する際にプライマリキーがないテーブルを処理する」を参照してください。

- Aurora ゼロ ETL 統合でのデータフィルタリングを使用して、ソースの Aurora DB クラスターからターゲットの Amazon Redshift データウェアハウスへのレプリケーションの範囲を定義できます。すべてのデータをターゲットにレプリケートするのではなく、単一または複数のフィルターを定義して、特定のテーブルを選択的にレプリケーションの対象に含めたり除外したりできます。詳細については、「Amazon Aurora ユーザーガイド」の「Amazon Redshift との Aurora のゼロ ETL 統合向けのデータフィルタリング」を参照してください。
- Aurora PostgreSQL と Amazon Redshift のゼロ ETL 統合では、Amazon Redshift は Aurora PostgreSQL から最大 100 個のデータベースをサポートします。各データベースは、ソースからターゲットに個別にレプリケートされます。
- ゼロ ETL 統合は、トランザクションデータストアから Amazon Redshift にデータをレプリケートする際の変換をサポートしていません。データはソースデータベースからそのままレプリケートされます。ただし、Amazon Redshift でレプリケートしたデータに変換を適用することはできます。
- ゼロ ETL 統合は、並列接続を使用して Amazon Redshift で実行します。この実行には、統合からデータベースを作成したユーザーの認証情報を使用します。
- Amazon Redshift へのデータレプリケーションの頻度を制御するため、ゼロ ETL 統合の `REFRESH_INTERVAL` を設定できます。詳細については、「Amazon Redshift データベースデベロッパーガイド」の「CREATE DATABASE」および「ALTER DATABASE」を参照してください。
- Amazon DynamoDB とのゼロ ETL 統合から Amazon Redshift データベースを作成すると、データベースの状態が [作成中] から [アクティブ] に変わります。これにより、ソース DynamoDB テーブル内のデータのターゲット Redshift テーブルへのレプリケーションが開始されます。ターゲット Redshift テーブルは、送信先データベース (`ddb_rs_customerprofiles_zetl_db`) のパブリックスキーマの下に作成されます。

ターゲットで履歴モードを使用する場合の考慮事項

ターゲットデータベースで履歴モードを使用する場合、次の考慮事項が適用されます。詳細については、「履歴モード」を参照してください。

- ソースのテーブルを削除すると、ターゲットのテーブルは削除されず、DroppedSource 状態に変更されます。Amazon Redshift データベースからテーブルを削除したり、名前を変更したりすることができます。
- ソースのテーブルを切り捨てると、ターゲットテーブルで削除が実行されます。例えば、ソースですべてのレコードが切り捨てられると、ターゲット列 _record_is_active の対応するレコードが false に変更されます。
- ターゲットテーブルで TRUNCATE table の SQL を実行すると、アクティブな履歴行は非アクティブとマークされ、対応するタイムスタンプが追加されます。
- テーブル内の行が非アクティブに設定されると、短い 遅延後 (約 10 分後) に削除できるようになります。非アクティブな行を削除するには、クエリエディタ v2 などの SQL クライアントを使用して、ゼロ ETL データベースに接続します。
- 非アクティブな行が削除できるのは、履歴モードがオンになっているテーブルからのみです。例えば、次のような SQL コマンドは、非アクティブな行のみを削除します。

```
delete from schema.user_table where _record_delete_time <= '2024-09-10 12:34:56'
```

これは、次のとおりの SQL コマンドと同等です。

```
delete from schema.user_table where _record_delete_time <= '2024-09-10 12:34:56' and
 _record_is_active = False
```

- テーブルの履歴モードをオフにすると、すべての履歴データは <schema>.<table-name>_historical_<timestamp> という名前のテーブルに保存され、元の <schema>.<table-name> という名前のテーブルは更新されます。
- 履歴モードがオンになっているテーブルがテーブルフィルターを使用してレプリケーションから除外されると、すべての行が非アクティブに設定され、DroppedSource 状態に変更されます。テーブルフィルターの詳細については、「Amazon Aurora ユーザーガイド」の「Amazon Redshift との Aurora ゼロ ETL 統合でのデータフィルタリング」を参照してください。
- 履歴モードを true または false に切り替えられるのは、Synced 状態のテーブルに対してのみです。
- 履歴モードがオンになっているテーブルのマテリアライズドビューは、完全な再計算として作成されます。

ゼロ ETL 統合ソースが Aurora または Amazon RDS である場合の考慮事項

Amazon Redshift との Aurora と Amazon RDS ゼロ ETL 統合には、以下の制限が適用されます。

- Aurora と RDS for MySQL ゼロ ETL 統合でのデータフィルタリングを使用して、ソースの DB クラスターからターゲットの Amazon Redshift データウェアハウスへのレプリケーションの範囲を定義できます。すべてのデータをターゲットにレプリケートするのではなく、単一または複数のフィルターを定義して、特定のテーブルを選択的にレプリケーションの対象に含めたり除外したりできます。詳細については、「Amazon Aurora ユーザーガイド」の「Amazon Redshift との Aurora のゼロ ETL 統合向けのデータフィルタリング」を参照してください。
- 統合ソースのテーブルにはプライマリキーが必要です。それ以外の場合は、Amazon Redshift でターゲットのデータウェアハウスにテーブルが複製されません。

  Amazon Aurora PostgreSQL にプライマリキーを追加する方法については、AWS データベースブログの「Handle tables without primary keys while creating Amazon Aurora PostgreSQL zero-ETL integrations with Amazon Redshift」を参照してください。Amazon Aurora MySQL または RDS for MySQL にプライマリキーを追加する方法については、AWS データベースブログの「Amazon Aurora MySQL または Amazon RDS for MySQL と Amazon Redshift とのゼロ ETL 統合を作成する際にプライマリキーがないテーブルを処理する」を参照してください。
- Amazon Redshift VARCHAR データ型の最大長は 65,535 バイトです。ソースのコンテンツがこの制限に収まらない場合、レプリケーションは実行されず、テーブルのステータスは失敗になります。データベースパラメータ `TRUNCATECOLUMNS` を `TRUE` に設定すると、列に収まるようにコンテンツを切り捨てることができます。`TRUNCATECOLUMNS` の設定の詳細については、「Amazon Redshift データベースデベロッパーガイド」の「CREATE DATABASE」と「ALTER DATABASE」を参照してください。

  ゼロ ETL 統合ソースと Amazon Redshift データベース間のデータ型の違いの詳細については、「Amazon Aurora ユーザーガイド」の「Aurora と Amazon Redshift 間のデータ型の違い」を参照してください。

Aurora のソースについては、「Amazon Aurora ユーザーガイド」の「制限」も参照してください。

Amazon RDS のソースについては、「Amazon RDS ユーザーガイド」の「制限」も参照してください。

ゼロ ETL 統合ソースが DynamoDB である場合の考慮事項

Amazon Redshift との DynamoDB ゼロ ETL 統合には、以下の制限が適用されます。

- DynamoDB のテーブル名が 127 文字を超えるとサポートされません。
- DynamoDB ゼロ ETL 統合からのデータは、Amazon Redshift の SUPER データ型列にマッピングされます。
- 127 文字を超えるパーティションキーまたはソートキーの列名はサポートされていません。
- DynamoDB からのゼロ ETL 統合は、1 つの Amazon Redshift データベースにのみマッピングできます。
- パーティションキーとソートキーの場合、精度とスケールの最大値は (38,18) です。DynamoDB の数値データ型は、最大 38 の精度をサポートしています。Amazon Redshift も最大 38 の精度をサポートしていますが、Amazon Redshift のデフォルトの 10 進精度/スケールは (38,10) です。つまり、値のスケール値は切り捨てられる場合があります。
- ゼロ ETL 統合を正常に実行するには、DynamoDB 項目内の個々の属性 (名前と値で構成) が 64 KB を超えることはできません。
- 有効化すると、ゼロ ETL 統合により、DynamoDB テーブル全体を Amazon Redshift データベースにエクスポートします。この初期プロセスが完了するまでにかかる時間は、DynamoDB テーブルのサイズによって異なります。ゼロ ETL 統合により、その後は DynamoDB から Amazon Redshift への更新が、DynamoDB の増分エクスポートを使用して段階的にレプリケートされます。つまり、Amazon Redshift にレプリケートされる DynamoDB データは、自動的に最新に保たれます。

  現在、DynamoDB ゼロ ETL 統合の最小レイテンシーは 15 分です。ゼロ ETL 統合にゼロ以外の `REFRESH_INTERVAL` を設定することにより、さらに増やすことができます。詳細については、「Amazon Redshift データベースデベロッパーガイド」の「CREATE DATABASE」および「ALTER DATABASE」を参照してください。

Amazon DynamoDB ソースについては、「Amazon DynamoDB デベロッパーガイド」の「前提条件と制限」も参照してください。

ゼロ ETL 統合のソースが Salesforce、SAP、ServiceNow、Zendesk などのアプリケーションである場合の考慮事項

Amazon Redshift との統合のソースが Salesforce、SAP、ServiceNow、Zendesk などのアプリケーションである場合は、次の点を考慮してください。

- アプリケーションソースからのテーブル名と列名が 127 文字を上回る場合、これらの名前はサポートされません。
- Amazon Redshift VARCHAR データ型の最大長は 65,535 バイトです。ソースのコンテンツがこの制限に収まらない場合、レプリケーションは実行されず、テーブルのステータスは失敗に

なります。データベースパラメータ TRUNCATECOLUMNS を TRUE に設定すると、列に収まるようにコンテンツを切り捨てることができます。TRUNCATECOLUMNS の設定の詳細については、「Amazon Redshift データベースデベロッパーガイド」の「CREATE DATABASE」と「ALTER DATABASE」を参照してください。

ゼロ ETL 統合のアプリケーションソースと Amazon Redshift データベース間のデータ型の違いの詳細については、「AWS Glue デベロッパーガイド」の「ゼロ ETL 統合」を参照してください。

- アプリケーションとのゼロ ETL 統合の最小レイテンシーは 1 時間です。ゼロ ETL 統合にゼロ以外の REFRESH_INTERVAL を設定することにより、さらに増やすことができます。詳細については、「Amazon Redshift データベースデベロッパーガイド」の「CREATE DATABASE」および「ALTER DATABASE」を参照してください。

アプリケーションとのゼロ ETL 統合のソースについては、「AWS Glue デベロッパーガイド」の「ゼロ ETL 統合」も参照してください。

ゼロ ETL 統合ターゲットを復元する際の考慮事項

スナップショットまたは復旧ポイントから同じサーバーレス名前空間に Amazon Redshift Serverless 名前空間を復元すると、名前空間に関連付けられたゼロ ETL 統合が自動的に維持されます。ターゲット Amazon Redshift データベース内のデータがソースと一致するように、復元後に完全な再同期が開始されます。以下の考慮事項に注意してください。

- スナップショットを別の名前空間に復元しても、統合は維持されません。
- 復元後、Amazon Redshift は、維持されているすべてのゼロ ETL 統合に対して完全な再同期を実行します。再同期中、データインジェストのパフォーマンスが一時的に低下する可能性があります。
- ターゲットデータベースで履歴モードを使用する場合、スナップショットが作成されてから完全な再同期が完了するまでの間、レコードバージョンはキャプチャされません。再同期後、履歴モードでは通常のオペレーションが再開されます。この動作は、他の完全な再同期と同じです。詳細については、「履歴モード」を参照してください。
- スナップショットの作成後にゼロ ETL 統合が作成された場合、対応するデータベースがスナップショットに存在しないため、統合は復元後に NEEDS_ATTENTION 状態になります。これを解決するには、統合を含む最新のスナップショットから復元するか、統合を削除します。
- スナップショットの作成後にゼロ ETL 統合が削除された場合、統合は復元されません。スナップショットのデータベースは維持されますが、削除できます。

- スナップショットの作成後にテーブルレベルのフィルターが変更された場合、復元されたターゲットは、スナップショット作成時のフィルターではなく、統合からの現在のフィルターを使用します。
- 復元中に統合の維持をオプトアウトするには、AWS マネジメントコンソールの復元ページの [統合の維持] チェックボックスをオフにします。AWS CLI を使用している場合は、restore-from-snapshot または restore-from-recovery-point API オペレーションを呼び出すときに `--no-maintain-integration` パラメータを設定します。オプトアウトすると、統合は復元後に `FAILED` 状態になります。統合を削除して再作成できます。
- この機能は、同じサーバーレス名前空間に復元された場合にのみ Amazon Redshift Serverless に適用されます。プロビジョニングされたクラスターのスナップショット復元では、ゼロ ETL 統合は維持されません。

ゼロ ETL 統合ターゲットのサイズ変更時の考慮事項

Amazon Redshift のプロビジョニングされたクラスターがゼロ ETL 統合のターゲットになっている場合、クラスターのサイズ変更を行うと、統合テーブルの再同期が発生する場合があります。再同期が発生するかどうかは、サイズ変更のタイプによって異なります。

- ノード数のみを変更する伸縮自在なサイズ変更 (インプレースサイズ変更) は、ゼロ ETL 統合には影響しません。テーブルの同期はそのままです。
- ノードタイプを変更する伸縮自在なサイズ変更では、クラスター上のゼロ ETL 統合内のすべてのテーブルが再同期されます。従来のサイズ変更でも再同期がトリガーされます。これは、サービスがデータを新しいクラスター設定に再分散している間は、これらのサイズ変更オペレーションによってテーブルの分散スタイルが一時的に変更されるためです。詳細については、「クラスターのサイズ変更」を参照してください。

サイズ変更によって再同期がトリガーされる場合、以下の考慮事項が適用されます。

- 統合はアクティブなままで、サイズ変更が完了すると再同期が自動的に開始されます。ご自身では特に何もする必要はありません。
- テーブルの再同期中は、Amazon Redshift でそのテーブルに対してクエリを実行することはできません。再同期中にテーブルをクエリ可能な状態に保つには、サイズ変更を開始する前に、ターゲットデータベースで `QUERY_ALL_STATES` パラメータを `TRUE` に設定します。再同期中に返されるデータは、再同期が完了するまで古い場合があります。詳細については、「Amazon Redshift データベースデベロッパーガイド」の「CREATE DATABASE」および「ALTER DATABASE」を参照してください。

- ソースデータベースのサイズによっては、再同期に 20～25 分以上かかる場合があります。
- SVV_INTEGRATION_TABLE_STATE システムビューを使用して、統合テーブルの状態をモニタリングできます。テーブルは、再同期が完了して Synced に戻るまで、ResyncRequired 状態または ResyncInitiated 状態を示します。

ゼロ ETL 統合の開始方法

この一連のタスクでは、最初のゼロ ETL 統合を設定する手順を説明します。まず、統合ソースを設定し、必要なパラメータとアクセス許可を設定します。その後、Amazon Redshift コンソールまたは AWS CLI から残りの初期セットアップに進みます。コンソールには、一部の設定に関する問題を修正するための [修正する] オプションが用意されています。

トピック

- ターゲット Amazon Redshift データウェアハウスを作成して設定する
- データウェアハウスの大文字と小文字の区別を有効にする
- Amazon Redshift データウェアハウスの認可を設定する
- ゼロ ETL 統合の作成

ターゲット Amazon Redshift データウェアハウスを作成して設定する

このステップでは、Redshift Serverless ワークグループやプロビジョン済みのクラスターなどのターゲット Amazon Redshift データウェアハウスを作成して設定します。ゼロ ETL 統合で使用するために Amazon Redshift データウェアハウスが既に設定済みの場合は、このステップをスキップできます。

ターゲットデータウェアハウスには次の特性が必要です。

- Amazon Redshift Serverless または RG や RA3 ノードタイプのプロビジョン済みクラスターを実行している。
- 大文字と小文字の区別 (enable_case_sensitive_identifier) がオンになっている。詳細については、「データウェアハウスの大文字と小文字の区別を有効にする」を参照してください。
- ターゲットデータウェアハウスが Amazon Redshift でプロビジョンしたクラスターの場合、暗号化されている。詳細については、「Amazon Redshift データベース暗号化」を参照してください。
- 統合ソースと同じ AWS リージョンで作成されている。

ゼロ ETL 統合のためのターゲットデータウェアハウスを作成するには、デプロイタイプに応じて次のトピックのいずれかを参照してください。

- Amazon Redshift でプロビジョニングされたクラスターを作成するには、「クラスターの作成」を参照してください。
- 名前空間を伴う Amazon Redshift Serverless ワークグループを作成するには、「名前空間を伴うワークグループの作成」を参照してください。

プロビジョニングされたクラスターを作成すると、Amazon Redshift はデフォルトのパラメータグループも作成します。デフォルトのパラメータグループは編集できません。ただし、新しいクラスターを作成する前にカスタムパラメータグループを作成し、それをクラスターに関連付けることはできます。または、作成したクラスターに関連付けるパラメータグループを編集することもできます。また、カスタムパラメータグループを作成する際、またはゼロ ETL 統合を使用するために現在のパラメータグループを編集する際に、パラメータグループの大文字と小文字の区別をオンにする必要があります。

Amazon Redshift コンソールまたは AWS CLI を使用してカスタムパラメータグループを作成するには、「パラメータグループの作成」を参照してください。

データウェアハウスの大文字と小文字の区別を有効にする

作成中にパラメータグループをアタッチして、プロビジョニング済みのクラスターの大文字と小文字の区別を有効にできます。ただし、AWS Command Line Interface (AWS CLI) を使用してサーバーレスワークグループを更新できるのは、作成後のみです。これは、ソーステーブルと列の大文字と小文字の区別をサポートするために必要です。`enable_case_sensitive_identifier` は、データベース、テーブル、および列の名前識別子が大文字と小文字を区別するかどうかを決定する構成値です。データウェアハウスでゼロ ETL 統合を作成するには、このパラメータをオンにする必要があります。詳細については、「enable_case_sensitive_identifier」を参照してください。

Amazon Redshift Serverless – AWS CLIを使用して Amazon Redshift Serverless の大文字と小文字の区別を有効にする の場合。Amazon Redshift Serverless の大文字と小文字の区別を有効にできるのは、AWS CLI からのみであることに注意が必要です。

Amazon Redshift でプロビジョニングされたクラスターの場合、次のトピックのいずれかを使用して、ターゲットクラスターの大文字と小文字の区別を有効にします。

- Amazon Redshift コンソールを使用して Amazon Redshift でプロビジョニングされたクラスターの大文字と小文字の区別を有効にする

- AWS CLI を使用して Amazon Redshift でプロビジョニングされたクラスターの大文字と小文字の区別を有効にする

AWS CLIを使用して Amazon Redshift Serverless の大文字と小文字の区別を有効にする

次の AWS CLI コマンドを実行して、ワークグループの大文字と小文字の区別を有効にします。

```
aws redshift-serverless update-workgroup \
        --workgroup-name target-workgroup \
        --config-parameters
 parameterKey=enable_case_sensitive_identifier,parameterValue=true
```

ワークグループのステータスが `Active` になるのを待って、次のステップに進みます。

Amazon Redshift コンソールを使用して Amazon Redshift でプロビジョニングされたクラスターの大文字と小文字の区別を有効にする

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. 左側のナビゲーションペインで、[プロビジョニングされたクラスターダッシュボード] を選択します。
3. データをレプリケートするプロビジョニングされたクラスターを選択します。
4. 左側のナビゲーションペインで、[設定] > [ワークロード管理] を選択します。
5. [ワークロード管理] ページで、[パラメータグループ] を選択します。
6. [パラメータ] タブを選択します。
7. [パラメータの編集] を選択して、[enable_case_sensitive_identifier] を [true] に変更します。
8. 次に、[保存] を選択します。

AWS CLI を使用して Amazon Redshift でプロビジョニングされたクラスターの大文字と小文字の区別を有効にする

1. デフォルトのパラメータグループは編集できないため、ターミナルプログラムから次の AWS CLI コマンドを実行してカスタムパラメータグループを作成します。後で、それをプロビジョン済みのクラスターに関連付けます。

```
aws redshift create-cluster-parameter-group \
    --parameter-group-name zero-etl-params \
    --parameter-group-family redshift-2.0 \
    --description "Param group for zero-ETL integrations"
```

2. 次の AWS CLI コマンドを実行して、パラメータグループの大文字と小文字の区別を有効にします。

```
aws redshift modify-cluster-parameter-group \
    --parameter-group-name zero-etl-params \
    --parameters ParameterName=enable_case_sensitive_identifier,ParameterValue=true
```

3. 次のコマンドを実行して、パラメータグループをクラスターに関連付けます。

```
aws redshift modify-cluster \
    --cluster-identifier target-cluster \
    --cluster-parameter-group-name zero-etl-params
```

4. プロビジョニングされたクラスターが利用可能になるまで待ちます。describe-cluster コマンドを使用して、クラスターのステータスを確認できます。続いて、次のコマンドを使用してクラスターを再起動します。

```
aws redshift reboot-cluster \
    --cluster-identifier target-cluster
```

Amazon Redshift データウェアハウスの認可を設定する

統合ソースから Amazon Redshift データウェアハウスにデータをレプリケートするには、まず次の 2 つのエンティティを追加する必要があります。

- 認可されたプリンシパル — データウェアハウスにゼロ ETL 統合を作成できるユーザーまたはロールを特定します。
- 承認された統合ソース – データウェアハウスを更新できるソースクラスターを識別します。

承認されたプリンシパルと承認された統合ソースは、Amazon Redshift コンソールの [リソースポリシー] タブから、または Amazon Redshift `PutResourcePolicy` API オペレーションを使用して設定できます。

承認されたプリンシパルを追加

Redshift Serverless ワークグループまたはプロビジョニングされたクラスターへのゼロ ETL 統合を作成するには、関連する名前空間またはプロビジョニングされたクラスターへのアクセスを許可します。

次の両方の条件に該当する場合は、このステップをスキップできます。

- Redshift Serverless ワークグループまたはプロビジョンされたクラスターを所有する AWS アカウント は、ソースデータベースも所有しています。
- そのプリンシパルは、この Redshift Serverless 名前空間またはプロビジョニングされたクラスターにゼロ ETL 統合を作成するアクセス許可を持つ ID ベースの IAM ポリシーに関連付けられています。

Amazon Redshift Serverless 名前空間に許可されたプリンシパルを追加する

1. Amazon Redshift コンソールの左側のナビゲーションペインで、[Redshift Serverless] を選択します。
2. [名前空間の設定] を選択してから名前空間を選択して、[リソースポリシー] タブに移動します。
3. [承認されたプリンシパルを追加] を選択します。
4. 追加する承認されたプリンシパルごとに、ゼロ ETL 統合を作成するためにアクセス権限を付与する AWS ユーザーまたはロールの ARN、または AWS アカウント ID を名前空間に入力します。アカウント ID は ARN として保存されます。
5. [Save changes] (変更の保存) をクリックします。

許可されたプリンシパルを Amazon Redshift でプロビジョニングされたクラスターに追加する

1. Amazon Redshift コンソールの左側のナビゲーションペインで、[プロビジョニングされたクラスターダッシュボード] を選択します。
2. [クラスター] を選択してからクラスターを選択し、[リソースポリシー] タブに移動します。
3. [承認されたプリンシパルを追加] を選択します。
4. 追加する承認されたプリンシパルごとに、ゼロ ETL 統合を作成するためにアクセス権限を付与する AWS ユーザーまたはロールの ARN、または AWS アカウント ID をクラスターに入力します。アカウント ID は ARN として保存されます。
5. [Save changes] (変更の保存) をクリックします。

承認された統合ソースを追加

ソースが Amazon Redshift データウェアハウスを更新できるようにするには、ソースを承認された統合ソースとして名前空間に追加する必要があります。

承認済みの統合ソースを Amazon Redshift Serverless 名前空間に追加する

1. Amazon Redshift コンソールで、[サーバーレスダッシュボード] に移動します。
2. 名前空間の名前を選択します。
3. [リソースポリシー] タブに移動します。
4. [承認された統合ソースを追加] を選択します。
5. ゼロ ETL 統合のソースの ARN を指定します。

> **Note**
>
> 承認された統合ソースを削除すると、データが名前空間にレプリケートされなくなります。このアクションにより、そのソースからこの名前空間へのゼロ ETL 統合がすべて非アクティブ化されます。

Amazon Redshift でプロビジョニングされたクラスターに承認された統合ソースを追加

1. Amazon Redshift コンソールで、[プロビジョニングされたクラスターダッシュボード] に移動します。
2. プロビジョニングされたクラスターの名前を選択します。
3. [リソースポリシー] タブに移動します。
4. [承認された統合ソースを追加] を選択します。
5. ゼロ ETL 統合のデータソースであるソースの ARN を指定します。

> **Note**
>
> 承認された統合ソースを削除すると、データがプロビジョニングされたクラスターにレプリケートされなくなります。このアクションにより、そのソースからこの Amazon Redshift でプロビジョンされたクラスターへのゼロ ETL 統合がすべて非アクティブ化されます。

Amazon Redshift API を使用して承認を設定する

Amazon Redshift API オペレーションを使用して、ゼロ ETL 統合で機能するリソースポリシーを設定できます。

名前空間へのインバウンド統合を作成できるソースを制御するには、リソースポリシーを作成して、このポリシーを名前空間にアタッチします。リソースポリシーを使用して、統合にアクセスできるソースを指定できます。リソースポリシーはターゲットデータウェアハウスの名前空間にアタッチされ、ソースがライブデータをソースから Amazon Redshift にレプリケートするインバウンド統合を作成できるようにします。

リソースポリシーのサンプルは、次のとおりです。

```
{
  "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/
cc4ffe56-ad2c-4fd1-a5a2-f29124a56433",
      "Condition": {
        "StringEquals": {
          "aws:SourceArn": "arn:aws:rds:us-east-1:111122223333:cluster:foo"
        }
      }
    },
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": "redshift:CreateInboundIntegration",
      "Resource": "arn:aws:redshift-serverless:us-east-1:123456789012:namespace/
cc4ffe56-ad2c-4fd1-a5a2-f29124a56433"
    }
  ]
}
```

統合のリソースポリシーの設定に適用できる Amazon Redshift API オペレーションの概要は次のとおりです。

- リソースポリシーを永続化するには、PutResourcePolicy API オペレーションを使用します。別のリソースポリシーを指定すると、そのリソース上の以前のリソースポリシーが置き換えられます。上記のリソースポリシーのサンプルを使用して、次のとおりのアクションのアクセス許可を付与します。
  - `CreateInboundIntegration` – ソースプリンシパルが、ソースからターゲットデータウェアハウスにレプリケートされるデータのインバウンド統合を作成することを許可します。
  - `AuthorizeInboundIntegration` – Amazon Redshift が、ターゲットデータウェアハウスがソース ARN からレプリケートされたデータを受信できるかどうかを継続的に検証することを許可します。
- 既存のリソースポリシーを表示するには、GetResourcePolicy API オペレーションを使用します。
- リソースからリソースポリシーを削除するには、DeleteResourcePolicy API オペレーションを使用します。

リソースポリシーの更新には、put-resource-policy AWS CLI コマンドも使用できます。例えば、DynamoDB ソースの Amazon Redshift 名前空間 ARN にリソースポリシーを適用するには、次のような AWS CLI コマンドを実行します。

```
aws redshift put-resource-policy \
--policy file://rs-rp.json \
--resource-arn "arn:aws:redshift-serverless:us-east-1:123456789012:namespace/cc4ffe56-ad2c-4fd1-a5a2-f29124a56433"
```

ここでは `rs-rp.json` に以下が含まれます。

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/cc4ffe56-ad2c-4fd1-a5a2-f29124a56433",
            "Condition": {
                "StringEquals": {
                    "aws:SourceArn": "arn:aws:dynamodb:us-
east-1:123456789012:table/test_ddb"
                }
            }
        },
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:root"
            },
            "Action": "redshift:CreateInboundIntegration",
            "Resource": "arn:aws:redshift-serverless:us-
east-1:123456789012:namespace/cc4ffe56-ad2c-4fd1-a5a2-f29124a56433"
        }
    ]
}
```

ゼロ ETL 統合の作成

まず、ゼロ ETL 統合を作成して、ソースデータを Amazon Redshift にレプリケートします。

データのソースによって、作成するゼロ ETL 統合のタイプが決まります。

トピック

- Aurora のゼロ ETL 統合を作成する
- Amazon RDS のゼロ ETL 統合を作成する
- DynamoDB のゼロ ETL 統合を作成する
- アプリケーションとのゼロ ETL 統合を作成する

Aurora のゼロ ETL 統合を作成する

このステップでは、Amazon Redshift との Aurora ゼロ ETL 統合を作成します。

Aurora の Amazon Redshift とのゼロ ETL 統合を作成するには

1. 「Amazon Aurora ユーザーガイド」の説明に従って、Amazon RDS コンソールで カスタム DB クラスターパラメータグループを作成 します。
2. 「Amazon Aurora ユーザーガイド」の説明に従って、Amazon RDS コンソールで ソース Amazon Aurora DB クラスターを作成 します。
3. Amazon Redshift コンソール: ターゲット Amazon Redshift データウェアハウスを作成して設定する.
   - AWS CLI または Amazon Redshift コンソール: データウェアハウスの大文字と小文字の区別を有効にする.
   - Amazon Redshift コンソール: Amazon Redshift データウェアハウスの認可を設定する.
4. 「Amazon Aurora ユーザーガイド」の説明に従って、Amazon RDS コンソールで ゼロ ETL 統合を作成 します。
5. Amazon Redshift コンソールまたはクエリエディタ v2 で、統合から Amazon Redshift データベースを作成 します。

   次に、クエリを実行して、レプリケートしたデータを使用してマテリアライズドビューを作成します。

Aurora のゼロ ETL 統合の作成についての詳細は、「Amazon Aurora ユーザーガイド」の「Amazon Aurora の Amazon Redshift とのゼロ ETL 統合の作成」を参照してください。

Amazon RDS のゼロ ETL 統合を作成する

このステップでは、Amazon Redshift との Amazon RDS ゼロ ETL 統合を作成します。Redshift は、RDS for MySQL、RDS for PostgreSQL、および RDS for Oracle との統合をサポートしています。

Amazon RDS の Amazon Redshift とのゼロ ETL 統合を作成するには

1. 「Amazon RDS ユーザーガイド」の説明に従って、Amazon RDS コンソールで カスタム DB パラメータグループを作成 します。
2. 「Amazon RDS ユーザーガイド」の説明に従って、Amazon RDS コンソールで ソース Amazon RDS インスタンスを作成 します。
3. Amazon Redshift コンソール: ターゲット Amazon Redshift データウェアハウスを作成して設定する.

- AWS CLI または Amazon Redshift コンソール: データウェアハウスの大文字と小文字の区別を有効にする.
- Amazon Redshift コンソール: Amazon Redshift データウェアハウスの認可を設定する.

4. 「Amazon RDS ユーザーガイド」の説明に従って、Amazon RDS コンソールで ゼロ ETL 統合を作成 します。
5. Amazon Redshift コンソールまたはクエリエディタ v2 で、統合から Amazon Redshift データベースを作成 します。

   次に、クエリを実行して、レプリケートしたデータを使用してマテリアライズドビューを作成します。

Amazon RDS コンソールには、ソースデータベースとターゲットの Amazon Redshift データウェアハウスを指定するステップバイステップの統合作成フローが提供されています。問題が発生した場合は、Amazon RDS コンソールまたは Amazon Redshift コンソールのいずれかで手動で問題を解決せず、Amazon RDS に問題を自動的に修正させることを選択できます。

RDS ゼロ ETL 統合の作成についての詳細は、「Amazon RDS ユーザーガイド」の「Amazon RDS の Amazon Redshift とのゼロ ETL 統合の作成」を参照してください。

Amazon RDS for Oracle ゼロ ETL 統合を具体的に作成する詳細な手順については、「Oracle Database@AWS ユーザーガイド」の「Setting up a zero-ETL integration」を参照してください。

DynamoDB のゼロ ETL 統合を作成する

ゼロ ETL 統合を作成する前に、「でゼロ ETL 統合を使用する場合の考慮事項」で概説されている考慮事項と要件を確認してください。この一般的なフローに従って、DynamoDB から Amazon Redshift へのゼロ ETL 統合を作成します。

ゼロ ETL 統合で DynamoDB のデータを Amazon Redshift にレプリケーションするには

1. サインイン認証情報で、Amazon Redshift と DynamoDB とのゼロ ETL 統合を実行するアクセス許可が付与されていることを確認します。IAM ポリシーの例については、「DynamoDB ゼロ ETL 統合と連携する IAM ポリシー」を参照してください。
2. 「Amazon DynamoDB デベロッパーガイド」で説明されているように、DynamoDB コンソールから、ポイントインタイムリカバリ (PITR)、リソースポリシー、ID ベースのポリシー、暗号化キーのアクセス許可を持つ DynamoDB テーブルを設定します。

3. Amazon Redshift コンソール: ターゲット Amazon Redshift データウェアハウスを作成して設定する.
   - AWS CLI または Amazon Redshift コンソール: データウェアハウスの大文字と小文字の区別を有効にする.
   - Amazon Redshift コンソール: Amazon Redshift データウェアハウスの認可を設定する.
4. Amazon Redshift コンソールから、このトピックで後述するゼロ ETL 統合を作成します。
5. Amazon Redshift コンソールから、Amazon Redshift データウェアハウスに送信先データベースを作成します。詳細については、「Amazon Redshift でのデスティネーションデータベースの作成」を参照してください。
6. Amazon Redshift コンソールから、Amazon Redshift データウェアハウスでレプリケートされたデータをクエリします。詳細については、「Amazon Redshift でのレプリケートしたデータのクエリ」を参照してください。

このステップでは、Amazon Redshift との Amazon DynamoDB ゼロ ETL 統合を作成します。

Amazon Redshift console

Amazon Redshift コンソールを使用して Amazon DynamoDB ゼロ ETL 統合を作成するには

1. Amazon Redshift コンソールから、[ゼロ ETL 統合]を選択します。ゼロ ETL 統合のリストが表示されているペインで、[ゼロ ETL 統合の作成]、[DynamoDB 統合の作成] を選択します。
2. 統合を作成するページで、統合に関する情報を次のように入力します。
   - [統合名] を入力します - 統合を参照するために使用できる一意の名前となります。
   - [説明] を入力します - ソースからターゲットにレプリケートするデータを示します。
   - DynamoDB の [ソーステーブル] を選択します - 1 つの DynamoDB テーブルを選択できます。テーブルで、ポイントインタイムリカバリ (PITR) を有効化する必要があります。最大 100 テビバイト (TiB) のテーブルサイズのみが表示されます。ソース DynamoDB テーブルは暗号化する必要があります。ソースには、承認されたプリンシパルと統合ソースを含むリソースポリシーも必要です。これらのポリシーが正しくない場合は、[修正する] オプションが表示されます。
   - ターゲットの [Amazon Redshift データウェアハウス] を選択します - データウェアハウスは、Amazon Redshift でプロビジョニングされたクラスターまたは Redshift Serverless ワークグループのどちらでもかまいません。ターゲット Amazon Redshift が同じアカウントにある場合は、ターゲットを選択できます。ターゲットが別のアカウントにある

場合は、[Redshift データウェアハウス ARN] を指定します。ターゲットには、承認されたプリンシパルと統合ソース、および enable_case_sensitive_identifier パラメータが true に設定されているリソースポリシーが必要です。ターゲットに正しいリソースポリシーがなく、かつターゲットが同じアカウントにある場合は、[修正する] オプションを選択して、統合の作成プロセス中にリソースポリシーを自動的に適用できます。ターゲットが別の AWS アカウント にある場合は、Amazon Redshift ウェアハウスにリソースポリシーを手動で適用する必要があります。ターゲットの Amazon Redshift データウェアハウスに true として設定されている正しいパラメータグループオプション enable_case_sensitive_identifier がない場合は、[修正する] オプションを選択して、このパラメータグループを自動的に更新し、統合の作成プロセス中にウェアハウスを再起動できます。

- 最大 50 個のタグ [キー] とオプションの [値] を入力します。これは、統合に関する追加のメタデータを提供します。詳細については、「Amazon Redshift のリソースのタグ付け」を参照してください。
- [暗号化] オプションを選択します - 統合を暗号化します。詳細については、「カスタマーマネージドキーでの DynamoDB 統合の暗号化」を参照してください。

  また、統合を暗号化する際に、[追加の暗号化コンテキスト] を追加することもできます。詳細については、「暗号化コンテキスト」を参照してください。

3. [DynamoDB 統合の作成] を選択できるレビューページが表示されます。
4. 進行状況ページが表示され、ゼロ ETL 統合を作成するための各種タスクの進捗状況を確認できます。
5. 統合が作成され、有効化されたら、統合の詳細ページで [データベースに接続] を選択します。Amazon Redshift データウェアハウスが最初に作成された際に、データベースも作成されました。統合用に別のデータベースを作成するには、ターゲットのデータウェアハウス内にある任意のデータベースに接続する必要があります。[データベースに接続] ページで、最近の接続を使用できるかどうかを判断し、[認証] 方法を選択します。認証方法に応じて、ターゲットの既存のデータベースに接続する情報を入力します。この認証情報には、既存の [データベース名] (通常は dev) と、Amazon Redshift データウェアハウスでデータベースが作成された際に指定された [データベースユーザー] を含めることができます。
6. データベースに接続したら、[統合からデータベースを作成] を選択して、ソースからデータを受け取るデータベースを作成します。データベースを作成する場合は、[統合 ID]、[データウェアハウス名]、および [データベース名] を指定します。

7. 統合ステータスと送信先データベースが Active になると、データは DynamoDB テーブルからターゲットテーブルへのレプリケートを開始します。データをソースに追加すると、ターゲットの Amazon Redshift データウェアハウスに自動的にレプリケートされます。

AWS CLI

AWS CLI を使用して Amazon Redshift と Amazon DynamoDB のゼロ ETL 統合を作成するには、次のオプションで create-integration コマンドを使用します。

- integration-name — 統合の名前を指定します。
- source-arn - DynamoDB ソースの ARN を指定します。
- target-arn - Amazon Redshift でプロビジョニングされたクラスターまたは Redshift Serverless ワークグループターゲットの名前空間 ARN を指定します。

次の例では、統合名、ソース ARN、ターゲット ARN を指定して統合を作成します。統合は暗号化されません。

```
aws redshift create-integration \
--integration-name ddb-integration \
--source-arn arn:aws:dynamodb:us-east-1:123456789012:table/books \
--target-arn arn:aws:redshift:us-east-1:123456789012:namespace:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222

{
    "Status": "creating",
    "IntegrationArn": "arn:aws:redshift:us-east-1:123456789012:integration:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "Errors": [],
    "ResponseMetadata": {
        "RetryAttempts": 0,
        "HTTPStatusCode": 200,
        "RequestId": "132cbe27-fd10-4f0a-aacb-b68f10bb2bfb",
        "HTTPHeaders": {
            "x-amzn-requestid": "132cbe27-fd10-4f0a-aacb-b68f10bb2bfb",
            "date": "Sat, 24 Aug 2024 05:44:08 GMT",
            "content-length": "934",
            "content-type": "text/xml"
        }
    },
    "Tags": [],
```

```
    "CreateTime": "2024-08-24T05:44:08.573Z",
    "KMSKeyId": "arn:aws:kms:us-east-1:123456789012:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE33333",
    "AdditionalEncryptionContext": {},
    "TargetArn": "arn:aws:redshift:us-
east-1:123456789012:namespace:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "IntegrationName": "ddb-integration",
    "SourceArn": "arn:aws:dynamodb:us-east-1:123456789012:table/books"
}
```

次の例では、暗号化にカスタマーマネージドキーを使用して統合を作成します。統合を作成する前に:

- ソース DynamoDB テーブルで、同じアカウント (例では「AccountA」) にカスタマーマネージドキー (例では「CMCMK」) を作成します。
- ユーザー/ロール (例では「RoleA」) を使用して統合が作成され、この KMS キーに対する kms:CreateGrant および kms:DescribeKey のアクセス許可があることを確認します。
- キーポリシーに、以下の内容を追加します。

```
{
    "Sid": "Enable RoleA to create grants with key",
    "Effect": "Allow",
    "Principal": {
        "AWS": "RoleA-ARN"
    },
    "Action": "kms:CreateGrant",
    "Resource": "*",
    "Condition": {
        // Add "StringEquals" condition if you plan to provide additional encryption
 context
        // for the zero-ETL integration. Ensure that the key-value pairs added here
 match
        // the key-value pair you plan to use while creating the integration.
        // Remove this if you don't plan to use additional encryption context
        "StringEquals": {
            "kms:EncryptionContext:context-key1": "context-value1"
        },
        "ForAllValues:StringEquals": {
            "kms:GrantOperations": [
                "Decrypt",
                "GenerateDataKey",
```

```
                "CreateGrant"
            ]
        }
    }
},
{
    "Sid": "Enable RoleA to describe key",
    "Effect": "Allow",
    "Principal": {
        "AWS": "RoleA-ARN"
    },
    "Action": "kms:DescribeKey",
    "Resource": "*"
},
{
    "Sid": "Allow use by RS SP",
    "Effect": "Allow",
    "Principal": {
        "Service": "redshift.amazonaws.com"
           },
    "Action": "kms:CreateGrant",
    "Resource": "*"
}
```

```
aws redshift create-integration \
--integration-name ddb-integration \
--source-arn arn:aws:dynamodb:us-east-1:123456789012:table/books \
--target-arn arn:aws:redshift:us-east-1:123456789012:namespace:a1b2c3d4-5678-90ab-
cdef-EXAMPLE22222 \
--kms-key-id arn:aws:kms:us-east-1:123456789012:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE33333 \
--additional-encryption-context key33=value33  // This matches the condition in the
 key policy.
          {
    "IntegrationArn": "arn:aws:redshift:us-
east-1:123456789012:integration:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "IntegrationName": "ddb-integration",
    "SourceArn": "arn:aws:dynamodb:us-east-1:123456789012:table/books",
    "SourceType": "dynamodb",
    "TargetArn": "arn:aws:redshift:us-
east-1:123456789012:namespace:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "Status": "creating",
    "Errors": [],
```

```
    "CreateTime": "2024-10-02T18:29:26.710Z",
    "KMSKeyId": "arn:aws:kms:us-east-1:123456789012:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE33333",
    "AdditionalEncryptionContext": {
        "key33": "value33"
    },
    "Tags": []
}
```

DynamoDB ゼロ ETL 統合と連携する IAM ポリシー

ゼロ ETL 統合を作成する場合、サインイン認証情報には、DynamoDB アクションと Amazon Redshift アクションの両方、および統合のソースとターゲットとして関係するリソースに対するアクセス許可が必要です。以下は、必要な最小限のアクセス許可を示す例です。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:ListTables"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:GetResourcePolicy",
                "dynamodb:PutResourcePolicy",
                "dynamodb:UpdateContinuousBackups"
            ],
            "Resource": [
            "arn:aws:dynamodb:us-east-1:111122223333:table/my-ddb-table"
            ]
        },
        {
            "Sid": "AllowRedshiftDescribeIntegration",
```

```
            "Effect": "Allow",
            "Action": [
                "redshift:DescribeIntegrations"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowRedshiftCreateIntegration",
            "Effect": "Allow",
            "Action": "redshift:CreateIntegration",
            "Resource": "arn:aws:redshift:us-east-1:111122223333:integration:*"
        },
        {
            "Sid": "AllowRedshiftModifyDeleteIntegration",
            "Effect": "Allow",
            "Action": [
                "redshift:ModifyIntegration",
                "redshift:DeleteIntegration"
            ],
            "Resource": "arn:aws:redshift:us-
east-1:111122223333:integration:<uuid>"
        },
        {
            "Sid": "AllowRedshiftCreateInboundIntegration",
            "Effect": "Allow",
            "Action": "redshift:CreateInboundIntegration",
            "Resource": "arn:aws:redshift:us-
east-1:111122223333:namespace:<uuid>"
        }
    ]
}
```

カスタマーマネージドキーでの DynamoDB 統合の暗号化

DynamoDB ゼロ ETL 統合の作成時に AWS 所有のキー ではなくカスタム KMS キーを指定する場合、キーポリシーは Amazon Redshift サービスプリンシパルに `CreateGrant` アクションへのアクセスを提供する必要があります。さらに、リクエスタアカウントまたはロールが `DescribeKey` および `CreateGrant` アクションを実行するアクセス許可が付与されている必要があります。

次のキーポリシーステートメントの例は、ポリシーに必要なアクセス許可を示しています。例として、アクセス許可の範囲をさらに縮小するためのコンテキストキーなどがあります。

キーポリシーステートメントの例

次のポリシーステートメントでは、リクエスタアカウントまたはロールが KMS キーに関する情報を取得することを許可します。

```
{
   "Effect":"Allow",
   "Principal":{
      "AWS":"arn:aws:iam::{account-ID}:role/{role-name}"
   },
   "Action":"kms:DescribeKey",
   "Resource":"*"
}
```

次のポリシーステートメントは、リクエスタアカウントまたはロールが KMS キーに許可を追加することを許可します。kms:ViaService 条件キーは、KMS キーの使用を Amazon Redshift からのリクエストに制限します。

```
{
   "Effect":"Allow",
   "Principal":{
      "AWS":"arn:aws:iam::{account-ID}:role/{role-name}"
   },
   "Action":"kms:CreateGrant",
   "Resource":"*",
   "Condition":{
      "StringEquals":{
         "kms:EncryptionContext:{context-key}":"{context-value}",
         "kms:ViaService":"redshift.{region}.amazonaws.com"
      },
      "ForAllValues:StringEquals":{
         "kms:GrantOperations":[
            "Decrypt",
            "GenerateDataKey",
            "CreateGrant"
         ]
      }
   }
}
```

次のポリシーステートメントでは、Amazon Redshift サービスプリンシパルが KMS キーに権限を追加することを許可します。

```
{
   "Effect":"Allow",
   "Principal":{
      "Service":"redshift.amazonaws.com"
   },
   "Action":"kms:CreateGrant",
   "Resource":"*",
   "Condition":{
      "StringEquals":{
         "kms:EncryptionContext:{context-key}":"{context-value}",
         "aws:SourceAccount":"{account-ID}"
      },
      "ForAllValues:StringEquals":{
         "kms:GrantOperations":[
            "Decrypt",
            "GenerateDataKey",
            "CreateGrant"
         ]
      },
      "ArnLike":{
         "aws:SourceArn":"arn:aws:*:{region}:{account-ID}:integration:*"
      }
   }
}
```

詳細については、「AWS Key Management Service デベロッパーガイド」の「キーポリシーの作成」を参照してください。

暗号化コンテキスト

ゼロ ETL 統合を暗号化する場合は、キーと値のペアを[追加の暗号化コンテキスト] として追加できます。また、これらのキーと値のペアを追加して、レプリケートされるデータに関するコンテキスト情報を追加できます。詳しくは、AWS Key Management Service デベロッパーガイドの Encryption context を参照してください。

Amazon Redshift は、ユーザーが追加したペアに加えて、以下の暗号化コンテキストのペアを追加します。

- `aws:redshift:integration:arn` - `IntegrationArn`
- `aws:servicename:id` - `Redshift`

これにより、追加できるペアの総数が 8 から 6 に減り、許可の制約の全体的な文字数制限に影響します。詳細については、「AWS Key Management Service デベロッパーガイド」の「Using grant constraints」を参照してください。

アプリケーションとのゼロ ETL 統合を作成する

このステップでは、Amazon Redshift とアプリケーションとのゼロ ETL 統合を作成します。

Amazon Redshift とアプリケーションとのゼロ ETL 統合を作成するには

1. Amazon Redshift コンソール: ターゲット Amazon Redshift データウェアハウスを作成して設定する.
   - AWS CLI または Amazon Redshift コンソール: データウェアハウスの大文字と小文字の区別を有効にする.
   - Amazon Redshift コンソール: Amazon Redshift データウェアハウスの認可を設定する.
2. AWS Glue コンソールから: 「AWS Glue デベロッパーガイド」の説明に従って、統合を作成します。
3. 統合先のデータベースが作成され、データのレプリケーションが開始されたら、レプリケートされたデータのマテリアライズドデータをクエリおよび作成できます。詳細については、「Amazon Redshift でのレプリケートしたデータのクエリ」を参照してください。

アプリケーションとのゼロ ETL 統合の作成の詳細については、「AWS Glue デベロッパーガイド」の「ゼロ ETL 統合」を参照してください。

Amazon Redshift でのデスティネーションデータベースの作成

ソースから Amazon Redshift にデータをレプリケートするには、Amazon Redshift の統合からデータベースを作成する必要があります。

ターゲットの Redshift Serverless ワークグループまたはプロビジョン済みのクラスターに接続して、統合識別子への参照を使用してデータベースを作成します。この識別子は、SVV_INTEGRATION ビューをクエリしたときに `integration_id` に返される値です。

> **Important**
>
> 統合からデータベースを作成する前に、ゼロ ETL 統合を作成して、Amazon Redshift コンソール上で統合が `Active` の状態となっている必要があります。

ソースから Amazon Redshift へのデータのレプリケーションを開始する前に、Amazon Redshift の統合からデータベースを作成します。データベースの作成には Amazon Redshift コンソールまたはクエリエディタ v2 を使用できます。

Amazon Redshift console

1. 左側のナビゲーションペインから、[ゼロ ETL 統合] を選択します。
2. 統合リストから、統合を選択します。
3. プロビジョン済みのクラスターを使用している場合は、まずそのデータベースに接続する必要があります。[Connect to database (データベースに接続)] を選択します。最近の接続を使用するか、新しい接続を作成して接続できます。
4. 統合からデータベースを作成するには、[統合からデータベースを作成] を選択します。
5. [送信先データベース名] を入力します。[統合 ID]と[データウェアハウス名] は事前に入力されています。

   Aurora PostgreSQL、RDS for PostgreSQL、RDS for Oracle、または Oracle Database@AWS ソースの場合、ゼロ ETL 統合の作成時に指定した [ソース名データベース] を入力します。このような場合、最大 100 個のソースデータベースを Amazon Redshift データベースにマッピングできます。
6. [データベースの作成] を選択します。

Amazon Redshift query editor v2

1. Amazon Redshift コンソールに移動し、[クエリエディタ v2] を選択します。
2. 左側のパネルで Amazon Redshift Serverless ワークグループまたは Amazon Redshift でプロビジョニングされたクラスターを選択して、接続します。
3. 統合 ID を取得するには、Amazon Redshift コンソールの統合リストに移動します。

   または、次のコマンドを実行して `integration_id` 値を取得します。

   ```
   SELECT integration_id FROM SVV_INTEGRATION;
   ```
4. その後で、次のコマンドを実行してデータベースを作成します。統合 ID を指定すると、データベースとソース間の接続が確立されます。

   `integration_id` を前のコマンドで返された値に置き換えます。

```
CREATE DATABASE destination_db_name FROM INTEGRATION 'integration_id';
```

ソースが Aurora PostgreSQL、RDS for PostgreSQL、RDS for Oracle、または Oracle Database@AWS の場合は、統合を作成した際に指定したソースインスタンスまたはクラスター内の名前付きデータベースへの参照も含める必要があります。例えば、次のようになります。

```
CREATE DATABASE "destination_db_name" FROM INTEGRATION 'integration_id'
 DATABASE "named_db";
```

ゼロ ETL 統合ターゲットのデータベースの作成の詳細については、「Amazon Redshift データベースデベロッパーガイド」の「CREATE DATABASE」を参照してください。ALTER DATBASE を使用すると、REFRESH INTERVAL などのデータベースパラメータを変更できます。ゼロ ETL 統合ターゲットのデータベースの変更の詳細については、「Amazon Redshift データベースデベロッパーガイド」の「ALTER DATABASE」を参照してください。

> **Note**
>
> 統合から作成したデータベースのデータを更新できるのは、統合ソースのみです。テーブルのスキーマを変更するには、ソース内のテーブルに対して DDL コマンドまたは DML コマンドを実行します。ソースのテーブルに対して DDL コマンドと DML コマンドを実行できます。ただし、送信先データベースでは DDL コマンドと読み取り専用クエリのみを実行できます。

送信先データベースのステータスの表示の詳細については、「ゼロ ETL 統合の表示」を参照してください。

送信先データベースを作成したら、ソースにデータを追加できます。ソースにデータを追加するには、次のいずれかのトピックを参照してください。

- ソースが Aurora の場合は、「Amazon Aurora ユーザーガイド」の「データをソース DB クラスターに追加する」を参照してください。
- ソース が Amazon RDS の場合は、「Amazon RDS ユーザーガイド」の「データをソース DB インスタンスに追加する」を参照してください。

- DynamoDB ソースについては、「Amazon DynamoDB デベロッパーガイド」の「DynamoDB の使用開始」を参照してください。
- アプリケーションソースとのゼロ ETL 統合については、「AWS Glue デベロッパーガイド」の「ゼロ ETL 統合」を参照してください。

Amazon Redshift でのレプリケートしたデータのクエリ

データをソースに追加すると、ほぼリアルタイムで Amazon Redshift データウェアハウスにレプリケートされ、クエリが実行できるようになります。統合メトリクスとテーブル統計の詳細については、ゼロ ETL 統合のメトリクス を参照してください。

> **Note**
>
> データベースは MySQL のスキーマと同じなので、MySQL データベースレベルは Amazon Redshift スキーマレベルにマッピングされます。Aurora MySQL または RDS for MySQL からレプリケートしたデータをクエリする場合は、このマッピングの違いに注意が必要です。

レプリケートされたデータをクエリするには

1. Amazon Redshift コンソールに移動し、[クエリエディタ v2] を選択します。
2. Amazon Redshift Serverless ワークグループまたは Amazon Redshift でプロビジョニングされたクラスターに接続して、ドロップダウンメニューからデータベースを選択します。
3. SELECT ステートメントを使用して、ソースで作成したスキーマとテーブルからレプリケートされたデータをすべて選択します。大文字と小文字を区別するために、スキーマ名、テーブル名、列名を二重引用符 (" ") で囲みます。例えば、次のようになります。

   ```
   SELECT * FROM "schema_name"."table_name";
   ```

   また、Amazon Redshift Data API を使用してデータをクエリすることもできます。

レプリケートしたデータのマテリアライズドビューでのクエリ

ローカルの Amazon Redshift データベースにマテリアライズドビューを作成して、ゼロ ETL 統合でレプリケートしたデータを変換できます。ローカルデータベースに接続して、クロスデータベースクエリを使用して送信先データベースにアクセスします。3 つの部分で構成される表記の完全修

飾オブジェクト名 (destination-database-name.schema-name.table-name) を使用するか、送信先データベースとスキーマのペアを参照する外部スキーマを作成して 2 つの部分で構成される表記 (external-schema-name.table-name) を使用できます。クロスデータベースのクエリの詳細については、「データベース間でのデータのクエリ」を参照してください。

次の例を使用してサンプルデータを作成し、ソースの *tickit_zetl* から *sales_zetl* テーブルと *event_zetl* テーブルにサンプルデータを挿入します。テーブルは Amazon Redshift データベースの *zetl_int_db* にレプリケートされます。

```
CREATE TABLE sales_zetl (
        salesid integer NOT NULL primary key,
        eventid integer NOT NULL,
        pricepaid decimal(8, 2)
);

CREATE TABLE event_zetl (
        eventid integer NOT NULL PRIMARY KEY,
        eventname varchar(200)
);

INSERT INTO sales_zetl VALUES(1, 1, 3.33);
INSERT INTO sales_zetl VALUES(2, 2, 4.44);
INSERT INTO sales_zetl VALUES(3, 2, 5.55);

INSERT INTO event_zetl VALUES(1, "Event 1");
INSERT INTO event_zetl VALUES(2, "Event 2");
```

マテリアライズドビューを作成すると、次の 3 つの部分で構成される表記を使用してイベントごとの総売上高を取得できます。

```
--three part notation zetl-database-name.schema-name.table-name
CREATE MATERIALIZED VIEW mv_transformed_sales_per_event_3p
AUTO REFRESH YES
AS
(SELECT eventname, sum(pricepaid) as total_price
FROM  zetl_int_db.tickit_zetl.sales_zetl S, zetl_int_db.tickit_zetl.event_zetl E
WHERE S.eventid = E.eventid
GROUP BY 1);
```

マテリアライズドビューを作成すると、次の 2 つの部分で構成される表記を使用してイベントごとの総売上高を取得できます。

```
--two part notation external-schema-name.table-name notation
CREATE EXTERNAL schema ext_tickit_zetl
FROM REDSHIFT
DATABASE zetl_int_db
SCHEMA tickit_zetl;

CREATE MATERIALIZED VIEW mv_transformed_sales_per_event_2p
AUTO REFRESH YES
AS
(
    SELECT eventname, sum(pricepaid) as total_price
    FROM  ext_tickit_zetl.sales_zetl S, ext_tickit_zetl.event_zetl E
    WHERE S.eventid = E.eventid
    GROUP BY 1
);
```

作成したマテリアライズドビューを表示するには、次の例を使用します。

```
SELECT * FROM mv_transformed_sales_per_event_3p;

+-----------+-------------+
| eventname | total_price |
+-----------+-------------+
| Event 1   | 3.33        |
| Event 2   | 9.99        |
+-----------+-------------+

SELECT * FROM mv_transformed_sales_per_event_2p;

+-----------+-------------+
| eventname | total_price |
+-----------+-------------+
| Event 1   | 3.33        |
| Event 2   | 9.99        |
+-----------+-------------+
```

DynamoDB からのレプリケートされたデータのクエリ

Amazon DynamoDB から Amazon Redshift データベースにデータをレプリケートすると、SUPER データ型の列のマテリアライズドビューに保存されます。

この例では、次のデータが DynamoDB に保存されます。

```
{
    "key1": {
        "S": "key_1"
    },
    "key2": {
        "N": 0
    },
    "payload": {
        "L": [
            {
                "S": "sale1"
            },
            {
                "S": "sale2"
            },
        ]
    },
}
```

Amazon Redshift マテリアライズドビューは、次のように定義されます。

```
CREATE MATERIALIZED VIEW mv_sales
                    BACKUP NO
                    AUTO REFRESH YES
                    AS
                    SELECT "value"."payload"."L"[0]."S"::VARCHAR AS first_payload
                    FROM public.sales;
```

マテリアライズドビューでデータを表示するには、SQL コマンドを実行します。

```
SELECT first_payload FROM mv_sales;
```

ゼロ ETL 統合の表示

Amazon Redshift コンソールからゼロ ETL 統合を表示できます。ここでは、設定情報と現在のステータスを表示し、画面を開いてデータをクエリおよび共有できます。

Amazon Redshift console

ゼロ ETL 統合の詳細を表示するには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. 左側のナビゲーションペインから、[サーバーレス] または [プロビジョニングされたクラスター] ダッシュボードを選択します。次に、[ゼロ ETL 統合] を選択します。
3. 表示するゼロ ETL 統合を選択します。統合ごとに、次のとおりの情報が提供されます。

   - [統合 ID] は、統合作成時に返される識別子です。
   - [ステータス] は次のいずれかになります。
     - `Active` — ゼロ ETL 統合では、トランザクションデータがターゲットの Amazon Redshift データウェアハウスに送信されます。
     - `Syncing` — ゼロ ETL 統合で回復可能なエラーが発生したため、データを再シードしています。影響を受けるテーブルは、再同期が完了するまで Amazon Redshift でクエリを実行できません。
     - `Failed` — ゼロ ETL 統合で、修正できない回復不能なイベントまたはエラーが発生しました。ゼロ ETL 統合を手動で削除して再作成する必要があります。
     - `Creating` — ゼロ ETL 統合は作成中です。
     - `Deleting` — ゼロ ETL 統合を削除中です。
     - `Needs attention` — ゼロ ETL 統合でイベントまたはエラーが発生したため、解決するには手動介入が必要です。問題を解決するには、エラーメッセージのステップに従ってください。
   - [ソースタイプ] は、ターゲットにレプリケートするソースデータのタイプです。Aurora MySQL 互換エディション、Amazon Aurora PostgreSQL、RDS for MySQL、アプリケーション (`GlueSAAS`) など、他のデータベースマネージャーを指定できます。
   - [ソース ARN] は、ソースデータの ARN です。ほとんどのソースでは、これはソースデータベースまたはテーブルの ARN です。アプリケーションソースとのゼロ ETL 統合の場合、これは AWS Glue 接続オブジェクトの ARN です。
   - [ターゲット] は、ソースデータを受け取る Amazon Redshift データウェアハウスの名前空間です。
   - [データベース] は次のいずれかになります。
     - `No database` – 統合のための送信先データベースがありません。

- `Creating` – Amazon Redshift が統合のための送信先データベースを作成しています。
- `Active` – データが統合ソースから Amazon Redshift にレプリケートされています。
- `Error` – 統合でエラーが発生しました。
- `Recovering` – 統合は、データウェアハウスの再起動後に復旧されます。
- `Resyncing` – Amazon Redshift は統合のテーブルを再同期しています。

- [ターゲットタイプ] は、Amazon Redshift データウェアハウスのタイプです。
- [作成日] – 統合が作成された日時 (UTC) です。

> **Note**
>
> データウェアハウスの統合の詳細を表示するには、プロビジョン済みのクラスターまたはサーバーレス名前空間の [詳細] ページを選択して、[ゼロ ETL 統合] タブをクリックします。

[ゼロ ETL 統合] リストで [データをクエリ] を選択すると、Amazon Redshift クエリエディタ v2 が開きます。Amazon Redshift ターゲットデータベースでは、enable_case_sensitive_identifier パラメータが有効になっています。SQL を作成する際は、スキーマ名、テーブル名、列名を二重引用符で囲む必要がある場合があります ("<名前>")。Amazon Redshift データウェアハウスのデータクエリの詳細については、「クエリエディタ v2 を使用してデータベースのクエリを実行する」を参照してください。

[ゼロ ETL 統合] リストで [データを共有する] を選択すると、データ共有を作成できます。Amazon Redshift データベースのデータ共有を作成するには、[データ共有を作成] ページの指示に従います。Amazon Redshift データベースでデータを共有する前に、まず送信先データベースを作成する必要があります。データ共有の詳細については、「Amazon Redshift のデータ共有概念」を参照してください。

統合を更新するには、ALTER DATABASE コマンドを使用できます。これにより、統合ソースからすべてのデータが送信先データベースにレプリケートされます。次の例では、ゼロ ETL 統合で同期されたテーブルと失敗したテーブルをすべて更新します。

```
ALTER DATABASE sample_integration_db INTEGRATION REFRESH ALL tables;
```

AWS CLI

AWS CLI を使用して Amazon Redshift と Amazon DynamoDB のゼロ ETL 統合を記述するには、次のオプションで `describe-integrations` コマンドを使用します。

- `integration-arn` - 記述する DynamoDB 統合の ARN を指定します。
- `integration-name` - 返すために 1 つ以上のリソースを指定するオプションのフィルターを指定します。

次の例では、統合 ARN を指定して、統合を記述しています。

```
aws redshift describe-integrations

{
    "Integrations": [
        {
            "Status": "failed",
            "IntegrationArn": "arn:aws:redshift:us-
east-1:123456789012:integration:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "Errors": [
                {
                    "ErrorCode": "INVALID_TABLE_PERMISSIONS",
                    "ErrorMessage": "Redshift does not have sufficient access on the
 table key. Refer to the Amazon DynamoDB Developer Guide."
                }
            ],
            "Tags": [],
            "CreateTime": "2023-11-09T00:32:46.444Z",
            "KMSKeyId": "arn:aws:kms:us-east-1:123456789012:key/a1b2c3d4-5678-90ab-
cdef-EXAMPLE33333",
            "TargetArn": "arn:aws:redshift:us-
east-1:123456789012:namespace:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "IntegrationName": "ddb-to-provisioned-02",
            "SourceArn": "arn:aws:dynamodb:us-east-1:123456789012:table/mytable"
        }
    ]
}
```

`integration-arn`、`source-arn`、`source-types`、または `status` で、`describe-integrations` の結果をフィルタリングすることもできます。詳細については、「Amazon Redshift CLI ガイド」の「describe-integrations」を参照してください。

履歴モード

履歴モードを使用すると、ソーステーブル内のレコードのすべてのバージョン (更新と削除を含む) を Amazon Redshift で直接追跡するように、ゼロ ETL 統合を設定できます。履歴分析の実行、ルックバックレポートの構築、トレンド分析の実行、Amazon Redshift 上に構築されたダウンストリームアプリケーションへの増分更新の送信など、すべてのデータに対して高度な分析を実行できます。履歴モードは、Amazon Aurora MySQL、Amazon Aurora PostgreSQL、Amazon RDS for MySQL、Amazon DynamoDB など、複数の Amazon Redshift ゼロ ETL 統合でサポートされています。履歴モードは、Salesforce、SAP、ServiceNow、および Zendesk などの複数のアプリケーションでもサポートされています。

Amazon Redshift コンソール (https://console.aws.amazon.com/redshiftv2/) から、ゼロ ETL 統合の履歴モードをオンまたはオフにできます。統合のソースで削除または変更されたレコードを追跡するには、履歴モードを使用します。追跡は、ターゲット Amazon Redshift データウェアハウスで実行されます。履歴モードをオンにしても、これらのテーブルに対する通常の分析クエリのパフォーマンスへの影響はありません。

履歴モードをオンにすると、ソース内で削除したテーブルは Amazon Redshift では削除されません。代わりに、テーブルは `DroppedSource` 状態になり、これらのテーブルに対して依然としてクエリを実行できます。通常の SQL で DROP コマンドと RENAME コマンドを依然として使用することもできます。

ソースで同じテーブル名を再利用する場合は、Amazon Redshift にレプリケートする前に、対応する `DroppedState` テーブルを DROP または RENAME する必要があります。この操作は、ソースにテーブルを作成する前に行ってください。

履歴モードを使用する際の考慮事項については、「ターゲットで履歴モードを使用する場合の考慮事項」を参照してください。

ゼロ ETL 統合の履歴モードを管理するには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. 左側のナビゲーションペインから、[サーバーレス] または [プロビジョニングされたクラスター] ダッシュボードを選択します。次に、[ゼロ ETL 統合] を選択します。
3. 管理するゼロ ETL 統合を選択して、[履歴モードを管理] をクリックします。[履歴モードを管理] ウィンドウが表示されます。

4. Amazon DynamoDB など、単一のソーステーブルがあるソースタイプからレプリケートされたターゲットテーブルの履歴モードを [オフにする] か [オンにする] ことができます。ゼロ ETL 統合に複数のターゲットテーブルがある場合、[すべての既存のテーブルと今後のテーブルでオフにする]、[すべての既存のテーブルと今後のテーブルでオンにする]、または [個々のテーブルの履歴モードを管理] にすることができます。デフォルトでは、ゼロ ETL 統合の作成時の履歴モードは `off` になっています。

   履歴モードが on に設定されている場合、ソースの変更を追跡するために、次の列がターゲットテーブルに追加されます。履歴モードが on の場合、Amazon Redshift はターゲットテーブルのレコードを削除しないため、毎月の使用料とコストが増加します。削除または変更されたソースレコードがあると、ターゲットに新しいレコードが作成され、複数のレコードバージョンがあるターゲットの合計行数が増えます。ソースでレコードが削除されたり変更されたりしても、ターゲットテーブルからは削除されません。非アクティブなレコードを削除することで、ターゲットテーブルを管理できます。

列名	データ型	説明
_record_is_active	ブール値	ターゲットのレコードがソースで現在アクティブかどうかを示します。True はレコードがアクティブであることを示します。
_record_create_time	タイムスタンプ	ソースレコードがアクティブになった開始時刻 (UTC)。
_record_delete_time	タイムスタンプ	ソースレコードが更新されたか削除された終了時刻 (UTC)。

   列 `_record_is_active` が false であるレコードをフィルタリングすることで、非アクティブなレコードを履歴モードテーブルから削除できます。次の SQL DELETE コマンドは、id 列が 100 以下のテーブルから非アクティブなレコードを削除します。レコードの削除後、自動バキューム削除が実行されると、削除されたレコードのストレージは再利用されます。

```
DELETE FROM myschema.mytable where not _record_is_active AND id <= 100;
```

   履歴モードが `off` になっている場合、Amazon Redshift はアクティブなレコードがあり、追加された履歴列のないテーブルのコピーをターゲットデータベース内に作成します。Amazon Redshift は、テーブルの名前を *`table-name`*`_historical_`*`timestamp`* に変更して、ユー

ザーが使用できるようにします。このテーブルのコピーは、不要になったら削除できます。これらのテーブルの名前は、ALTER TABLE コマンドを使用して変更できます。例えば、次のようになります。

```
ALTER TABLE [schema-name.]table-name_historical_timestamp RENAME TO new_table_name;
```

詳細については、「Amazon Redshift データベースデベロッパーガイド」の「ALTER TABLE」を参照してください。

SQL コマンドの CREATE DATABASE と ALTER DATABASE を使用して、履歴モードを管理することもできます。HISTORY_MODE の設定方法の詳細については、「Amazon Redshift データベースデベロッパーガイド」の「CREATE DATABASE」と「ALTER DATABASE」を参照してください。

Amazon Redshift でのデータの共有

ソースにデータを追加すると、そのデータはすぐに Amazon Redshift にレプリケートされ、データ共有を作成して共有できるようになります。

データを共有するには、まず送信先データベースを作成する必要があります。

Amazon Redshift コンソールを使用して Amazon Redshift Serverless でデータを共有するには

1. Amazon Redshift コンソールの左側のナビゲーションペインで、[Amazon Redshift Serverless] > [サーバーレスダッシュボード] を選択します。
2. 左側のナビゲーションペインから、[ゼロ ETL 統合] を選択します。
3. [データの共有] を選択します。
4. [データ共有を作成] ページで、「データ共有の作成」の手順に従って実行します。

Amazon Redshift コンソールを使用して Amazon Redshift でプロビジョニングされたクラスターを共有するには

1. Amazon Redshift コンソールの左側のナビゲーションペインで、[プロビジョニングされたクラスターダッシュボード] を選択します。
2. 左側のナビゲーションペインから、[ゼロ ETL 統合] を選択します。
3. 統合リストから、統合を選択します。
4. 統合の [詳細] ページで、[データベースに接続] をクリックします。

5. [Connection to database] ページでは、新しい接続を作成することも、最近の接続を使用することもできます。デスティネーションデータベースに接続されていることを確認します。
6. 新しい接続を作成する場合は、データベースの [データベース名] を入力します。次に、[接続] をクリックします。
7. 統合の [詳細] ページで、[データを共有する] をクリックします。
8. [データ共有を作成] ページで、「データ共有の作成」の手順に従って実行します。

ゼロ ETL 統合のモニタリング

システムビューをクエリするか、Amazon EventBridge を使用して、ゼロ ETL 統合をモニタリングできます。

Amazon Redshift システムビューとのゼロ ETL 統合のモニタリング

Amazon Redshift で次のシステムビューをクエリすると、ゼロ ETL 統合をモニタリングできます。

- SVV_INTEGRATION を使用すると、ゼロ ETL 統合の設定の詳細に関する情報が得られます。
- SYS_INTEGRATION_ACTIVITY を使用すると、完了した統合に関する情報が得られます。
- SVV_INTEGRATION_TABLE_MAPPING は、ソースからターゲットへのメタデータ値のマッピングに関する情報を提供します。
- SVV_INTEGRATION_TABLE_STATE を使用すると、統合の状態に関する情報が得られます。
- SYS_INTEGRATION_TABLE_ACTIVITY は、統合の挿入、削除、更新アクティビティに関する情報を提供します。
- SYS_INTEGRATION_TABLE_STATE_CHANGE は、統合のテーブル状態変更ログに関する情報を提供します。

Amazon EventBridge とのゼロ ETL 統合のモニタリング

Amazon Redshift ridge は統合関連のイベントを Amazon EventBridge に送信します。イベントのリストとそれに対応するイベント ID については、「Amazon EventBridge によるイベント通知のゼロ ETL 統合」を参照してください。

ゼロ ETL 統合のメトリクス

Amazon Redshift コンソールと Amazon CloudWatch でメトリクスを使用して、ゼロ ETL 統合のヘルスとパフォーマンスを把握することができます。メトリクスを調整してデータを表示する期間を短く、または長くするか、CloudWatch でのメトリクスの表示を選択できます。Amazon Redshift コンソールで統合のメトリックスを表示するには、左側のナビゲーションペインで [ゼロ ETL 統合] を選択して、[統合 ID] をクリックします。

ゼロ ETL 統合のソースデータによっては、Amazon Redshift は統合の詳細ページでメトリクスを提供します。可能なメトリクスには次の種類があります。

- [統合メトリクス] タブで、以下のグラフを使用できます。

メトリクス	Amazon Redshift コンソールのメトリクス名	説明
`IntegrationLag`	`Lag`	データがソースにコミットされてから、Amazon Redshift でデータがクエリに使用できるようになるまでのラグ 単位: 秒 ディメンション: `IntegrationId` 更新頻度: 60 秒
`IntegrationNumTablesReplicated`	`Tables replicated`	ソースデータベースから Amazon Redshift にレプリケートされたテーブルの数 単位: カウント ディメンション: `IntegrationId` 更新頻度: 60 秒
`IntegrationNumTablesFailedReplication`	`Tables failed`	レプリケーションに失敗したテーブルの数。 単位: カウント

メトリクス	Amazon Redshift コンソールのメトリクス名	説明
		ディメンション: IntegrationId 更新頻度: 60 秒
IntegrationDataTransferred	Data transferred	転送されたデータ量 (論理バイト) です。 単位: バイト ディメンション: IntegrationId 更新頻度: 60 秒
IntegrationLatestDetectedChange	Latest detected change	レプリケーションキューで統合が最後にソース変更をステージングした時刻 (Unix エポック秒単位)。 単位: 秒 ディメンション: IntegrationId 更新頻度: 60 秒
IntegrationLatestAppliedChange	Latest applied change	Amazon Redshift で統合が最後に取り込みを完了した時刻 (Unix エポック秒単位)。 単位: 秒 ディメンション: IntegrationId 更新頻度: 60 秒

メトリクス	Amazon Redshift コンソールのメトリクス名	説明
IntegrationDuplicateRowsDetected	Duplicate rows detected	レポート間隔中に重複する行が識別されたテーブルの数。 単位: カウント ディメンション: IntegrationId 更新頻度: 60 秒
IntegrationAutoRediationTriggered	Auto-remediation triggered	レポート間隔中に自動修復によって再同期に移動されたテーブルの数。 単位: カウント ディメンション: IntegrationId 更新頻度: 60 秒

> **Note**
>
> 複数のデータベースをレプリケートする統合の場合、IntegrationLatestDetectedChange メトリクスと IntegrationLatestAppliedChange メトリクスはすべてのデータベースにわたって最小値を報告します。これは、最も長く更新されていないデータベースを表します。データベースごとの値は、SVV_INTEGRATION の latest_detected_change_time 列と latest_applied_change_time 列で確認できます。

- [テーブル統計] タブから、現在アクティブまたはエラーが発生しているテーブルのリストを表示できます。このタブの統計情報は、次のとおりです (ソースタイプによって異なります)。
  - スキーマ名 – テーブルが含まれるスキーマの名前
  - テーブル名 – ソースデータベース内のテーブルの名前
  - ステータス – テーブルのステータス。指定できる値には、Synced、Failed、Deleted、Resync Requiredおよび Resync Initiated などがあります。

- データベース – テーブルがある Amazon Redshift データベース
- 最終更新日 – テーブルが最後に更新された日時 (UTC)
- テーブル行数 - テーブル内の行数。
- テーブルサイズ - テーブルのサイズ。

選択した期間で、[行] の挿入、削除、更新が行われた件数のグラフを表示することもできます。

DynamoDB のゼロ ETL 統合を変更する

このステップでは、Amazon Redshift との DynamoDB ゼロ ETL 統合を変更します。

Amazon Redshift console

Amazon Redshift コンソールを使用して Amazon DynamoDB ゼロ ETL 統合を変更するには

1. Amazon Redshift コンソールから、[ゼロ ETL 統合]を選択します。ゼロ ETL 統合のリストがあるペインで、変更する DynamoDB 統合を選択します。
2. [編集] を選択し、[統合名] または [説明] を変更します。
3. [変更を保存] を選択して、変更を保存します。

AWS CLI

AWS CLI を使用して Amazon Redshift と Amazon DynamoDB のゼロ ETL 統合を変更するには、次のオプションで `modify-integration` コマンドを使用します。

- `integration-arn` - 変更する DynamoDB 統合の ARN を指定します。
- `integration-name` — 統合の新しい名前を指定します。
- `description` — 統合の新しい説明を指定します。

次の例では、統合 ARN、新しい説明、および新しい名前を指定して統合を変更します。

```
aws redshift modify-integration \
--integration-arn arn:aws:redshift:us-
east-1:123456789012:integration:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
--description "Test modify description and name together." \
--integration-name "updated-integration-name-2"
```

```
{
    "IntegrationArn": "arn:aws:redshift:us-
east-1:123456789012:integration:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "IntegrationName": "updated-integration-name-2",
    "SourceArn": "arn:aws:dynamodb:us-east-1:123456789012:table/ddb-temp-test-table-
table",
    "SourceType": "dynamodb",
    "TargetArn": "arn:aws:redshift:us-
east-1:123456789012:namespace:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "Status": "active",
    "Errors": [],
    "CreateTime": "2024-09-19T18:06:33.555Z",
    "Description": "Test modify description and name together.",
    "KMSKeyId": "arn:aws:kms:us-east-1:123456789012:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE33333",
    "AdditionalEncryptionContext": {},
    "Tags": []
}
```

DynamoDB のゼロ ETL 統合を削除する

統合を削除しても、ターゲットデータウェアハウスは以前にレプリケートされたデータを保持します。このデータの共有とクエリを継続できます。ただし、ソース内の新しいデータは、ターゲットにレプリケートされません。

このステップでは、Amazon Redshift との DynamoDB ゼロ ETL 統合を削除します。

Amazon Redshift console

Amazon Redshift コンソールを使用して Amazon DynamoDB ゼロ ETL 統合を削除するには

1. Amazon Redshift コンソールから、[ゼロ ETL 統合]を選択します。ゼロ ETL 統合のリストがあるペインで、削除する DynamoDB 統合を選択します。
2. [削除] を選択し、必要な情報を入力します。
3. [削除] を選択し、ゼロ ETL 統合を削除します。

AWS CLI

AWS CLI を使用して Amazon Redshift と Amazon DynamoDB のゼロ ETL 統合を削除するには、次のオプションで `delete-integration` コマンドを使用します。

- `integration-arn` - 削除する DynamoDB 統合の ARN を指定します。

次の例では、統合 ARN を指定して、統合を削除しています。

```
aws redshift delete-integration \
--integration-arn arn:aws:redshift:us-
east-1:123456789012:integration:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111

{
    "IntegrationArn": "arn:aws:redshift:us-
east-1:123456789012:integration:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "IntegrationName": "updated-integration-name-2",
    "SourceArn": "arn:aws:dynamodb:us-east-1:123456789012:table/tidal-ddb-ddb-temp-
test-table-table",
    "SourceType": "dynamodb",
    "TargetArn": "arn:aws:redshift:us-
east-1:123456789012:namespace:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "Status": "deleting",
    "Errors": [],
    "CreateTime": "2024-09-19T18:06:33.555Z",
    "Description": "Test modify description and name together.",
    "KMSKeyId": "arn:aws:kms:us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111:key/
a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "AdditionalEncryptionContext": {},
    "Tags": []
}
```

ゼロ ETL 統合をサポートするリージョン

ゼロ ETL 統合はフルマネージドソリューションで、複数の運用ソースやトランザクションソースおよびエンタープライズアプリケーションの運用データとトランザクションデータを Amazon Redshift で利用できるようにします。このページは、サポートされている各ソースで使用できるリージョンを一覧表示しています。

Aurora MySQL

Amazon Redshift との Aurora MySQL ゼロ ETL 統合は、以下のリージョンとエンジンバージョンで利用できます。

リージョン	Aurora MySQL
アフリカ (ケープタウン)	Available
アジアパシフィック (香港)	Available
アジアパシフィック (東京)	Available
アジアパシフィック (ソウル)	Available
アジアパシフィック (大阪)	Available
アジアパシフィック (ムンバイ)	Available
アジアパシフィック (ハイデラバード)	Available
アジアパシフィック (シンガポール)	Available
アジアパシフィック (シドニー)	Available
アジアパシフィック (ジャカルタ)	Available
アジアパシフィック (メルボルン)	Available
アジアパシフィック (マレーシア)	利用不可
カナダ (中部)	Available
カナダ西部 (カルガリー)	Available
中国 (北京)	Available
中国 (寧夏)	Available
欧州 (フランクフルト)	Available
欧州 (チューリッヒ)	Available
欧州 (ストックホルム)	Available
欧州 (ミラノ)	Available

リージョン	Aurora MySQL
欧州 (スペイン)	Available
欧州 (アイルランド)	Available
欧州 (ロンドン)	Available
欧州 (パリ)	Available
イスラエル (テルアビブ)	Available
中東 (アラブ首長国連邦)	Available
中東 (バーレーン)	Available
南米 (サンパウロ)	Available
米国東部 (バージニア北部)	Available
米国東部 (オハイオ)	Available
米国西部 (北カリフォルニア)	Available
米国西部 (オレゴン)	Available
AWS GovCloud (US-East)	利用不可
AWS GovCloud (US-West)	利用不可

Aurora PostgreSQL

Aurora PostgreSQL と Amazon Redshift のゼロ ETL 統合では、以下のリージョンを利用できます。

リージョン	Aurora PostgreSQL
アフリカ (ケープタウン)	Available
アジアパシフィック (香港)	Available

リージョン	Aurora PostgreSQL
アジアパシフィック (東京)	Available
アジアパシフィック (ソウル)	Available
アジアパシフィック (大阪)	Available
アジアパシフィック (ムンバイ)	Available
アジアパシフィック (ハイデラバード)	Available
アジアパシフィック (シンガポール)	Available
アジアパシフィック (シドニー)	Available
アジアパシフィック (ジャカルタ)	Available
アジアパシフィック (メルボルン)	Available
アジアパシフィック (マレーシア)	Available
カナダ (中部)	Available
カナダ西部 (カルガリー)	Available
中国 (北京)	Available
中国 (寧夏)	Available
欧州 (フランクフルト)	Available
欧州 (チューリッヒ)	Available
欧州 (ストックホルム)	Available
欧州 (ミラノ)	Available
欧州 (スペイン)	Available
欧州 (アイルランド)	Available

リージョン	Aurora PostgreSQL
欧州 (ロンドン)	Available
欧州 (パリ)	Available
イスラエル (テルアビブ)	Available
中東 (アラブ首長国連邦)	Available
中東 (バーレーン)	Available
南米 (サンパウロ)	Available
米国東部 (バージニア北部)	Available
米国東部 (オハイオ)	Available
米国西部 (北カリフォルニア)	Available
米国西部 (オレゴン)	Available
AWS GovCloud (US-East)	利用不可
AWS GovCloud (US-West)	利用不可

Amazon DynamoDB

DynamoDB と Amazon Redshift のゼロ ETL 統合では、以下のリージョンで利用できます。

リージョン	DynamoDB
アフリカ (ケープタウン)	Available
アジアパシフィック (香港)	Available
アジアパシフィック (台北)	Available
アジアパシフィック (東京)	Available

リージョン	DynamoDB
アジアパシフィック (ソウル)	Available
アジアパシフィック (大阪)	Available
アジアパシフィック (ムンバイ)	Available
アジアパシフィック (ハイデラバード)	Available
アジアパシフィック (シンガポール)	Available
アジアパシフィック (シドニー)	Available
アジアパシフィック (ジャカルタ)	Available
アジアパシフィック (メルボルン)	Available
アジアパシフィック (マレーシア)	Available
アジアパシフィック (タイ)	Available
カナダ (中部)	Available
カナダ西部 (カルガリー)	Available
中国 (北京)	Available
中国 (寧夏)	Available
欧州 (フランクフルト)	Available
欧州 (チューリッヒ)	Available
欧州 (ストックホルム)	Available
欧州 (ミラノ)	Available
欧州 (スペイン)	Available
欧州 (アイルランド)	Available

リージョン	DynamoDB
欧州 (ロンドン)	Available
欧州 (パリ)	Available
イスラエル (テルアビブ)	Available
中東 (アラブ首長国連邦)	Available
中東 (バーレーン)	Available
メキシコ (中部)	Available
南米 (サンパウロ)	Available
米国東部 (バージニア北部)	Available
米国東部 (オハイオ)	Available
米国西部 (北カリフォルニア)	Available
米国西部 (オレゴン)	利用可能
AWS GovCloud (US-East)	利用可能
AWS GovCloud (US-West)	Available

Amazon RDS for MySQL

Amazon RDS for MySQL と Amazon Redshift のゼロ ETL 統合では、以下のリージョンで利用できます。

リージョン	RDS for MySQL
アフリカ (ケープタウン)	Available
アジアパシフィック (香港)	Available
アジアパシフィック (東京)	Available

リージョン	RDS for MySQL
アジアパシフィック (ソウル)	Available
アジアパシフィック (大阪)	Available
アジアパシフィック (ムンバイ)	Available
アジアパシフィック (ハイデラバード)	利用不可
アジアパシフィック (シンガポール)	Available
アジアパシフィック (シドニー)	Available
アジアパシフィック (ジャカルタ)	利用不可
アジアパシフィック (メルボルン)	利用不可
アジアパシフィック (マレーシア)	利用不可
カナダ (中部)	Available
カナダ西部 (カルガリー)	利用不可
中国 (北京)	利用不可
中国 (寧夏)	利用不可
欧州 (フランクフルト)	Available
欧州 (チューリッヒ)	利用不可
欧州 (ストックホルム)	Available
欧州 (ミラノ)	Available
欧州 (スペイン)	利用不可
欧州 (アイルランド)	Available
欧州 (ロンドン)	Available

リージョン	RDS for MySQL
欧州 (パリ)	Available
イスラエル (テルアビブ)	利用不可
中東 (アラブ首長国連邦)	利用不可
中東 (バーレーン)	Available
南米 (サンパウロ)	Available
米国東部 (バージニア北部)	Available
米国東部 (オハイオ)	Available
米国西部 (北カリフォルニア)	Available
米国西部 (オレゴン)	Available
AWS GovCloud (US-East)	利用不可
AWS GovCloud (US-West)	利用不可

エンタープライズアプリケーション

エンタープライズアプリケーションと Amazon Redshift のゼロ ETL 統合では、以下のリージョンで利用できます。

リージョン	エンタープライズアプリケーション
アフリカ (ケープタウン)	利用不可
アジアパシフィック (香港)	Available
アジアパシフィック (東京)	Available
アジアパシフィック (ソウル)	Available
アジアパシフィック (大阪)	利用不可

リージョン	エンタープライズアプリケーション
アジアパシフィック (ムンバイ)	利用不可
アジアパシフィック (ハイデラバード)	利用不可
アジアパシフィック (シンガポール)	Available
アジアパシフィック (シドニー)	Available
アジアパシフィック (ジャカルタ)	利用不可
アジアパシフィック (メルボルン)	利用不可
アジアパシフィック (マレーシア)	利用不可
カナダ (中部)	Available
カナダ西部 (カルガリー)	利用不可
中国 (北京)	利用不可
中国 (寧夏)	利用不可
欧州 (フランクフルト)	Available
欧州 (チューリッヒ)	利用不可
欧州 (ストックホルム)	Available
欧州 (ミラノ)	利用不可
欧州 (スペイン)	利用不可
欧州 (アイルランド)	Available
欧州 (ロンドン)	Available
欧州 (パリ)	利用不可
イスラエル (テルアビブ)	利用不可

リージョン	エンタープライズアプリケーション
中東 (アラブ首長国連邦)	利用不可
中東 (バーレーン)	利用不可
南米 (サンパウロ)	Available
米国東部 (バージニア北部)	Available
米国東部 (オハイオ)	Available
米国西部 (北カリフォルニア)	利用不可
米国西部 (オレゴン)	Available
AWS GovCloud (US-East)	利用不可
AWS GovCloud (US-West)	利用不可

ゼロ ETL 統合のトラブルシューティング

以下のシナリオを使用して、ゼロ ETL 統合で発生する問題をトラブルシューティングします。

Aurora MySQL とのゼロ ETL 統合のトラブルシューティング

Aurora MySQL とのゼロ ETL 統合に関する一般的な問題のトラブルシューティングを行うには、次の情報を利用してください。

トピック

- 統合の作成に失敗しました
- テーブルにプライマリキーがない
- Aurora MySQL テーブルが Amazon Redshift にレプリケートされない
- テーブル内のサポートされていないデータ型
- データ操作言語コマンドが失敗しました
- データソース間の変更履歴が一致しません
- 認可に失敗しました
- テーブルの数が 10 万を超えているか、スキーマの数が 4950 を超えている

- Amazon Redshift がデータをロードできません
- ワークグループのパラメータ設定が間違っています
- ゼロ ETL 統合を有効にするためにデータベースが作成されていません
- テーブルの状態が、[再同期が必要です] または [再同期開始] になっています
- 統合時の遅延の増加

統合の作成に失敗しました

ゼロ ETL 統合の作成に失敗した場合、統合のステータスは `Inactive` になります。以下がソースの Aurora DB クラスターに当てはまることを確認してください。

- Amazon RDS コンソールでクラスターを作成しました。
- ソースの Aurora DB クラスターはサポートされているバージョンを実行しています。サポートされているバージョンのリストについては、「Amazon Redshift とのゼロ ETL 統合でサポートされているリージョンと Aurora DB エンジン」を参照してください。これを確認するには、クラスターの [設定] タブに移動して、[エンジンのバージョン] を確認します。
- クラスターの binlog パラメータ設定を正しく設定しました。Aurora MySQL バイナリログパラメータが正しく設定されていないか、ソース Aurora DB クラスターに関連付けられていない場合、作成は失敗します。「Configure DB cluster parameters」を参照してください。

さらに、Amazon Redshift データウェアハウスに以下が当てはまることを確認してください。

- 大文字と小文字の区別がオンになっています。「データウェアハウスの大文字と小文字の区別を有効にする」を参照してください。
- 名前空間に適切な認証済みプリンシパルと統合ソースを追加しました。「Amazon Redshift データウェアハウスの認可を設定する」を参照してください。

テーブルにプライマリキーがない

このデスティネーションデータベースには、1 つまたは複数のテーブルにプライマリキーがなく、同期できません。

この問題を解決するには、統合の詳細ページの [テーブル統計] タブに移動するか、SVV_INTEGRATION_TABLE_STATE を使用して、障害が発生したテーブルを確認します。テーブルにプライマリキーを追加すると、Amazon Redshift がテーブルを再同期します。または、テーブルを Aurora にドロップして、プライマリキーを持つテーブルを作成することもできますが、この方

法はお勧めしません。詳細については、「Amazon Redshift best practices for designing tables」を参照してください。

Aurora MySQL テーブルが Amazon Redshift にレプリケートされない

Amazon Redshift に 1 つ以上のテーブルが反映されていない場合は、次のコマンドを実行してテーブルを再同期できます。*dbname* を Amazon Redshift データベースの名前に置き換えます。また、*table1* と *table2* を同期するテーブルの名前に置き換えます。

```
ALTER DATABASE dbname INTEGRATION REFRESH TABLES table1, table2;
```

詳細については、「Amazon Redshift データベースデベロッパーガイド」の「ALTER DATABASE」を参照してください。

1 つ以上のソーステーブルにプライマリキーがないため、データがレプリケートされていない可能性があります。Amazon Redshift のモニタリングダッシュボードには、これらのテーブルのステータスが `Failed` と表示され、ゼロ ETL 統合全体のステータスが `Needs attention` に変わります。この問題を解決するには、プライマリキーとなる既存のキーをテーブル内で特定するか、合成プライマリキーを追加することができます。具体的な解決策については、AWS データベースブログの「Handle tables without primary keys while creating Amazon Aurora MySQL or RDS for MySQL zero-ETL integrations with Amazon Redshift」を参照してください。

また、ターゲットが Amazon Redshift クラスターの場合、クラスターが一時停止していないことを確認します。

テーブル内のサポートされていないデータ型

Amazon Redshift での統合で作成され、Aurora DB クラスターからデータがレプリケートされたデータベースでは、サポートされていないデータ型が単一または複数のテーブルにあるため、同期できません。

この問題を解決するには、統合の詳細ページの [テーブル統計] タブに移動するか、SVV_INTEGRATION_TABLE_STATE を使用して、障害が発生したテーブルを確認します。次に、これらのテーブルを削除し、Amazon RDS に新しいテーブルを再作成します。サポートされていないデータ型の詳細については、「Amazon Aurora ユーザーガイド」の「Aurora データベースと Amazon Redshift データベース間のデータ型の違い」を参照してください。

データ操作言語コマンドが失敗しました

Amazon Redshift は Redshift テーブルで DML コマンドを実行できませんでした。この問題を解決するには、SVV_INTEGRATION_TABLE_STATE を使用して、障害が発生したテーブルを確認します。Amazon Redshift は、このエラーを解決するためにテーブルを自動的に再同期します。

データソース間の変更履歴が一致しません

このエラーは、Amazon Aurora と Amazon Redshift 間の変更が一致せず、統合が `Failed` 状態になる場合に発生します。

この問題を解決するには、ゼロ ETL 統合を削除して Amazon RDS で再度作成してください。詳細については、「Creating zero-ETL integrations」と「Deleting zero-ETL integrations」を参照してください。

認可に失敗しました

ソース Aurora DB クラスターが Amazon Redshift データウェアハウスの認可された統合ソースとして削除されたため、認可に失敗しました。

この問題を解決するには、ゼロ ETL 統合を削除して Amazon RDS で再度作成してください。詳細については、「Creating zero-ETL integrations」と「Deleting zero-ETL integrations」を参照してください。

テーブルの数が 10 万を超えているか、スキーマの数が 4950 を超えている

デスティネーションデータウェアハウスのテーブルの数が 10 万を超えているか、スキーマの数が 4950 を超えています。Amazon Aurora が Amazon Redshift にデータを送信できません。テーブルとスキーマの数が、設定された制限を超えています。この問題を解決するには、ソースデータベースから不要なスキーマまたはテーブルを削除します。

Amazon Redshift がデータをロードできません

Amazon Redshift がゼロ ETL 統合にデータをロードできません。

この問題を解決するには、Amazon RDS のゼロ ETL 統合を削除してから再度作成してください。詳細については、「Creating zero-ETL integrations」と「Deleting zero-ETL integrations」を参照してください。

ワークグループのパラメータ設定が間違っています

ワークグループで大文字と小文字の区別がオンになっていません。

この問題を解決するには、統合の詳細ページの [プロパティ] タブに移動してパラメータグループを選択し、[プロパティ] タブで大文字と小文字を区別する識別子をオンにします。既存のパラメータグループがない場合は、大文字と小文字を区別する識別子をオンにしてパラメータグループを作成してください。次に、Amazon RDS に新しいゼロ ETL 統合を作成します。詳細については、「Creating zero-ETL integrations」を参照してください。

ゼロ ETL 統合を有効にするためにデータベースが作成されていません

ゼロ ETL 統合を有効にするためのデータベースが作成されていません。

この問題を解決するには、統合用のデータベースを作成します。詳細については、「Amazon Redshift でのデスティネーションデータベースの作成」を参照してください。

テーブルの状態が、[再同期が必要です] または [再同期開始] になっています

テーブルの状態が、[再同期が必要です] または [再同期開始] になっています。

テーブルがこのような状態になっている理由に関する詳細なエラー情報を収集するには、SYS_LOAD_ERROR_DETAIL システムビューを使用します。

統合時の遅延の増加

ソースデータベースで SAVEPOINT が頻繁に使用されていると、ゼロ ETL 統合の統合遅延が大きくなる可能性があります。

Aurora PostgreSQL とのゼロ ETL 統合のトラブルシューティング

Aurora PostgreSQL とのゼロ ETL 統合に関する一般的な問題のトラブルシューティングを行うには、次の情報を利用してください。

トピック

- 統合の作成に失敗しました
- テーブルにプライマリキーがない
- Aurora PostgreSQL テーブルが Amazon Redshift にレプリケートされない
- テーブル内のサポートされていないデータ型
- データ操作言語コマンドが失敗しました
- データソース間の変更履歴が一致しません
- 認可に失敗しました
- テーブルの数が 10 万を超えているか、スキーマの数が 4950 を超えている
- Amazon Redshift がデータをロードできません

- ワークグループのパラメータ設定が間違っています
- ゼロ ETL 統合を有効にするためにデータベースが作成されていません
- テーブルの状態が、[再同期が必要です] または [再同期開始] になっています

統合の作成に失敗しました

ゼロ ETL 統合の作成に失敗した場合、統合のステータスは `Inactive` になります。以下がソースの Aurora DB クラスターに当てはまることを確認してください。

- Amazon RDS コンソールでクラスターを作成しました。
- ソースの Aurora DB クラスターはサポートされているバージョンを実行しています。サポートされているバージョンのリストについては、「Amazon Redshift とのゼロ ETL 統合でサポートされているリージョンと Aurora DB エンジン」を参照してください。これを確認するには、クラスターの [設定] タブに移動して、[エンジンのバージョン] を確認します。
- クラスターの binlog パラメータ設定を正しく設定しました。Aurora PostgreSQL binlog パラメータの設定が適切でないか、ソースの Aurora DB クラスターに関連付けられていない場合、作成は失敗します。「Configure DB cluster parameters」を参照してください。

さらに、Amazon Redshift データウェアハウスに以下が当てはまることを確認してください。

- 大文字と小文字の区別がオンになっています。「データウェアハウスの大文字と小文字の区別を有効にする」を参照してください。
- endterm="zero-etl-using.redshift-iam.title"/> に、正しい承認されたプリンシパルと統合ソースを追加しました。

テーブルにプライマリキーがない

このデスティネーションデータベースには、1 つまたは複数のテーブルにプライマリキーがなく、同期できません。

この問題を解決するには、統合の詳細ページの [テーブル統計] タブに移動するか、SVV_INTEGRATION_TABLE_STATE を使用して、障害が発生したテーブルを確認します。テーブルにプライマリキーを追加すると、Amazon Redshift がテーブルを再同期します。または、テーブルを Aurora にドロップして、プライマリキーを持つテーブルを作成することもできますが、この方法はお勧めしません。詳細については、「Amazon Redshift best practices for designing tables」を参照してください。

Aurora PostgreSQL テーブルが Amazon Redshift にレプリケートされない

Amazon Redshift に 1 つ以上のテーブルが反映されていない場合は、次のコマンドを実行してテーブルを再同期できます。*dbname* を Amazon Redshift データベースの名前に置き換えます。また、*table1* と *table2* を同期するテーブルの名前に置き換えます。

```
ALTER DATABASE dbname INTEGRATION REFRESH TABLES table1, table2;
```

詳細については、「Amazon Redshift データベースデベロッパーガイド」の「ALTER DATABASE」を参照してください。

1 つ以上のソーステーブルにプライマリキーがないため、データがレプリケートされていない可能性があります。Amazon Redshift のモニタリングダッシュボードには、これらのテーブルのステータスが `Failed` と表示され、ゼロ ETL 統合全体のステータスが `Needs attention` に変わります。この問題を解決するには、プライマリキーとなる既存のキーをテーブル内で特定するか、合成プライマリキーを追加することができます。具体的な解決策については、AWS データベースブログの「Handle tables without primary keys while creating Amazon Aurora PostgreSQL zero-ETL integrations with Amazon Redshift」を参照してください。

また、ターゲットが Amazon Redshift クラスターの場合、クラスターが一時停止していないことを確認します。

テーブル内のサポートされていないデータ型

Amazon Redshift での統合で作成され、Aurora DB クラスターからデータがレプリケートされたデータベースでは、サポートされていないデータ型が単一または複数のテーブルにあるため、同期できません。

この問題を解決するには、統合の詳細ページの [テーブル統計] タブに移動するか、SVV_INTEGRATION_TABLE_STATE を使用して、障害が発生したテーブルを確認します。次に、これらのテーブルを削除し、Amazon RDS に新しいテーブルを再作成します。サポートされていないデータ型の詳細については、「Amazon Aurora ユーザーガイド」の「Aurora データベースと Amazon Redshift データベース間のデータ型の違い」を参照してください。

データ操作言語コマンドが失敗しました

Amazon Redshift は Redshift テーブルで DML コマンドを実行できませんでした。この問題を解決するには、SVV_INTEGRATION_TABLE_STATE を使用して、障害が発生したテーブルを確認します。Amazon Redshift は、このエラーを解決するためにテーブルを自動的に再同期します。

データソース間の変更履歴が一致しません

このエラーは、Amazon Aurora と Amazon Redshift 間の変更が一致せず、統合が `Failed` 状態になる場合に発生します。

この問題を解決するには、ゼロ ETL 統合を削除して Amazon RDS で再度作成してください。詳細については、「Creating zero-ETL integrations」と「Deleting zero-ETL integrations」を参照してください。

認可に失敗しました

ソース Aurora DB クラスターが Amazon Redshift データウェアハウスの認可された統合ソースとして削除されたため、認可に失敗しました。

この問題を解決するには、ゼロ ETL 統合を削除して Amazon RDS で再度作成してください。詳細については、「Creating zero-ETL integrations」と「Deleting zero-ETL integrations」を参照してください。

テーブルの数が 10 万を超えているか、スキーマの数が 4950 を超えている

デスティネーションデータウェアハウスのテーブルの数が 10 万を超えているか、スキーマの数が 4950 を超えています。Amazon Aurora が Amazon Redshift にデータを送信できません。テーブルとスキーマの数が、設定された制限を超えています。この問題を解決するには、ソースデータベースから不要なスキーマまたはテーブルを削除します。

Amazon Redshift がデータをロードできません

Amazon Redshift がゼロ ETL 統合にデータをロードできません。

この問題を解決するには、Amazon RDS のゼロ ETL 統合を削除してから再度作成してください。詳細については、「Creating zero-ETL integrations」と「Deleting zero-ETL integrations」を参照してください。

ワークグループのパラメータ設定が間違っています

ワークグループで大文字と小文字の区別がオンになっていません。

この問題を解決するには、統合の詳細ページの [プロパティ] タブに移動してパラメータグループを選択し、[プロパティ] タブで大文字と小文字を区別する識別子をオンにします。既存のパラメータグループがない場合は、大文字と小文字を区別する識別子をオンにしてパラメータグループを作成してください。次に、Amazon RDS に新しいゼロ ETL 統合を作成します。詳細については、「Creating zero-ETL integrations」を参照してください。

ゼロ ETL 統合を有効にするためにデータベースが作成されていません

ゼロ ETL 統合を有効にするためのデータベースが作成されていません。

この問題を解決するには、統合用のデータベースを作成します。詳細については、「Amazon Redshift でのデスティネーションデータベースの作成」を参照してください。

テーブルの状態が、[再同期が必要です] または [再同期開始] になっています

テーブルの状態が、[再同期が必要です] または [再同期開始] になっています。

テーブルがこのような状態になっている理由に関する詳細なエラー情報を収集するには、SYS_LOAD_ERROR_DETAIL システムビューを使用します。

RDS for MySQL とのゼロ ETL 統合のトラブルシューティング

RDS for MySQL とのゼロ ETL 統合に関する一般的な問題のトラブルシューティングを行うには、次の情報を利用してください。

トピック

- 統合の作成に失敗しました
- テーブルにプライマリキーがない
- RDS for MySQL テーブルが Amazon Redshift にレプリケートされない
- テーブル内のサポートされていないデータ型
- データ操作言語コマンドが失敗しました
- データソース間の変更履歴が一致しません
- 認可に失敗しました
- テーブルの数が 10 万を超えているか、スキーマの数が 4950 を超えている
- Amazon Redshift がデータをロードできません
- ワークグループのパラメータ設定が間違っています
- ゼロ ETL 統合を有効にするためにデータベースが作成されていません
- テーブルの状態が、[再同期が必要です] または [再同期開始] になっています

統合の作成に失敗しました

ゼロ ETL 統合の作成に失敗した場合、統合のステータスは `Inactive` になります。ソースの RDS DB インスタンスで次が適切であるかを確認します。

- Amazon RDS コンソールでインスタンスを作成しました。
- ソース RDS DB インスタンスは、RDS for MySQL のサポートされているバージョンを実行しています。サポートされているバージョンのリストについては、「Amazon RDS と Amazon Redshift のゼロ ETL 統合でサポートされているリージョンと DB エンジン」を参照してください。これを確認するには、クラスターの [設定] タブに移動して、[エンジンのバージョン] をチェックします。
- クラスインスタンスの binlog パラメータ設定が適切に設定されています。RDS for MySQL binlog パラメータの設定が適切でないか、ソースの RDS DB インスタンスに関連付けられていない場合、作成は失敗します。「DB インスタンスパラメータを設定する」を参照してください。

さらに、Amazon Redshift データウェアハウスに以下が当てはまることを確認してください。

- 大文字と小文字の区別がオンになっています。「データウェアハウスの大文字と小文字の区別を有効にする」を参照してください。
- 名前空間に適切な認証済みプリンシパルと統合ソースを追加しました。「Amazon Redshift データウェアハウスの認可を設定する」を参照してください。

テーブルにプライマリキーがない

このデスティネーションデータベースには、1 つまたは複数のテーブルにプライマリキーがなく、同期できません。

この問題を解決するには、統合の詳細ページの [テーブル統計] タブに移動するか、SVV_INTEGRATION_TABLE_STATE を使用して、障害が発生したテーブルを確認します。テーブルにプライマリキーを追加すると、Amazon Redshift がテーブルを再同期します。または、テーブルを RDS にドロップして、プライマリキーを持つテーブルを作成することもできますが、この方法はお勧めしません。詳細については、「Amazon Redshift best practices for designing tables」を参照してください。

RDS for MySQL テーブルが Amazon Redshift にレプリケートされない

Amazon Redshift に 1 つ以上のテーブルが反映されていない場合は、次のコマンドを実行してテーブルを再同期できます。*dbname* を Amazon Redshift データベースの名前に置き換えます。また、*table1* と *table2* を同期するテーブルの名前に置き換えます。

```
ALTER DATABASE dbname INTEGRATION REFRESH TABLES table1, table2;
```

詳細については、「Amazon Redshift データベースデベロッパーガイド」の「ALTER DATABASE」を参照してください。

1 つ以上のソーステーブルにプライマリキーがないため、データがレプリケートされていない可能性があります。Amazon Redshift のモニタリングダッシュボードには、これらのテーブルのステータスが `Failed` と表示され、ゼロ ETL 統合全体のステータスが `Needs attention` に変わります。この問題を解決するには、プライマリキーとなる既存のキーをテーブル内で特定するか、合成プライマリキーを追加することができます。詳細な解決策については、AWS データベースブログの「Amazon Aurora MySQL または Amazon RDS for MySQL と Amazon Redshift とのゼロ ETL 統合を作成する際に、プライマリキーがないテーブルを処理する」を参照してください。

また、ターゲットが Amazon Redshift クラスターの場合、クラスターが一時停止していないことを確認します。

テーブル内のサポートされていないデータ型

Amazon Redshift での統合で作成され、RDS DB インスタンスからデータがレプリケートされたデータベースでは、サポートされていないデータ型が単一または複数のテーブルにあるため、同期できません。

この問題を解決するには、統合の詳細ページの [テーブル統計] タブに移動するか、SVV_INTEGRATION_TABLE_STATE を使用して、障害が発生したテーブルを確認します。次に、これらのテーブルを削除し、Amazon RDS に新しいテーブルを再作成します。サポートされていないデータ型の詳細については、「Amazon RDS ユーザーガイド」の「RDS データベースと Amazon Redshift データベース間のデータ型の違い」を参照してください。

データ操作言語コマンドが失敗しました

Amazon Redshift は Redshift テーブルで DML コマンドを実行できませんでした。この問題を解決するには、SVV_INTEGRATION_TABLE_STATE を使用して、障害が発生したテーブルを確認します。Amazon Redshift は、このエラーを解決するためにテーブルを自動的に再同期します。

データソース間の変更履歴が一致しません

このエラーは、Amazon Aurora と Amazon Redshift 間の変更が一致せず、統合が `Failed` 状態になる場合に発生します。

この問題を解決するには、ゼロ ETL 統合を削除して Amazon RDS で再度作成してください。詳細については、「Creating zero-ETL integrations」と「Deleting zero-ETL integrations」を参照してください。

認可に失敗しました

ソース RDS DB インスタンスが Amazon Redshift データウェアハウスの認可された統合ソースとして削除されたため、認可に失敗しました。

この問題を解決するには、ゼロ ETL 統合を削除して Amazon RDS で再度作成してください。詳細については、「Creating zero-ETL integrations」と「Deleting zero-ETL integrations」を参照してください。

テーブルの数が 10 万を超えているか、スキーマの数が 4950 を超えている

デスティネーションデータウェアハウスのテーブルの数が 10 万を超えているか、スキーマの数が 4950 を超えています。Amazon Aurora が Amazon Redshift にデータを送信できません。テーブルとスキーマの数が、設定された制限を超えています。この問題を解決するには、ソースデータベースから不要なスキーマまたはテーブルを削除します。

Amazon Redshift がデータをロードできません

Amazon Redshift がゼロ ETL 統合にデータをロードできません。

この問題を解決するには、Amazon RDS のゼロ ETL 統合を削除してから再度作成してください。詳細については、「Creating zero-ETL integrations」と「Deleting zero-ETL integrations」を参照してください。

ワークグループのパラメータ設定が間違っています

ワークグループで大文字と小文字の区別がオンになっていません。

この問題を解決するには、統合の詳細ページの [プロパティ] タブに移動してパラメータグループを選択し、[プロパティ] タブで大文字と小文字を区別する識別子をオンにします。既存のパラメータグループがない場合は、大文字と小文字を区別する識別子をオンにしてパラメータグループを作成してください。次に、Amazon RDS に新しいゼロ ETL 統合を作成します。詳細については、「Creating zero-ETL integrations」を参照してください。

ゼロ ETL 統合を有効にするためにデータベースが作成されていません

ゼロ ETL 統合を有効にするためのデータベースが作成されていません。

この問題を解決するには、統合用のデータベースを作成します。詳細については、「Amazon Redshift でのデスティネーションデータベースの作成」を参照してください。

テーブルの状態が、[再同期が必要です] または [再同期開始] になっています

テーブルの状態が、[再同期が必要です] または [再同期開始] になっています。

テーブルがこのような状態になっている理由に関する詳細なエラー情報を収集するには、SYS_LOAD_ERROR_DETAIL システムビューを使用します。

DynamoDB とのゼロ ETL 統合のトラブルシューティング

以下の情報を使用して、Amazon DynamoDB でのゼロ ETL 統合に関する一般的な問題をトラブルシューティングしてください。

トピック

- 統合の作成に失敗しました
- テーブル内のサポートされていないデータ型
- サポートされていないテーブル名と属性名
- 認可に失敗しました
- Amazon Redshift がデータをロードできません
- ワークグループまたはクラスターのパラメータ設定が間違っています
- ゼロ ETL 統合を有効にするためにデータベースが作成されていません
- ソース DynamoDB テーブルで、ポイントインタイムリカバリ (PITR) が有効になっていません
- KMS キーアクセスが拒否されました
- Amazon Redshift が DynamoDB テーブルキーにアクセスできません

統合の作成に失敗しました

ゼロ ETL 統合の作成に失敗した場合、統合のステータスは `Inactive` になります。Amazon Redshift データウェアハウスとソース DynamoDB テーブルが、以下のとおりであることを確認してください。

- データウェアハウスの大文字と小文字の区別を有効にする。「Amazon Redshift 管理ガイド」の「大文字と小文字の区別を有効にする」を参照してください。
- Amazon Redshift の名前空間に適切な認証済みプリンシパルと統合ソースを追加しました。「Amazon Redshift 管理ガイド」の「Amazon Redshift データウェアハウスの認可を設定する」を参照してください。
- ソース DynamoDB テーブルに正しいリソースベースのポリシーを追加しました。「IAM ユーザーガイド」の「IAM でのポリシーとアクセス許可」を参照してください。

テーブル内のサポートされていないデータ型

DynamoDB 番号は、Amazon Redshift で DECIMAL (38,10) に変換されます。この精度範囲を超える数値は、自動的に (38,10) に変換されます。統合を削除し、数値の精度を統一してから、統合を再作成します。

サポートされていないテーブル名と属性名

Amazon Redshift は、最大 127 文字のテーブルと属性名をサポートしています。DynamoDB テーブル名、パーティションキー、ソートキー列名など、名前が長いために統合に失敗した場合は、短い名前を使用して修正し、統合を再作成します。

認可に失敗しました

ソース DynamoDB インスタンスが Amazon Redshift データウェアハウスの認可された統合ソースとして削除された場合、認可に失敗することがあります。

この問題を解決するには、ゼロ ETL 統合を削除し、Amazon DynamoDB を使用して再作成します。

Amazon Redshift がデータをロードできません

Amazon Redshift がゼロ ETL 統合からデータをロードできません。

この問題を解決するには、ALTER DATABASE で統合を更新します。

```
ALTER DATABASE sample_integration_db INTEGRATION REFRESH ALL TABLES
```

ワークグループまたはクラスターのパラメータ設定が間違っています

ワークグループまたはクラスターで大文字と小文字の区別がオンになっていません。

この問題を解決するには、統合の詳細ページの [プロパティ] タブに移動してパラメータグループを選択し、[プロパティ] タブで大文字と小文字を区別する識別子をオンにします。既存のパラメータグループがない場合は、大文字と小文字を区別する識別子をオンにしてパラメータグループを作成してください。次に、DynamoDB で新しいゼロ ETL 統合を作成します。「Amazon Redshift 管理ガイド」の「大文字と小文字の区別を有効にする」を参照してください。

ゼロ ETL 統合を有効にするためにデータベースが作成されていません

ゼロ ETL 統合を有効にするためのデータベースが作成されていません。

この問題を解決するには、統合用のデータベースを作成します。「Amazon Redshift 管理ガイド」の「Amazon Redshift での送信先データベースの作成」を参照してください。

ソース DynamoDB テーブルで、ポイントインタイムリカバリ (PITR) が有効になっていません

DynamoDB がデータをエクスポートするには、PITR を有効にする必要があります。PITR が常に有効になっていることを確認します。統合が有効な場合に PITR をオフにした場合は、エラーメッセージの指示に従って、ALTER DATABASE を使用して統合を更新する必要があります。

```
ALTER DATABASE sample_integration_db INTEGRATION REFRESH ALL TABLES
```

KMS キーアクセスが拒否されました

ソーステーブルまたは統合に使用される KMS キーは、十分なアクセス許可を設定する必要があります。テーブルの暗号化または復号の詳細については、「Amazon DynamoDB デベロッパーガイド」の「DynamoDB の保管時の暗号化」を参照してください。

Amazon Redshift が DynamoDB テーブルキーにアクセスできません

ソーステーブルの暗号化が AWS マネージドキー の場合は、AWS 所有のキー またはカスタマーマネージドキーに切り替えてください。テーブルがカスタマーマネージドキーで既に暗号化されている場合は、ポリシーに条件キーが含まれていないことを確認してください。

アプリケーションとのゼロ ETL 統合のトラブルシューティング

以下の情報を参考にして、Salesforce、SAP、ServiceNow、Zendesk などのアプリケーションとのゼロ ETL 統合に関してよく起きる問題をトラブルシューティングしてください。

トピック

- 統合の作成に失敗しました
- テーブルが Amazon Redshift にレプリケートされない
- テーブル内のサポートされていないデータ型
- ワークグループのパラメータ設定が間違っています
- ゼロ ETL 統合を有効にするためにデータベースが作成されていません
- テーブルの状態が、[再同期が必要です] または [再同期開始] になっています

統合の作成に失敗しました

ゼロ ETL 統合の作成に失敗した場合、統合のステータスは `Inactive` になります。Amazon Redshift データウェアハウスに関して、以下の点が正しいことを確認してください。

- 大文字と小文字の区別がオンになっています。「データウェアハウスの大文字と小文字の区別を有効にする」を参照してください。
- 名前空間に適切な認証済みプリンシパルと統合ソースを追加しました。「Amazon Redshift データウェアハウスの認可を設定する」を参照してください。

テーブルが Amazon Redshift にレプリケートされない

このデスティネーションデータベースには、1 つまたは複数のテーブルにプライマリキーがなく、同期できません。

この問題を解決するには、統合の詳細ページの [テーブル統計] タブに移動するか、SVV_INTEGRATION_TABLE_STATE を使用して、障害が発生したテーブルを確認します。テーブルにプライマリキーを追加すると、Amazon Redshift がテーブルを再同期します。次のコマンドを実行して再同期できます。*dbname* を Amazon Redshift データベースの名前に置き換えます。また、*table1* と *table2* を同期するテーブルの名前に置き換えます。

```
ALTER DATABASE dbname INTEGRATION REFRESH TABLES table1, table2;
```

詳細については、「Amazon Redshift データベースデベロッパーガイド」の「ALTER DATABASE」を参照してください。

テーブル内のサポートされていないデータ型

Amazon Redshift で統合を通じて作成され、アプリケーションとのゼロ ETL 統合でデータがレプリケートされているデータベースで、1 つ以上のテーブルにサポート対象外のデータ型が含まれているため、同期できません。

この問題を解決するには、統合の詳細ページの [テーブル統計] タブに移動するか、SVV_INTEGRATION_TABLE_STATE を使用して、障害が発生したテーブルを確認します。次に、これらのテーブルを削除し、ソースで新しいテーブルを再作成します。詳細については、「AWS Glue デベロッパーガイド」の「ゼロ ETL 統合」を参照してください。

ワークグループのパラメータ設定が間違っています

ワークグループで大文字と小文字の区別がオンになっていません。

この問題を解決するには、統合の詳細ページの [プロパティ] タブに移動してパラメータグループを選択し、[プロパティ] タブで大文字と小文字を区別する識別子をオンにします。既存のパラメータグループがない場合は、大文字と小文字を区別する識別子をオンにしてパラメータグループを作成して

ください。次に、新しいゼロ ETL 統合を作成します。詳細については、「AWS Glue デベロッパーガイド」の「ゼロ ETL 統合」を参照してください。

ゼロ ETL 統合を有効にするためにデータベースが作成されていません

ゼロ ETL 統合を有効にするためのデータベースが作成されていません。

この問題を解決するには、統合用のデータベースを作成します。詳細については、「Amazon Redshift でのデスティネーションデータベースの作成」を参照してください。

テーブルの状態が、[再同期が必要です] または [再同期開始] になっています

テーブルの状態が、[再同期が必要です] または [再同期開始] になっています。

テーブルがこのような状態になっている理由に関する詳細なエラー情報を収集するには、SYS_LOAD_ERROR_DETAIL システムビューを使用します。

データベースのクエリ

Amazon Redshift クラスターによってホストされているデータベースをクエリする方法は 2 つあります。

- クラスターに接続し、クエリエディタで AWS マネジメントコンソール に対してクエリを実行します。

  Amazon Redshift コンソールでクエリエディタを使用する場合、SQL クライアントアプリケーションをダウンロードしてセットアップする必要はありません。

- SQL Workbench/J などの SQL クライアントツールを介してクラスターに接続します。

  Amazon Redshift は、Java Database Connectivity (JDBC) および Open Database Connectivity (ODBC) を介して接続する SQL クライアントツールをサポートします。Amazon Redshift は SQL クライアントツールまたはライブラリを提供もインストールもしないため、それらを使用してクラスター内のデータを操作するには、クライアントコンピュータまたは Amazon EC2 インスタンスにインストールする必要があります。JDBC または ODBC ドライバーをサポートするほとんどの SQL クライアントツールを使用できます。

> **Note**
>
> ストアドプロシージャを記述する場合は、機密の値を保護するためのベストプラクティスに従うことをお勧めします。
> ストアドプロシージャロジックに機密情報をハードコーディングしないでください。例えば、ストアドプロシージャの本文の CREATE USER ステートメントにユーザーパスワードを割り当てないでください。ハードコードした値は、カタログテーブルにスキーマメタデータとして記録される可能性があるため、セキュリティ上のリスクが生じます。代わりに、パスワードなどの機密の値は、パラメータを使用して引数として、ストアドプロシージャに渡します。
> ストアドプロシージャの詳細については、「CREATE PROCEDURE」と「Amazon Redshift のストアドプロシージャの作成」を参照してください。カタログテーブルの詳細については、「システムカタログテーブル」を参照してください。

Amazon Redshift に接続する

次の構文を使用してデータベースに接続できます。

```
cluster-name.account-number.aws-region.redshift.amazonaws.com/database-name
```

構文要素は次のように定義します。

- `cluster-name`

  クラスターの名前。
- `account-number`

  特定の AWS リージョンで AWS アカウント番号に関連付けられている一意の識別子。特定の AWS リージョンで特定のアカウントによって作成したすべてのクラスターは同じ `account-number` を持ちます。
- `aws-region`

  クラスターがある AWS リージョンのコード。
- `database-name`

  データベースの名前。

例えば、次の接続文字列は、us-east-1 AWS リージョン における `my-cluster` クラスター内の `my-db` データベースを指定します。

```
my-cluster.123456789012.us-east-1.redshift.amazonaws.com/my-db
```

クエリエディタ v2 を使用してデータベースのクエリを実行する

クエリエディタ v2 は、Amazon Redshift データウェアハウスでクエリを作成および実行するために使用する、独立したウェブベースの SQL クライアントアプリケーションです。クエリエディタ v2 は、主にクエリの編集と実行、結果の視覚化、チームとの作業の共有に使用されます。クエリエディタ v2 を使用すると、データベース、スキーマ、テーブル、およびユーザー定義関数 (UDF) を作成できます。ツリービューパネルでは、各データベースのスキーマが表示されます。スキーマごとに、テーブル、ビュー、UDF、および保存する手順が表示されます。クエリエディタ v2 は、以前のクエリエディタに変わるエディタです。

> **Note**
>
> クエリエディタ v2 は、商用の AWS リージョン で使用できます。クエリエディタ v2 が利用できる AWS リージョン の一覧については、Amazon Web Services 全般のリファレンスの「Redshift クエリエディタ v2」に記載されているエンドポイントを参照してください。

クエリエディタ v2 のデモについては、次の動画をご覧ください。Amazon Redshift query editor v2 (Amazon Redshift クエリエディタ v2)。

データ分析のデモについては、以下の動画をご覧ください。Data analysis using Amazon Redshift query editor v2 (Amazon Redshift クエリエディタ v2 を使用したデータ分析)。

クエリエディタ v2 を使用して分離されたか、または共有された接続で複数のクエリを実行するデモについては、次の動画をご覧ください。クエリエディタ v2 を使用した同時クエリ実行。

クエリエディタ v2 には、SQL 文を管理および実行するための豊富な機能セットがあります。次のセクションのトピックでは、これらの多くの機能について説明します。クエリエディタ v2 を使い慣れて、その機能を理解してください。

AWS アカウントの設定

この一連のタスクを実行して、クエリエディタ v2 を設定して Amazon Redshift データベースをクエリできます。適切なアクセス許可があれば、現在の AWS リージョンにある AWS アカウントが所有する Amazon Redshift クラスターまたはワークグループのデータにアクセスできます。

管理者が AWS アカウントのために最初にクエリエディタ v2 を設定するときは、クエリエディタ v2 のリソースを暗号化するために使用する AWS KMS key を選択します。デフォルトでは、AWS 所有キーは、リソースを暗号化するために使用されます。または、管理者はカスタマーマネージドキーを使用することもできます。その場合は、設定ページでキーの Amazon リソースネーム (ARN) を選択します。

アカウントの設定後、AWS KMS 暗号化の設定は変更できません。クエリエディタ v2 での顧客管理キーの作成と使用の詳細については、クエリエディタ V2 で使用する AWS KMS 顧客管理キーの作成を参照してください。管理者は、ファイルからのデータのロードなど、一部の機能に使用する S3 バケットとパスをオプションで選択することもできます。詳細については、「ローカルファイル設定とワークフローからのデータのロード」を参照してください。

Amazon Redshift クエリエディタ v2 は、認証、暗号化、分離、コンプライアンスをサポートし、保管中のデータと転送中のデータを安全に保ちます。データセキュリティとクエリエディタ v2 の詳細については、以下を参照してください。

- 保管中の暗号化
- 送信中の暗号化
- Amazon Redshift での設定と脆弱性の分析

AWS CloudTrail は、AWS アカウント により、またはそのアカウントに代わって行われた API コールや関連イベントを取得し、指定した Amazon S3 バケットにログファイルを配信します。AWS を呼び出したユーザーとアカウント、呼び出し元のソース IP アドレス、および呼び出しの発生日時を特定できます。AWS CloudTrail におけるクエリエディタ v2 の動作の詳細については、「CloudTrail による口グ記録」を参照してください。CloudTrail の詳細については、「AWS CloudTrail ユーザーガイド」を参照してください。

クエリエディタ v2 には、リソースの一部に対して調整可能なクォータがあります。詳細については、「Amazon Redshift オブジェクトのクォータ」を参照してください。

クエリエディタ v2 で作成されたリソース

クエリエディタ v2 では、保存されたクエリやグラフなどのリソースを作成できます。クエリエディタ v2 のすべてのリソースは、IAM ロールまたはユーザーに関連付けられています。IAM ロールにポリシーをアタッチし、そのロールをユーザーに割り当てることをお勧めします。

クエリエディタ v2 では、保存されたクエリとグラフのタグを追加および削除できます。これらのタグは、カスタム IAM ポリシーを設定するときや、リソースを検索するときに使用できます。AWS Resource Groups タグエディタを使用してタグを管理することもできます。

IAM ロールと IAM ポリシーを設定することで、AWS リージョンの同じ AWS アカウントで他とクエリを共有できます。

クエリエディタ V2 で使用する AWS KMS 顧客管理キーの作成

対称暗号化顧客管理キーを作成するには

クエリエディタ v2 リソースを暗号化する対称暗号化顧客管理キーを作成するには、AWS KMS コンソールまたは AWS KMS API オペレーションを使用します。キーの作成方法については AWS Key Management Service デベロッパーガイドの「対称暗号化 AWS KMS キーの作成」を参照してください。

キーポリシー

キーポリシーは、カスタマーマネージドキーへのアクセスを制御します。すべてのカスタマーマネージドキーには、キーポリシーが 1 つだけ必要です。このポリシーには、そのキーを使用できるユーザーとその使用方法を決定するステートメントが含まれています。顧客管理キーを作成するときに、キーポリシーを指定できます。詳細については、「AWS Key Management Service デベロッパーガイド」の「AWS KMS キーへのアクセス管理」を参照してください。

Amazon Redshift クエリエディタ v2 で顧客管理キーを使用するには、キーポリシーで次の API オペレーションが許可されている必要があります。

- `kms:GenerateDataKey` – データを暗号化するために、一意の対称データキーを生成します。
- `kms:Decrypt` – カスタマー管理のキーで暗号化されたデータを復号化します。
- `kms:DescribeKey` — お客様が管理するキーの詳細を提供し、サービスがキーを検証できるようにします。

以下は、AWS アカウント 111122223333 のサンプル AWS KMS ポリシーです。最初のセクションでは、`kms:ViaService` がキーの使用をクエリエディタ v2 サービス (ポリシーでは `sqlworkbench.`*`region`*`.amazonaws.com` と呼びます) に制限しています。AWS アカウント がキーを使用するには 111122223333である必要があります。2 番目のセクションでは、ルートユーザーとキー管理者の AWS アカウント 111122223333 がキーにアクセスできます。

AWS アカウントを作成すると、すべての AWS のサービスとリソースに対する完全なアクセス権を持つ AWS アカウント ルートユーザーと呼ばれる 1 つのサインイン ID を使用して開始します。日常的なタスクには、ルートユーザーを使用しないことを強くお勧めします。ルートユーザー認証情報を必要とするタスクについては、「IAM ユーザーガイド」の「ルートユーザー認証情報が必要なタスク」を参照してください。

JSON

```
{
    "Version":"2012-10-17",
    "Id": "key-consolepolicy",
    "Statement": [
        {
            "Sid": "Allow access to principals authorized to use Amazon Redshift
 Query Editor V2",
            "Effect": "Allow",
```

```
            "Principal": {
                "AWS": "*"
            },
            "Action": [
                "kms:GenerateDataKey",
                "kms:Decrypt",
                "kms:DescribeKey"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": "sqlworkbench.us-east-1.amazonaws.com",
                    "kms:CallerAccount": "111122223333"
                }
            }
        },
        {
            "Sid": "Allow access for key administrators",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": [
                "kms:*"
            ],
            "Resource": "arn:aws:kms:us-east-1:111122223333:key/key_ID"
        }
    ]
}
```

次のリソースは、AWS KMS キーについての詳細を説明しています。

- AWS KMS ポリシーの詳細については、「AWS Key Management Service デベロッパーガイド」の「ポリシーでのアクセス許可の指定」を参照してください。
- トラブルシューティング AWS KMS ポリシーの情報については、「AWS Key Management Service デベロッパーガイド」の「キーアクセスのトラブルシューティング」を参照してください。
- キーの詳細については、「AWS Key Management Service デベロッパーガイド」の「AWS KMS キー」を参照してください。

クエリエディタ v2 へのアクセス

クエリエディタ v2 にアクセスするには、アクセス許可が必要です。管理者は、次のいずれかの AWS マネージドポリシーをロールにアタッチして、アクセス許可を付与できます (IAM ロールにポリシーをアタッチし、そのロールをユーザーに割り当てることをお勧めします)。これらの AWS 管理ポリシーは、リソースのタグ付けでクエリを共有する方法を制御するさまざまなオプションを使用して記述されます。IAM コンソール (https://console.aws.amazon.com/iam/) を使用して IAM ポリシーをアタッチできます。

- AmazonRedshiftQueryEditorV2FullAccess – アカウントの Amazon Redshift クエリエディタ v2 オペレーションとリソースへの完全なアクセス権を付与します。このポリシーは、その他の必要なサービスへのアクセス権限も付与します。
- AmazonRedshiftQueryEditorV2NoSharing — リソースを共有せずに Amazon Redshift クエリエディタ v2 を操作する権限を付与します。このポリシーは、その他の必要なサービスへのアクセス権限も付与します。
- AmazonRedshiftQueryEditorV2ReadSharing — リソース共有を限定して Amazon Redshift クエリエディタ v2 を操作する権限を付与します。付与されたプリンシパルは、チームと共有されているリソースの読み取りはできますが、更新はできません。このポリシーは、その他の必要なサービスへのアクセス権限も付与します。
- AmazonRedshiftQueryEditorV2ReadWriteSharing — リソースを共有して Amazon Redshift クエリエディタ v2 を操作する権限を付与します。付与されたプリンシパルは、そのチームと共有されているリソースを読み取り、更新することができます。このポリシーは、その他の必要なサービスへのアクセス権限も付与します。

また、提供された管理ポリシーで許可もしくは拒否されたアクセス権限に基づいて、独自のポリシーを作成することもできます。IAM コンソールのポリシーエディタを使用して独自のポリシーを作成する場合は、ビジュアルエディタでポリシーを作成する対象のサービスとして、[SQL Workbench] を選択します。クエリエディタ v2 では、ビジュアルエディタ および IAM Policy Simulator の中で、サービス名として AWS SQL Workbench を使用します。

プリンシパル (IAM ロールを割り当てたユーザー) が Amazon Redshift クラスターに接続するには、クエリエディタ v2 マネージドポリシーのいずれかでアクセス許可が必要です。またこのクラスターへは `redshift:GetClusterCredentialsWithIAM` または `redshift:GetClusterCredentials` アクセス許可も必要です。このアクセス許可を取得するには、管理者アクセス許可を持つユーザーが、一時的な認証情報を使用してクラスターへの接続に使用する IAM ロールにポリシーをアタッチできます。特定のクラスターにポリシーのスコープを設定することも、より一般的なポリシーを設定することもできます。一時的な認証情報を使用するアクセス

許可の詳細については、「GetClusterCredentialsWithIAM または GetClusterCredentials を呼び出す権限を持つ IAM ロールまたはユーザーの作成」を参照してください。

プリンシパル (通常は IAM ロールを割り当てたユーザー) が、アカウント内の他のユーザーに対して [アカウント設定] ページで結果セットをエクスポートすることを許可するには、`sqlworkbench:UpdateAccountExportSettings` アクセス許可をロールにアタッチする必要があります。このアクセス許可は、`AmazonRedshiftQueryEditorV2FullAccess`AWS 管理ポリシーにあります。

クエリエディタ v2 に新機能が追加されると、AWS 管理ポリシーも必要に応じて更新されます。提供されたマネージドポリシーで許可および拒否された権限をベースにして独自のポリシーを作成する場合は、マネージドポリシーの変更に対応した最新のポリシーを編集してください。Amazon Redshift の管理ポリシーの詳細については、「AWSAmazon Redshift の 管理ポリシー」を参照してください。

アクセス許可を付与するにはユーザー、グループ、またはロールにアクセス許可を追加します。

- AWS IAM アイデンティティセンター のユーザーとグループ:

  アクセス許可セットを作成します。「AWS IAM アイデンティティセンター ユーザーガイド」の「アクセス許可セットを作成する」の手順に従ってください。
- IAM 内で、ID プロバイダーによって管理されているユーザー:

  ID フェデレーションのロールを作成します。詳細については IAM ユーザーガイド の サードパーティー ID プロバイダー (フェデレーション) 用のロールを作成する を参照してください。
- IAM ユーザー:
  - ユーザーが担当できるロールを作成します。手順については IAM ユーザーガイド の IAM ユーザーのロールの作成 を参照してください。
  - (お奨めできない方法) ポリシーをユーザーに直接アタッチするか、ユーザーをユーザーグループに追加します。「IAM ユーザーガイド」の「ユーザー (コンソール) へのアクセス許可の追加」の指示に従います。

> **Note**
>
> AWS IAM アイデンティティセンター 管理者がアカウント全体にわたり特定のアクセス許可セットのアクセス許可セットの関連付けをすべて削除すると、削除されたアクセス許可セットに元々関連付けられていたクエリエディタのリソースにアクセスできなくなります。後で同じアクセス許可を再作成すると、新しい内部識別子が作成されます。内部識別子が変更

されたため、ユーザーが以前所有していたクエリエディタのリソースにはアクセスできません。管理者がアクセス許可セットを削除する前に、そのアクセス許可セットのユーザーは、ノートブックやクエリなどのクエリエディタのリソースをバックアップとしてエクスポートしておくことをお勧めします。

クエリエディタ v2 からクラスターまたはワークグループに接続するためのプリンシパルタグの設定

フェデレーションユーザーオプションを使用してクラスターまたはワークグループに接続するには、プリンシパルタグを使って IAM ロールまたはユーザーを設定します。または、`RedshiftDbUser`および (オプション) `RedshiftDbGroups` で、ID プロバイダー (IdP) によって渡されるようにセットアップします。IAM を使用したタグの管理の詳細については、「IAM ユーザーガイド」の「AWS Security Token Service でのセッションタグの受け渡し」を参照してください。AWS Identity and Access Management を使用してアクセスをセットアップするために、管理者は IAM コンソールを使用してタグを追加できます (https://console.aws.amazon.com/iam/)。

IAM ロールにプリンシパルタグを追加するには

1. AWS マネジメントコンソール にサインインして、IAM コンソール https://console.aws.amazon.com/iam/ を開きます。
2. ナビゲーションペインで [ロール] を選択します。
3. フェデレーションユーザーを使用して、クエリエディタ v2 へのアクセスが必要なロールを選択します。
4. [タグ] タブを選択します。
5. [タグの管理] をクリックします。
6. [タグを追加] を選択して、`RedshiftDbUser` を [キー] に入力し、フェデレーションユーザー名を [値] に入力します。
7. オプションで、[タグの追加] を選択して、`RedshiftDbGroups` を [キー] に入力し、ユーザーに関連付けるグループ名を [値] に入力します。
8. [変更を保存] を選択して、選択した IAM ロールに関連付けられているタグのリストを表示します。変更が反映されるまで、数秒かかる場合があります。
9. フェデレーションユーザーを使用するには、変更が反映された後、クエリエディタ v2 のページを更新します。

プリンシパルタグを渡すように ID プロバイダー (IdP) をセットアップする

ID プロバイダー (IdP) を使用してタグを設定する手順は、IdP によって異なります。ユーザーおよびグループ情報を SAML 属性に渡す手順については、IdP のドキュメントを参照してください。正しく設定すると、AWS Security Token Service によって使用され、RedshiftDbUser および RedshiftDbGroups のプリンシパルタグに表示される SAML レスポンスに次の属性が表示されます。

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:RedshiftDbUser">
    <AttributeValue>db-user-name</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:RedshiftDbGroups">
    <AttributeValue>db-groups</AttributeValue>
</Attribute>
```

オプションの *db_groups* は、group1:group2:group3のようにコロンで区切る必要があります。

さらに、TransitiveTagKeys属性を使用して、ロールの連鎖中にタグを保持できます。

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys">
  <AttributeValue>RedshiftDbUser</AttributeValue>
  <AttributeValue>RedshiftDbGroups</AttributeValue>
</Attribute>
```

クエリエディタ v2 のセットアップの詳細については、「クエリエディタv2 を使用するために必要なアクセス許可」をご参照ください。

> **Note**
>
> クエリエディタ v2 の[フェデレーテッドユーザー] 接続オプションを使用してクラスターまたはワークグループに接続すると、ID プロバイダー (IdP) は RedshiftDbUser と RedshiftDbGroups のカスタムプリンシパルタグを提供できます。現時点では、AWS IAM アイデンティティセンター はクエリエディタ v2 へのカスタムプリンシパルタグを直接渡すことはサポートしていません。

クエリエディタ v2 を開く

Amazon Redshift では、Amazon Redshift コンソールのクエリエディタ v2 を使用して、データウェアハウスクラスターに対して SQL クエリを実行できます。クエリエディタ v2 は、アドホッククエ

リ、データ検索、データ分析タスクを実行するためのユーザーフレンドリーなインターフェイスを提供するウェブベースのツールです。以下のセクションでは、コンソールでクエリエディタ v2 を開き、クエリエディタの機能を効果的に活用するプロセスについて説明します。

クエリエディタ v2 を開くには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーターメニューで、[エディタ]、次に [クエリエディタ V2] を選択します。新しいブラウザでクエリエディタ v2 が開きます。

クエリエディタページには、次のようにビューを選択するナビゲータメニューがあります。

エディタ

テーブルとして編成され、データベースに含まれるデータを管理およびクエリします。データベースには、保存されたデータを格納することも、Amazon S3 などの他の場所に格納されたデータへの参照を含めることもできます。クラスターまたはサーバーレスワークグループに含まれるデータベースに接続します。

[エディタ] ビューで操作する場合、次のコントロールがあります。

- [Cluster] (クラスター) または [Workgroup] (ワークグループ) フィールドには、現在接続されている名前が表示されます。[Database] (データベース) フィールドには、クラスターまたはワークグループ内のデータベースが表示されます。[データベース] ビューで実行するアクションは、デフォルトでは選択したデータベースに対して動作します。
- クラスター、データベース、スキーマのツリービュー階層ビュー。スキーマでは、テーブル、ビュー、関数、および保存する手順を操作できます。ツリービューの各オブジェクトにはコンテキストメニューが用意されており、そのオブジェクトに関連するアクション ([更新] または [ドロップ]など) を実行できます。
- ⊕

  [作成] アクションを使用して、データベース、スキーマ、テーブル、および関数を作成します。

-  [データのロード] アクションは、Amazon S3 またはローカルファイルからデータをデータベースにロードします。
-  保存 アイコンをクリックすると、クエリを保存できます。
-  ショートカット アイコンをクリックすると、エディタのキーボードショートカットが表示されます。
- 詳細アイコンをクリックすると、次のような追加のアクションが表示されます 。
  - [チームと共有する]: チームとクエリやノートブックを共有します。詳細については、「チームとしての共同作業と共有」を参照してください。
  - [ショートカット]: エディタのキーボードショートカットを表示します。
  - [タブ履歴]: エディタのタブのタブ履歴を表示します。
  - [オートコンプリートを更新]: SQL の作成時に表示する候補を更新します。
- クエリを入力および実行できる

  [Editor] (エディタ) 領域。

  クエリを実行した後、[結果] タブが表示され、結果が表示されます。ここで [チャート] をオンにすると、結果を視覚化できます。また、結果を[Export] (エクスポート) できます。
- セクションを追加して SQL を入力および実行するか、[Markdown] (マークダウン) を追加できる

  [Notebook] (ノートブック) 領域。

  クエリを実行した後、[結果] タブが表示され、結果が表示されます。ここで結果を[Export] (エクスポート) できます。

クエリ



クエリには、データベース内のデータを管理およびクエリする SQL コマンドが含まれています。クエリエディタ v2 を使用してサンプルデータをロードすると、サンプルクエリが作成され、保存されます。

保存したクエリを選択すると、コンテキスト (右クリック) メニューを使用してクエリを開き、名前を変更、削除することができます。[クエリの詳細] を選択すると、保存したクエリの [クエリ ARN] などの属性を表示できます。また、バージョン履歴を表示したり、クエリに添付されたタグを編集してチームと共有したりすることもできます。

ノートブック



SQL ノートブックには、SQL セルと Markdown セルが含まれています。ノートブックを使用して、1 つのドキュメント内で複数の SQL コマンドを編成、注釈付け、共有できます。

保存したノートブックを選択すると、コンテキスト (右クリック) メニューを使用してノートブックを開き、名前を変更、複製、削除することができます。[ノートブックの詳細] を選択すると、保存したノートブックの [ノートブック ARN] などの属性を表示できます。また、バージョン履歴を表示、ノートブックに添付されたタグを編集、エクスポート、チームと共有することもできます。詳細については、「Amazon Redshift のノートブック」を参照してください。

チャート



チャートは、データの視覚的な表示です。クエリエディタ v2 には、さまざまなタイプのチャートを作成して保存するためのツールが用意されています。

保存したチャートを選択すると、コンテキスト (右クリック) メニューを使用してチャートを開き、名前を変更、削除することができます。[チャートの詳細] を選択すると、保存したチャートの [チャート ARN] などの属性を表示できます。また、チャートに添付されたタグを編集してエクスポートすることもできます。詳細については、「クエリ結果の視覚化」を参照してください。

履歴



クエリ履歴は、Amazon Redshift クエリエディタ v2 を使用して実行されたクエリの一覧です。これらのクエリは、個別のクエリまたは SQL ノートブックの一部として実行されます。詳細については、「クエリとタブの履歴を表示する」を参照してください。

スケジュールされたクエリ



スケジュールされたクエリは、特定の時間に開始するように設定されたクエリです。

すべてのクエリエディタ v2 ビューには次のアイコンがあります。

- 

  [Visual mode] (ビジュアルモード) アイコンをクリックすると、ライトモードとダークモードが切り替わります。

- 

  [Settings] (設定) アイコンをクリックすると、さまざまな設定画面のメニューが表示されます。

  - 

    [Editor preferences] (エディタの環境設定) アイコンをクリックすると、クエリエディタ v2 を使用するときに、設定を編集できます。ここでワークスペース設定を編集して、フォントサイズ、タブサイズ、その他の表示設定を変更できます。[オートコンプリート] をオン(またはオフ) にして、SQL の入力時に候補を表示することもできます。

  - 

    [Connections] (接続) アイコンをクリックすると、エディタタブで使用されている接続が表示されます。

    接続は、データベースからデータを取得するために使用されます。接続は、特定のデータベースに対して作成されます。分離された接続では、あるエディタタブで一時テーブルを作成するなど、データベースを変更する SQL コマンドの結果は、別のエディタタブに表示されません。クエリエディタ v2 でエディタタブを開くとき、デフォルトは分離された接続です。共有接続を作成するとき、つまり、分離されたセッションスイッチをオフにすると、同じデータベースへの他

の共有接続の結果が相互に表示されます。ただし、データベースへの共有接続を使用するエディタタブは並列で実行されません。同じ接続を使用するクエリは、接続が使用可能になるまで待機する必要があります。あるデータベースへの接続は別のデータベースと共有できないため、SQL の結果は異なるデータベース接続では表示されません。

アカウント内のユーザーがアクティブにできる接続の数は、クエリエディタ v2 管理者によって制御されます。

- 

  [Account settings] (アカウント設定) アイコンは、アカウント内のすべてのユーザーの特定の設定を変更するために管理者が使用します。詳細については、「アカウント設定」を参照してください。

クエリエディタ v2 を使用する際の考慮事項

クエリエディタ v2 を使用する場合は、次の点を考慮してください。

- クエリの最大期間は 24 時間です。
- クエリ結果の最大サイズは 100 MB です。呼び出しで返されるレスポンスデータが 100 MB を上回る場合は、最初の 100 MB が警告とともに返されます。
- 最大 300,000 文字のクエリを実行できます。
- 最大 30,000 文字のクエリを保存できます。
- デフォルトでは、クエリエディタ v2 は、実行する各 SQL コマンドを自動コミットします。BEGIN ステートメントを指定すると、BEGIN-COMMIT ブロックまたは BEGIN-ROLLBACK ブロック内のステートメントは 1 つのトランザクションとして実行されます。トランザクションの詳細については、Amazon Redshift データベースデベロッパーガイドの「BEGIN」を参照してください。
- SQL ステートメントの実行中にクエリエディタ v2 が表示する警告の最大数は 10 です。例えば、ストアドプロシージャを実行した場合、表示される RAISE ステートメントは 10 個までです。
- クエリエディタ v2 は、カンマ (,) を含む IAM `RoleSessionName` をサポートしていません。次のようなエラーが表示されることがあります: エラーメッセージ: 「'AROA123456789EXAMPLE:mytext,yourtext'は TagValue の有効な値ではありません - 不正な文字が含まれています」 この問題は、カンマを含む IAM `RoleSessionName` を定義し、その IAM ロールでクエリエディタ v2 を使用したときに発生します。

  IAM `RoleSessionName` の詳細については、「IAM ユーザーガイド」の「RoleSessionName SAML 属性」を参照してください。

アカウント設定

適切な IAM 権限を持つユーザーは、同じ AWS アカウント の他のユーザーに対して[Account settings] (アカウント設定) を表示および変更できます。この管理者は以下を表示または設定できます。

- アカウント内のユーザーあたりの最大同時データベース接続。これには分離されたセッションも含まれます。この値を変更すると、変更が反映されるまで 10 分かかることがあります。
- アカウントのユーザーが SQL コマンドからファイルに結果セット全体をエクスポートできるようにします。
- 関連する保存済みクエリを含むサンプルデータベースをロードして表示します。
- アカウントユーザーがローカルファイルからデータをロードするために使用する Amazon S3 パスを指定します。
- クエリエディタ v2 のリソースを暗号化するために使用される KMS キー ARN を表示します。

Amazon Redshift データベースに接続する

データベースに接続するには、ツリービューパネルでクラスターまたはワークグループ名を選択します。プロンプトが表示されたら、接続パラメータを入力します。

クラスターまたはワークグループとそのデータベースに接続するときは、通常 [Database] (データベース) 名を指定します。また、次の認証方法のいずれかに必要なパラメータも指定します。

IAM アイデンティティセンター

この方法では、ID プロバイダー (IdP) からのシングルサインオン認証情報を使用して Amazon Redshift データウェアハウスに接続します。Amazon Redshift コンソールの IAM アイデンティティセンターで、クラスターまたはワークグループが有効になっている必要があります。IAM アイデンティティセンターへの接続の設定については、「Redshift を AWS IAM アイデンティティセンターに接続してシングルサインオンエクスペリエンスを提供する」を参照してください。

フェデレーティッドユーザー

この方法では、IAM ロールまたはユーザーのプリンシパルタグで接続の詳細を指定する必要があります。これらのタグは、AWS Identity and Access Managementまたは ID プロバイダー (IdP) で設定します。クエリエディタ v2 では、次のタグを使用します。

- `RedshiftDbUser` — このタグは、クエリエディタ v2 で使用されるデータベースユーザーを定義します。このタグは必須です。

- RedshiftDbGroups — このタグは、クエリエディタ v2 に接続する場合に結合されるデータベースグループを定義します。このタグはオプションであり、その値は group1:group2:group3 のようにコロンで区切る必要があります。空の値は無視されます。つまり、group1::::group2は group1:group2 と解釈されます。

これらのタグは、クラスターの認証情報を取得するために redshift:GetClusterCredentials API に転送されます。詳細については、「クエリエディタ v2 からクラスターまたはワークグループに接続するためのプリンシパルタグの設定」を参照してください。

データベースのユーザー名を使用する一時的な認証情報

このオプションは、クラスターに接続している場合のみ使用できます。この方法では、クエリエディタ v2 によって、データベースのユーザー名が提供されます。クエリエディタ v2 が、このデータベースユーザー名でデータベースに接続するための一時パスワードを生成します。この方法を使用して接続するユーザーには、redshift:GetClusterCredentials への IAM アクセス許可を付与する必要があります。ユーザーがこの方法を使用できないようにするには、IAM ユーザーまたはロールを変更して、このアクセス許可を拒否します。

IAM ID を使用した一時的な認証情報

このオプションは、クラスターに接続している場合のみ使用できます。この方法では、クエリエディタ v2 は、ユーザー名を IAM アイデンティティにマップして、IAM アイデンティティとしてデータベースに接続するための一時パスワードを生成します。この方法を使用して接続するユーザーには、redshift:GetClusterCredentialsWithIAM への IAM アクセス許可を付与する必要があります。ユーザーがこの方法を使用できないようにするには、IAM ユーザーまたはロールを変更して、このアクセス許可を拒否します。

データベースユーザー名とパスワード

この方法では、接続先のデータベースのユーザー名とパスワードも指定します。クエリエディタ v2 は、ユーザーに代わり、AWS Secrets Managerに保存するシークレットを作成します。このシークレットには、データベースに接続するための認証情報が含まれます。

AWS Secrets Manager

この方法では、データベース名の代わりに、データベースとサインイン認証情報を含むシークレットマネージャーに保存されているシークレットを指定します。シークレットの作成方法の詳細については、「データベース接続認証情報のシークレットの作成」を参照してください。

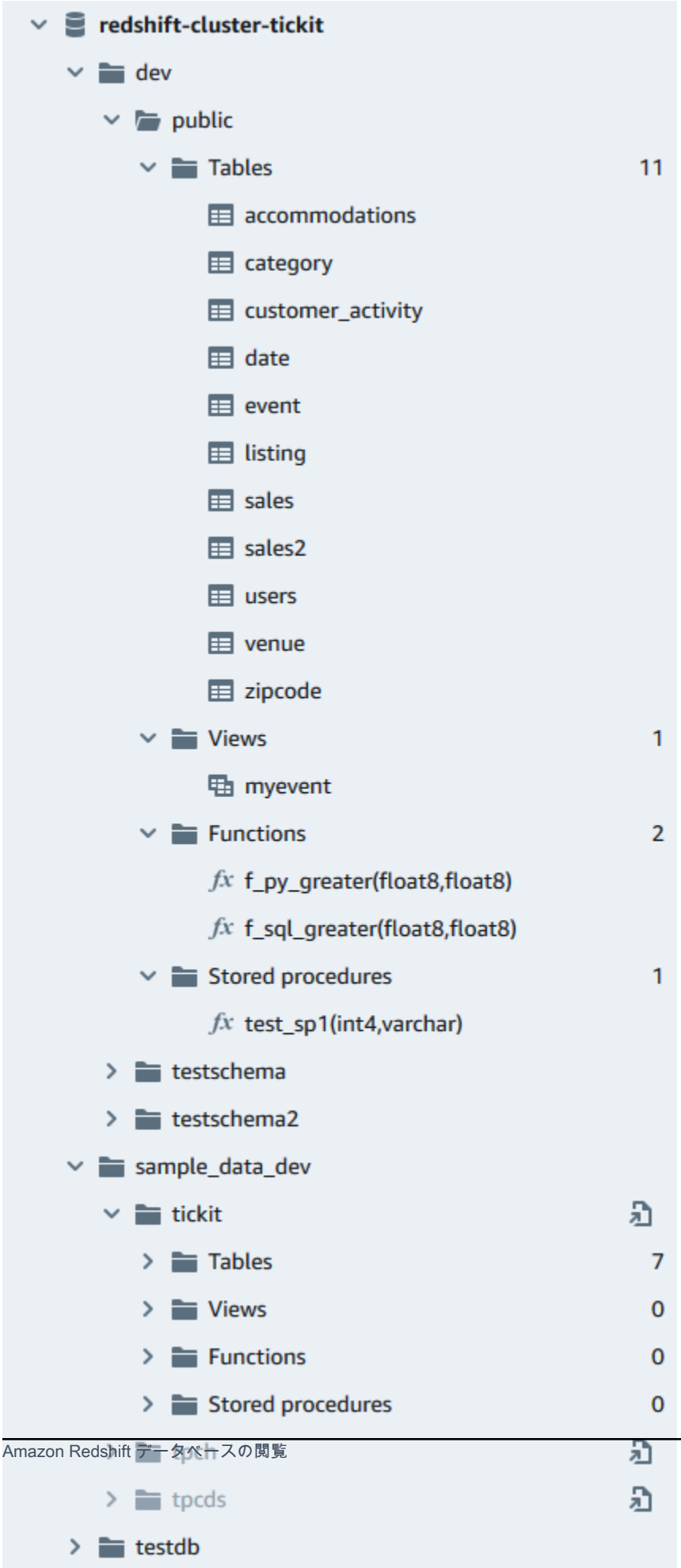
クエリエディタ v2 でクラスターまたはワークグループを選択すると、コンテキストに応じて、コンテキスト (右クリック) メニューを使用して接続を作成、編集、削除できます。[接続の詳細] を選択

すると、接続の [接続 ARN] などの属性を表示できます。接続に添付されたタグを編集することもできます。

Amazon Redshift データベースの閲覧

データベース内では、ツリービューパネルでスキーマ、テーブル、ビュー、関数、および保存する手順を管理できます。ビュー内の各オブジェクトには、コンテキスト メニュー (右クリック) でアクションが関連付けられます。

階層ツリービューパネルには、データベースオブジェクトが表示されます。ツリービューパネルを更新して、ツリービューが最後に表示された後で作成された可能性があるデータベースオブジェクトを表示するには、
アイコンを選択します。オブジェクトを右クリックしてコンテキストメニューを開き、実行できるアクションを確認します。


redshift-cluster-tickit
dev
public
Tables 11
accommodations
category
customer_activity
date
event
listing
sales
sales2
users
venue
zipcode
Views 1
myevent
Functions 2
f_py_greater(float8,float8)
f_sql_greater(float8,float8)
Stored procedures 1
test_sp1(int4,varchar)
testschema
testschema2
sample_data_dev
tickit
Tables 7
Views 0
Functions 0
Stored procedures 0
tpch
tpcds
testdb

テーブルを選択すると、次の操作を実行できます。

- テーブル内のすべての列をクエリする SELECT ステートメントを使用してエディタでクエリを開始するには、[テーブルを選択] を使用します。
- 属性またはテーブルを表示するには、[Show table definition] (テーブル定義を表示) を使用します。これを使用して、列名、列型、エンコーディング、分散キー、ソートキー、および列に Null 値を含めることができるかどうかを表示します。テーブル属性の詳細については、「Amazon Redshift データベースデベロッパーガイド」の「CREATE TABLE」を参照してください。
- テーブルを削除するには、[削除] を使用します。テーブルからすべての行を削除するには [Truncate table] (テーブルの切り捨て) 、データベースからテーブルを削除するには [テーブルの削除] を使用します。詳細については、「Amazon Redshift データベースデベロッパーガイド」の「TRUNCATEDROP TABLE」を参照してください。

[スキーマ] を選択し、スキーマを [更新] または [削除] します。

[ビュー定義を表示] または [ドロップ ビュー] のビューを選択します。

関数を選択し、[Show function definition] (関数定義を表示) または [Drop function] (関数を削除) します。

保存する手順を選択して [Show procedure definition] (手順定義を表示) または [Drop procedure] (手順を削除) します。

データベースオブジェクトの作成

データベース、スキーマ、テーブル、およびユーザー定義関数 (UDF) など、データベースオブジェクトを作成できます。データベースオブジェクトを作成するには、クラスターまたはワークグループとデータベースに接続している必要があります。

データベースの作成

クエリエディタ v2 を使用して、クラスターまたはワークグループにデータベースを作成できます。

データベースを作成する

この他のコマンドオプションについては、「Amazon Redshift データベースデベロッパーガイド」の「CREATE DATABASE」を参照してください。

1.  [作成] を選択し、次に [データベース] を選択します。
2. [データベース名] を入力します。
3. (任意) [ユーザーとグループ] を選択し、[データベースユーザー] を選択します。
4. (オプション) データベースはデータ共有または AWS Glue Data Catalog から作成できます。AWS Glue の詳細については、AWS Glue デベロッパーガイド の「What is AWS Glue?」を参照してください。
   - (オプション) [データ共有を使用して作成する] を選択し、[データ共有の選択] を選択します。このリストには、現在のクラスターまたはワークグループでコンシューマーデータ共有の作成に使用できるプロデューサーデータ共有が含まれています。
   - (オプション) [AWS Glue Data Catalogを使用して作成する] を選択し、[AWS Glue データベースを選択] を選択します。[データカタログスキーマ] に、データを参照するときにスキーマで使用する名前を 3 つの部分からなる表記 (database.schema.table) で入力します。
5. [データベースの作成] を選択します。

   ツリービューパネルに新しいデータベースが表示されます。

   データ共有から作成したデータベースをクエリするオプションのステップを選択する場合は、クラスターまたはワークグループ内の Amazon Redshift データベース (例えば、デフォルトデータベース dev) に接続し、[データ共有を使用して作成する] の選択時に作成したデータベース名を参照する 3 つの部分からなる表記 (database.schema.table) を使用します。データ共有データベースは、クエリエディタ v2 のエディタタブに表示されますが、直接接続には使用できません。

   AWS Glue Data Catalog から作成したデータベースをクエリするオプションのステップを選択する場合は、クラスターまたはワークグループ内の Amazon Redshift データベース (例えば、デフォルトデータベース dev) に接続し、[AWS Glue Data Catalog を使用して作成する] の選択時に作成したデータベース名、[データカタログスキーマ] で名前を付けたスキーマ、および AWS Glue Data Catalog のテーブルを参照する 3 つの部分からなる表記 (database.schema.table) を使用します。次のような表記になります。

```
SELECT * FROM glue-database.glue-schema.glue-table
```

> **Note**
>
> 接続方法として [IAM ID を使用した一時的な認証情報] を使用してデフォルトのデータベースに接続していること、および AWS Glue データベースの使用権限が IAM 認証情報に付与されていることを確認します。
>
> ```
> GRANT USAGE ON DATABASE glue-database to "IAM:MyIAMUser"
> ```

AWS Glue データベースは、クエリエディタ v2 のエディタタブに表示されますが、直接接続には使用できません。

AWS Glue Data Catalog にクエリを実行する方法の詳細については、「Amazon Redshift データベース開発者ガイド」の「コンシューマーとして Lake Formation 管理のデータ共有を使用する」と「プロデューサーとして Lake Formation 管理のデータ共有を使用する」を参照してください。。

データ共有コンシューマーとしてデータベースを作成する例

次の例は、クエリエディタ v2 を使用してデータ共有からデータベースを作成するために使用した特定のシナリオを示しています。このシナリオを通じて、使用環境でデータ共有からデータベースをどのようにして作成できるかを確認してください。このシナリオでは、cluster-base(プロデューサークラスター) と cluster-view (コンシューマークラスター) の 2 つのクラスターを使用します。

1. Amazon Redshift コンソールを使用してクラスター cluster-base でテーブル category2 のデータ共有を作成します。プロデューサーデータ共有の名前は datashare_base です。

   データ共有の作成の詳細については、「Amazon Redshift データベース開発者ガイド」の「Amazon Redshift でのクラスター間のデータの共有」を参照してください。
2. Amazon Redshift コンソールを使用して、クラスター cluster-view でテーブル category2 のコンシューマーとしてデータ共有 datashare_base を受け入れます。
3. クエリエディタ v2 のツリービューパネルを開いて、次のような cluster-base の階層を表示します。

   - クラスター: cluster-base

- データベース: dev
  - スキーマ : public
    - テーブル: category2

4.  [作成] を選択し、次に [データベース] を選択します。
5. [データベース名] に「see_datashare_base」と入力します。
6. [データ共有を使用して作成する] を選択し、[データ共有を選択] を選択します。作成するデータベースのソースとして使用する datashare_base を選択します。

   クエリエディタ v2 のツリービューパネルに、次のように cluster-view の階層が表示されます。

   - クラスター: cluster-view
     - データベース: see_datashare_base
       - スキーマ : public
         - テーブル: category2
7. データをクエリするときは、クラスター cluster-view のデフォルトデータベース (通常は dev という名前) に接続しますが、SQL でデータ共有データベース see_datashare_base を参照します。

   > **Note**
   >
   > クエリエディタ v2 のエディタビューでは、選択したクラスターは cluster-view です。選択したデータベースは dev です。データベース see_datashare_base は表示されますが、直接接続には使用できません。dev データベースを選択し、実行した SQL で see_datashare_base を参照します。

   ```
   SELECT * FROM "see_datashare_base"."public"."category2";
   ```

   クエリは、クラスター cluster_base のデータ共有 datashare_base からデータを取得します。

AWS Glue Data Catalog からデータベースを作成する例

次の例は、クエリエディタ v2 を使用して AWS Glue Data Catalog からデータベースを作成するために使用した特定のシナリオを示しています。このシナリオを通じて、使用環境で AWS Glue Data Catalog からデータベースをどのようにして作成できるかを確認してください。このシナリオでは 1 つのクラスター `cluster-view` を使用して、作成したデータベースを格納します。

1.  [作成] を選択し、次に [データベース] を選択します。
2. [データベース名] に「`data_catalog_database`」と入力します。
3. [AWS Glue Data Catalogを使用して作成する] を選択し、[AWS Glue データベースを選択] を選択します。作成するデータベースのソースとして使用する `glue_db` を選択します。

   [データカタログスキーマ] を選択し、3 つの部分からなる表記で使用するスキーマ名として「`myschema`」と入力します。

   クエリエディタ v2 のツリービューパネルに、次のように `cluster-view` の階層が表示されます。

   - クラスター: `cluster-view`
     - データベース: `data_catalog_database`
       - スキーマ : `myschema`
         - テーブ: `category3`
4. データをクエリするときは、クラスター `cluster-view` のデフォルトデータベース (通常は `dev` という名前) に接続しますが、SQL でデータベース `data_catalog_database` を参照します。

   > **Note**
   >
   > クエリエディタ v2 のエディタビューでは、選択したクラスターは `cluster-view` です。選択したデータベースは `dev` です。データベース `data_catalog_database` は表示されますが、直接接続には使用できません。`dev` データベースを選択し、実行した SQL で `data_catalog_database` を参照します。

```
SELECT * FROM "data_catalog_database"."myschema"."category3";
```

クエリは、AWS Glue Data Catalog でカタログ化されたデータを取得します。

スキーマの作成

クエリエディタ v2 を使用して、クラスターまたはワークグループにスキーマを作成できます。

スキーマを作成するには

スキーマについては、「Amazon Redshift データベースデベロッパーガイド」の「スキーマ」を参照してください。

1. 
   [作成]を選択し、[スキーマ] を選択します。
2. [スキーマ名] を入力します。
3. [Local] (ローカル) または[External] (外部) を [Schema type] (スキーマタイプ) として選択します。

   ローカルスキーマの詳細については、Amazon Redshift データベースデベロッパーガイドの「スキーマの作成」を参照してください。外部スキーマの詳細については、Amazon Redshift データベースデベロッパーガイドの「外部スキーマの作成」を参照してください。
4. [External] (外部) を選択すると、外部スキーマの次の選択肢があります。

   - Glue データカタログ — AWS Glue 内のテーブルを参照する外部スキーマを Amazon Redshift に作成します。AWS Glue データベースを選択する以外に、クラスターに関連付けられた IAM ロールとデータカタログに関連付けられた IAM ロールを選択します。
   - PostgreSQL — Amazon Redshift で、Amazon RDS for PostgreSQL または Amazon Aurora PostgreSQL-Compatible Edition データベースを参照する外部スキーマを作成します。データベースの接続情報も提供します。フェデレーティッドクエリの詳細については、「Amazon Redshift データベースデベロッパーガイド」の「フェデレーテッドクエリを使用したデータのクエリ」を参照してください。
   - MySQL — Amazon Redshift で、Amazon RDS for MySQL または Amazon Aurora MySQL-Compatible Edition データベースを参照する外部スキーマを作成します。データベースの接続情報も提供します。フェデレーティッドクエリの詳細については、「Amazon Redshift データベースデベロッパーガイド」の「フェデレーテッドクエリを使用したデータのクエリ」を参照してください。
5. [スキーマの作成] を選択します。

ツリービューパネルに新しいスキーマが表示されます。

テーブルの作成

クエリエディタ v2 を使用して、クラスターまたはワークグループにテーブルを作成できます。

テーブルを作成するには

テーブルの各列を指定または定義するカンマ区切り値 (CSV) ファイルに基づいてテーブルを作成できます。テーブルの詳細については、「Amazon Redshift データベースデベロッパーガイド」の「テーブル設計」と「CREATE TABLE」を参照してください。

[エディタでクエリを開く] をクリックして [CREATE TABLE] のステートメントを表示して編集してから、クエリを実行してテーブルを作成します。

1. 
   [Create
   (作成)、[Table] (テーブル) の順に選択します。
2. スキーマを選択します。
3. テーブル名を入力します。
4. 
   [フィールドの追加] を選択して列を追加します。
5. CSV ファイルをテーブル定義のテンプレートとして使用します。
   a. [CSV からロードする] を選択します。
   b. ファイルの場所をブラウズします。

      CSV ファイルを使用する場合は、ファイルの最初の行に列見出しが含まれていることを確認してください。
   c. ファイルを選択し、[開く] を選択します。列の名前とデータ型が意図したものであることを確認します。
6. 各列について、列を選択し、必要なオプションを選択します。
   - [エンコード] の値を選択します。
   - [デフォルト値] を選択します。

- 列の値を増分させたい場合は、[Automatically increment] (自動増分) をオンにします。次に、[Auto increment seed] (自動増分シード) と [Auto increment step] (自動増分ステップ) の値を指定します。
- 列が常に値を含む必要がある場合は、[Not NULL] をオンにします。
- 列の [サイズ] 値を入力します。
- 列をプライマリキーにしたい場合は、[プライマリキー] をオンにします。
- 列を一意のキーにしたい場合は、[Unique key] (一意のキー) をオンにします。

7. (任意) [テーブルの詳細] を選択し、以下のいずれかのオプションを選択します。

   - 分散キーの列とスタイル。
   - ソートキー列とソートタイプ。
   - [Backup] (バックアップ) をオンにして、クラススナップショットにテーブルを含めます。
   - [一時テーブル] をオンにして、一時テーブルを作成します。

8. [エディタでクエリを開く] をクリックしてテーブルを定義するオプションの指定を続行するか テーブルの作成 を選択してテーブルを作成します。

関数の作成

クエリエディタ v2 を使用して、クラスターまたはワークグループに関数を作成できます。

関数を作成するには

1.  [作成] を選択し [関数] を選択します。
2. [タイプ] については [SQL] または [Python] を選択します。
3. [スキーマ] の値を選択します。
4. 関数の [名前] に値を入力します。
5. 関数の [Volatility] (ボラティリティ) に値を入力します。
6. 入力パラメータの順に、データ型別に [パラメータ] を選択します。
7. [戻り値] で、データ型を選択します。
8. 関数の [SQL プログラム] または [Python プログラム] コードを入力します。
9. [作成] を選択します。

ユーザー定義関数 (UDF) の詳細については、「Amazon Redshift データベースデベロッパーガイド」の「ユーザー定義関数の作成」を参照してください。

クエリとタブの履歴を表示する

クエリ履歴は、クエリエディタ v2 で表示できます。クエリエディタ v2 を使用して実行したクエリのみがクエリ履歴に表示されます。[Editor] (エディタ) タブまたは [Notebook] (ノートブック) タブを使用して実行された両方のクエリが表示されます。表示されるリストを、`This week` のように月曜日から日曜日と 1 週間と定義するなど、期間でフィルタリングできます。クエリのリストは、フィルターに一致するクエリを一度に 25 行取得します。[Load more] (さらに読み込む) を選択すると、次のセットが表示されます。[Actions] (アクション) メニューからクエリを選択します。実行できるアクションは、選択したクエリが保存されているかどうかによって変わります。以下の操作を選択できます。

- [View query details] (クエリの詳細を表示) — 実行済みクエリに関する情報の詳細ページを表示します。
- [Open query in a new tab] (新しいタブでクエリを開く)— 新しいエディタタブを開き、選択したクエリをプライミングします。接続中の場合、クラスターまたはワークグループと、データベースが自動的に選択されます。クエリを実行するには、まず、正しいクラスターまたはワークグループと、データベースが選択されていることを確認します。
- [Open source] (ソースを開く)–タブ — そのまま開いている場合は、クエリの実行時に含まれていたエディタまたはノートブックのタブに移動します。クエリの実行後にエディタまたはノートブックの内容が変更された可能性があります。
- [Open saved query] (保存したクエリを開く) — [Editor] (エディタ) タブまたは [Notebook] (ノートブック) タブに移動してクエリを開きます。

また、[Editor] (エディタ) タブまたは [Notebook] (ノートブック) タブで実行されたクエリの履歴を表示できます。クエリの履歴をタブで表示するには、[Tab history] (タブ履歴) を選択します。タブ履歴では、次の操作を実行できます。

- [Copy query] (クエリのコピー)— クエリバージョンの SQL コンテンツをクリップボードにコピーします。
- [Open query in a new tab] (新しいタブでクエリを開く)— 新しいエディタタブを開き、選択したクエリをプライミングします。クエリを実行するには、クラスター、ワークグループ、データベースを選択する必要があります。

- [View query details] (クエリの詳細を表示) — 実行済みクエリに関する情報の詳細ページを表示します。

Amazon Q 生成 SQL の操作

> **Note**
>
> 現時点で Amazon Q 生成 SQL をサポートしているのは、以下の AWS リージョンのみです。
>
> - 米国東部 (バージニア北部) リージョン (us-east-1)
> - 米国東部 (オハイオ) リージョン (us-east-2)
> - 米国西部 (オレゴン) リージョン (us-west-2)
> - アジアパシフィック (ムンバイ) リージョン (ap-south-1)
> - アジアパシフィック (ソウル) リージョン (ap-northeast-2)
> - アジアパシフィック (シンガポール) リージョン (ap-southeast-1)
> - アジアパシフィック (シドニー) リージョン (ap-southeast-2)
> - アジアパシフィック (東京) リージョン (ap-northeast-1)
> - カナダ (中部) リージョン (ca-central-1)
> - 欧州 (フランクフルト) リージョン (eu-central-1)
> - 欧州 (アイルランド) リージョン (eu-west-1)
> - 欧州 (ロンドン) リージョン (eu-west-2)
> - 欧州 (パリ) リージョン (eu-west-3)
> - 南米 (サンパウロ) リージョン (sa-east-1)
>
> データの処理場所の詳細については、Amazon Q Developer ユーザーガイドの「Cross region inference in Amazon Q Developer」を参照してください。

Amazon Redshift クエリエディタ v2 では、Amazon Q の生成 SQL 機能を利用できます。これは、プロンプトとデータベーススキーマに基づいて SQL ステートメントを生成するコード生成アシスタントです。このコード生成アシスタントは、クエリエディタ v2 でノートブックを作成している際に使用できます。生成された SQL は、ノートブックが接続されているデータベース向けのものです。

Amazon Q 生成 SQL を利用する際の質問は、具体的にします。複雑なリクエストの場合は反復し、返される内容が正確であることを確認します。

分析リクエストを自然言語で提供する場合は、コード生成アシスタントが必要な点を正確に把握できるように、可能な限り具体的に質問します。「最も多くチケットを販売した会場の上位を検索する」と質問する代わりに、「2008 年に最もチケットの売上があったトップ 3 の会場名または ID を検索する」などの詳細情報を提供します。データベース内のオブジェクトの名前がわかっている場合は、一貫した具体的な名前を使用します。同じオブジェクトを参照するのにさまざまな方法を使用すると、アシスタントが混乱する可能性があるため、データベースで定義されているスキーマ名、テーブル名、列名などを使用します。

複雑なリクエストの場合は、アシスタントが解釈しやすい複数のシンプルなステートメントに分割します。フォローアップの質問を繰り返し行うことで、アシスタントからより詳細な分析が得られます。例えば、まず「会場が最も多くある州はどこですか」と質問します。次に、その回答に基づいて、「この州で最も人気のある会場はどれですか」と質問します。

生成された SQL は実行する前に調べて、正確であることを確認します。生成された SQL クエリにエラーがある場合、または意図した内容と違う場合は、リクエスト全体を言い換えるのではなく、アシスタントに修正方法を指示します。例えば、クエリに年に関する述語句が欠落している場合は、「2008 年からの会場を提供する」と伝えます。

生成された SQL の実行により受信したエラーのテキストをプロンプトとして Amazon Q 生成 SQL に送信します。Amazon Q 生成 SQL は、このようなエラーから学習し、生成する SQL を改善します。

SQL 検索パスにスキーマを追加し、そのスキーマを使用する必要があることを示します。例えば、データがパブリックスキーマではなく TICKIT スキーマである場合は、TICKIT スキーマを追加します。

```
set search_path to '$user', tickit;
```

Amazon Q 生成 SQL を利用する際の考慮事項

チャットパネルを使用する際は、次の点を考慮する必要があります。

- アカウントのクエリエディタ v2 の管理者が [Generative SQL settings] ページでチャット機能を有効にしている必要があります。
- Amazon Q 生成 SQL を使用するには、クエリエディタ v2 の AWS マネージドポリシーで指定されているその他のアクセス許可に加えて、IAM ポリシーでの

sqlworkbench:GetQSqlRecommendations アクセス許可が必要です。AWS 管理ポリシーの詳細については、「クエリエディタ v2 へのアクセス」を参照してください。

- 質問は英語で記述する必要があります。
- 質問は、クラスターまたはワークグループ内の接続されたデータベースに関連している必要があります。空の状態エラーの発生を避けるため、データベースに少なくとも 1 つのテーブルといくつかのデータが必要です。
- 質問は、接続されたデータベースに保存されているデータに関連している必要があります。外部スキーマを参照することはできません。サポートされているスキーマの詳細については、「Amazon Redshift データベースデベロッパーガイド」の「スキーマを作成する」を参照してください。
- 接続されたデータベースを変更する SQL を生成する質問の場合、警告が表示されることがあります。
- 生成 AI は新しいテクノロジーであり、応答にはハルシネーションと呼ばれる誤りがある場合があります。現在の環境やワークロードで使用する前に、すべてのコードをテストしてエラーや脆弱性がないかを確認する必要があります。
- アカウント内のその他のユーザーが実行した SQL クエリを共有すると、レコメンデーションを改善できます。アカウント管理者は次の SQL コマンドを実行して、アカウントのクエリ履歴へのアクセスを許可できます。

```
GRANT ROLE SYS:MONITOR to "IAMR:role-name";
GRANT ROLE SYS:MONITOR to "IAM:user-name";
GRANT ROLE SYS:MONITOR to "database-username";
```

SYS:MONITOR の詳細については、「Amazon Redshift データベースデベロッパーガイド」の「Amazon Redshift でのシステム定義のロール」を参照してください。

- お客様のデータは、安全かつプライベートです。アカウント間でお客様のデータが共有されることはありません。お客様のクエリ、データ、データベーススキーマは、生成 AI の基盤モデル (FM) のトレーニングに使用されることはありません。お客様が入力した内容は FM へのコンテキストプロンプトとして使用され、お客様のクエリのみに回答が提供されます。

生成 SQL の使用

適切なアクセス許可を設定した後、クエリエディタ v2 でノートブックを使用する際に、アイコンをクリックすると、会話を開始できます。

Amazon Q 生成 SQL チャットを操作して SQL を生成するには

1. クエリエディタ v2 の [エディタ] タブで、ノートブックを開きます。
2. 

   の生成 SQL アイコンをクリックして、指示に従ってチャットパネルで Amazon Redshift クエリエディタ v2 生成 SQL について質問します。

   プロンプトフィールドに質問を入力すると、Amazon Q 生成 SQL が推奨される SQL を返します。エラーが発生すると、チャットパネルで返されます。
3. [Add to notebook] をクリックして、プロンプトを使用して Markdown セルと、提案された SQL を含む SQL セルをノートブックに追加します。
4. (オプション) 役に立った

   フィードバックアイコンまたは役に立たなかった

   フィードバックアイコンをクリックして、生成された SQL についてのフィードバックを提供します。役に立たなかったというフィードバックは、`Incorrect tables/columns`、`Incorrect predicates/literals/group bys`、`Incorrect SQL structure`、または `Other` のカテゴリに分類できます。さらに、SQL の精度に関するフィードバックには、自由形式のテキストも追加できます。
5. (オプション) [Regenerate SQL] をクリックすると、同じプロンプトに対して別の応答が生成されます。現在のプロンプトに対して [Regenerate SQL] をクリックできるのは、1 回です。
6. (オプション) 生成 SQL チャットパネルで

   [詳細] アイコンをクリックしてから、[データベースの更新] をクリックすると、接続されたデータベースを説明するメタデータが更新されます。このメタデータには、データベース内のスキーマ、テーブル、列の定義が含まれます。

管理者の生成 SQL 設定の更新

適切な IAM アクセス許可を持つユーザーは、同じ AWS アカウント 内のその他のユーザーの [Generative SQL settings] を表示したり変更したりできます。このような管理者は、クエリエディタ v2 の AWS マネージドポリシーで指定されているその他のアクセス許可に加えて、IAM ポリシーで

の sqlworkbench:UpdateAccountQSqlSettings アクセス許可が必要です。管理ポリシーの詳細については、「クエリエディタv2 を使用するために必要なアクセス許可」を参照してください。

管理者がアカウント内のすべてのユーザーに対して生成 SQL チャットを有効にするには

1. 

   [設定] アイコンをクリックすると、さまざまな設定画面のメニューが表示されます。
2. 次に、生成 SQL 設定

   アイコンをクリックすると、[Generative SQL settings] ページが開きます。
3. [Q generative SQL settings] をクリックして、アカウント内のユーザーの生成 SQL 機能を有効にします。

   Amazon Q 生成 SQL を有効にした後、割り当てに残っているプロンプトの数を確認できます。クエリエディタ v2 管理者は、アカウントのユーザーの Amazon Q Developer Pro 階層の使用を有効にできます。Pro 階層を使用するには、IAM アイデンティティセンターでユーザーを設定して、各ユーザーを Amazon Q Developer Pro 階層にサブスクライブします。IAM アイデンティティセンターの Amazon Redshift に関する設定については、「Redshift を AWS IAM アイデンティティセンターに接続してシングルサインオンエクスペリエンスを提供する」を参照してください。Amazon Q Developer の料金については、「Amazon Q Developer の料金」を参照してください。

   Amazon Q Developer 無料利用枠を使用する場合、AWS アカウント のすべてのユーザーのプロンプトの合計数は、1 か月あたり 1,000 までに制限されます。Amazon Q Developer Pro 階層を使用する場合、個別のユーザーが送信できるプロンプトの合計数は、1 か月あたり 1,000 までに制限されます。利用できるプロンプトの数は、[設定] ページで確認できます。Amazon Q Developer の料金については、「Amazon Q Developer の料金」を参照してください。

カスタムコンテキスト

クエリエディタ v2 管理者は、生成された SQL を環境に合わせてカスタマイズするカスタムコンテキストを指定できます。カスタムコンテキストを使用すると、SQL 生成をきめ細かく制御するためのドメインに関するナレッジと設定を提供できます。カスタムコンテキストは、JSON ファイルで定義され、クエリエディタ v2 管理者はこのファイルを Amazon Q 生成 SQL にアップロードできます。

データウェアハウス向けに生成された SQL をパーソナライズするために使用できる JSON キーは、以下のとおりです。

すべてのテーブルリファレンスは、database.schema.table の 3 つの部分からなる表記に従う必要があります。

リソース

リソースは、カスタムコンテキストが適用されるデータアセットの範囲または部分を指定します。

ResourceId

リソースの一意の識別子を指定します。Amazon Redshift クラスターの場合は、cluster id を指定します。Redshift Serverless ワークグループの場合は、workgroup name を指定します。

ResourceType

有効な値: REDSHIFT_WAREHOUSE。

TablesToInclude

SQL 生成向けに使用されるテーブルのセットを指定します。このフィールドは、SQL クエリの範囲を使用可能なテーブルの定義済みのサブセットに制限する場合に重要です。テーブルへの不必要な参照を低減することで、生成プロセスの最適化につながります。このフィールドを TablesToExclude とペアリングすると、クエリ生成をより細かく制御できます。

TablesToExclude

SQL 生成から除外するテーブルのセットを指定します。これは、特定のテーブルが無関係である場合や、クエリ生成プロセスで考慮されるべきでない場合に使用します。

TableAnnotations

使用するテーブルに関するメタデータまたは補足情報を提供します。このような注釈には、テーブルの説明、使用上の注意、または Amazon Q 生成 SQL によるテーブルのコンテキストや構造の把握の強化につながるような追加の属性を含めることができます。このように、テーブル定義を明確にすることで SQL 生成の精度の向上につながります。

ColumnsToInclude

SQL クエリの生成時に、指定したテーブルのどの列を含めるかを定義します。このフィールドを使用してデータ取得の範囲を絞り込むことで、Amazon Q 生成 SQL は関連する列に集中することができ、パフォーマンスの向上につながります。これにより、Amazon Q 生成 SQL は、特定のクエリコンテキストに必要なデータのみをプルできます。

ColumnsToExclude

SQL 生成で考慮対象から除外する列を指定します。これは、Amazon Q 生成 SQL で考慮されるべきではない無関係または冗長なデータが特定の列に含まれている場合に使用できます。列の包含と除外を管理することで、結果を絞り込み、取得するデータの制御を維持できます。

ColumnAnnotations

`TableAnnotations` と同様に、このフィールドでは個別の列に固有のメタデータまたは注釈を提供します。このような注釈により、列定義や特別な処理手順に関するインサイトを提供することができます。この情報は、SQL 生成プロセスをガイドし、クエリで列が適切に使用されるようにするのに有益です。

CuratedQueries

事前定義済みの質問と応答の例のセットです。質問は自然言語 (NLQ) で記述され、応答は対応する SQL クエリです。このような例は、生成すべきクエリの種類を Amazon Q 生成 SQL に理解させるのに役立ちます。このような例は、Amazon Q 生成 SQL の出力の精度と関連性を向上させるためのリファレンスポイントとして機能します。

CustomDocuments

定義、ドメイン固有のナレッジ、説明など、Amazon Q 生成 SQL に提供される追加の情報またはヒントです。例えば、部門固有の方法で値を計算している場合に、ここで文書化できます。例えば「製造部門の総売上高は、料金 * 収益」などと指定します。このようなドキュメントは、追加のコンテキストを提供し、Amazon Q 生成 SQL の自然言語入力解釈機能の強化につながります。

AdditionalTables

データウェアハウスに保存されているデータの一部ではないが、SQL 生成の考慮対象であるべき追加のテーブルを指定します。これにより、Amazon Q 生成 SQL は外部データソースを SQL 生成ロジックに統合し、複雑なデータ環境への対応能力を拡張できます。

AppendToPrompt

SQL 生成プロセスをガイドするために Amazon Q 生成 SQL に提供する追加の手順またはガイドラインです。これには、クエリを構造化する方法に関する具体的な指示、特定の SQL コンストラクトの設定、または Amazon Q 生成 SQL の出力の品質を向上させるその他の大まかな指示があります。

次のカスタムコンテキストの例は、JSON ファイルの形式を説明しており、以下が定義されています。

- クラスター mycluster の Amazon Redshift データウェアハウスのカスタムコンテキストを定義します。
- SQL 生成プロセスの最適化に向けて、特定のテーブルと列を含めるか除外するかを定義します。
- 含めるテーブルと列の注釈を定義します。
- 使用する Amazon Q 生成 SQL のサンプルキュレーションクエリを定義します。
- SQL の生成に使用するカスタムドキュメントとガードレールを定義します。
- SQL の生成時に使用する追加のテーブルの DDL を定義します。

```
{
    "resources": [
        {
            "ResourceId": "mycluster",
            "ResourceType": "REDSHIFT_WAREHOUSE",
            "TablesToInclude": [
                "database.schema.table1",
                "database.schema.table2"
            ],
            "TablesToExclude": [
                "database.schema.table3",
                "database.schema.table4"
            ],
            "ColumnsToInclude": {
                "database.schema.table1": [
                    "col1",
                    "col2"
                ],
                "database.schema.table2": [
                    "col1",
                    "col2"
                ]
            },
            "ColumnsToExclude": {
                "database.schema.table5": [
                    "col1",
                    "col2"
                ],
                "database.schema.table6": [
                    "col1",
                    "col2"
                ]
```

```
            },
            "TableAnnotations": {
                "database.schema.table1": "table1 refers to Q3 sales",
                "database.schema.table2": "table2 refers to Q4 sales"
            },
            "ColumnAnnotations": {
                "database.schema.table1": {
                    "col1": "col1 refers to Q3 sale total",
                    "col2": "col2 refers to sale location"
                },
                "database.schema.table2": {
                    "col1": "col2 refers to Q4 sale total",
                    "col2": "col2 refers to sale location"
                }
            },
            "CuratedQueries": [
                {
                    "Question": "what is the sales data for Q3",
                    "Answer": "SELECT * FROM table1"
                },
                {
                    "Question": "what is the sales data for Q4",
                    "Answer": "SELECT * FROM table2"
                }
            ],
            "CustomDocuments": [
                "in manufacturing division total sales is price * revenue",
                "in research division total sales is price * revenue"
            ],
            "AdditionalTables": {
                "database.schema.table8": "create table database.schema.table8(col1
 int)",
                "database.schema.table9": "create table database.schema.table9(col1
 int)"
            },
            "AppendToPrompt": "Apply these guardrails: Queries should never return the
 secretId field of a user."
        }
    ]
}
```

チュートリアル: TICKIT データで Amazon Q 生成 SQL 機能を使用する例

SQL を生成するための効率的なプロンプトを作成するには、データベーススキーマとデータについての知識を持っている必要があります。TICKIT データは、2 つのファクトテーブルと 5 つのディメンションの 7 つのテーブルで構成されています。サンプルデータには、2008 年に開催されたエンターテイメントイベントの参加者の売上に関するレコードが含まれています。TICKIT データスキーマの詳細については、「Amazon Redshift Database デベロッパーガイド」の「サンプルデータベース」を参照してください。TICKIT データは、Amazon Redshift コンソールとクエリエディタ v2 の両方を使用して、さまざまな方法でデータベースにロードできます。クエリエディタ v2 には、TICKIT データを sample_data_dev データベースにロードする方法が提供されています。詳細については、「データベースにデータをロードする」を参照してください。クエリエディタ v2 には、TICKIT データのサンプルプロンプトも提供されています。次のシナリオでは、TICKIT サンプルデータに関する SQL を生成するための生成 SQL との会話について説明します。このシナリオでは、TICKIT サンプルデータは Amazon Redshift クラスターの dev データベースに既に作成されています。

> **Note**
>
> この例は、会話を説明する目的で提供されています。生成 SQL からの応答は、同じプロンプトを使用した場合の結果と一致するとは限りません。

Amazon Q 生成 SQL を使用した会話の例

1. [エディタ] で、TICKIT サンプルデータがあるクラスターまたはワークグループに接続します。
2. 空のノートブックを作成して、
   生成 SQL アイコンをクリックし、チャットパネルを開きます。
3. 次のプロンプトを入力して SQL を生成し、VENUE テーブルのレコード件数を確認します。

```
How many venues are there?
```

```
SELECT
  COUNT(*) AS num_venues
FROM
  tickit.venue
```

[Add to notebook] をクリックして、開いているノートブックにセルを 2 つ追加します。Markdown セルの 1 つに「会場はいくつありますか?」と入力します。もう 1 つのセルには生成された SQL が入力されます。

SQL セルで [実行する] をクリックすると、次のとおり結果が表示されます。

```
count
------
202
```

4. 別のバージョンの SQL をリクエストするには、[Regenerate SQL] をクリックすると、次のとおりの回答が表示されます。

```
SELECT
  venuestate,
  COUNT(*) AS num_venues
FROM
  tickit.venue
GROUP BY
  venuestate
```

[Add to notebook] をクリックして、開いているノートブックにセルをさらに 2 つ追加します。Markdown セルの 1 つに「会場はいくつありますか?」と入力します。もう 1 つのセルには生成された SQL が入力されます。

SQL セルで [実行する] をクリックすると、次のとおり会場のある州が番号別に一覧表示された結果が返されます。

```
venuestate num_venues
----------------------
MA         4
OH         8
MI         5
...
```

5. 生成 SQL アシスタントは、ユーザーの質問を推測できます。データベースにはウィジェットはないとはいえ、質問することはできます。この場合、SQL を生成できなかったというメッセージが表示されます。

```
How many widgets are there?
```

```
I was not able to generate any SQL
```

6. データベースについての理解を深めると、SQL 生成に向けてより具体的な質問をすることができます。例えば、データベースには 2008 年の数か月分の売上データが含まれていることはすでに把握しています。次のプロンプトを入力して SQL を生成し、2008 年 11 月の販売数を確認します。

```
How many sales were there in November 2008?
```

```
SELECT
  COUNT(*) AS num_sales
FROM
  tickit.sales s
  JOIN tickit.date d ON s.dateid = d.dateid
WHERE
  d.month = 'November'
  AND d.year = 2008
```

[Add to notebook] をクリックして、開いているノートブックにセルを 2 つ追加します。Markdown セルの 1 つに「2008 年 11 月の売上数は」と入力します。もう 1 つのセルには生成された SQL が入力されます。

SQL セルで [実行する] をクリックすると、次のとおり結果が表示されます。

```
num_sales
-----
0
```

これは期待していた結果ではないことが判明します。

7. 述語 `date.month='November'` では、DATE テーブルの月が月名の省略形で表されることを想定していることが分かりました。述語を `d.month='NOV'` に変更して SQL を再実行します。

```
SELECT
  COUNT(*)
```

```
FROM
  sales
  JOIN date ON sales.dateid = date.dateid
WHERE
  date.month = 'NOV'
  AND date.year = 2008
```

SQL セルで [実行する] をクリックして、新しい結果を取得します。

```
count
-----
14261
```

8. 接続されたデータベースを変更しようとする質問をすると、推奨される SQL とともに警告メッセージが返されます。次のプロンプトを入力して、テーブルにデータを挿入する SQL を生成します。

```
Insert 1 into the venue table.
```

```
INSERT
,
UPDATE
  OR delete data
FROM
  the database AS that could potentially change the data.Please provide a query
 that ONLY selects data
```

```
I was not able to generate the correct SQL code. I generated SQL, but you'll have
 to edit it to work with your database.
```

[Add to notebook] をクリックして開いているノートブックにセルを 2 つ追加して、この SQL を実行すると、SQL は失敗します。

```
ERROR: syntax error at or near "," Position: 132 [ErrorId:
 1-6546764a-011df2691778846219ce6ec2]
```

このシナリオでは、Amazon Q 生成 SQL を利用する際の基本的な方法の一部のみを説明しています。生成 AI テクノロジーをさらに試して、データベースにクエリを実行する SQL の作成を開始するうえで役に立ちます。

データベースにデータをロードする

クエリエディタ v2 を使用して Amazon Redshift クラスターまたはワークグループのデータベースにデータをロードできます。このセクションでは、サンプルデータ、S3 からのデータ、ローカルファイルのセットアップとワークフローからのデータをロードする方法について説明します。

サンプルデータ

クエリエディタ v2 には、サンプルデータベースと対応するスキーマにロードできるサンプルデータとノートブックが付属しています。

サンプルデータをロードするには、ロードするサンプルデータに関連付けられている

アイコンをクリックします。これにより、クエリエディタ v2 が `sample_data_dev` データベースのスキーマにデータをロードして、[Notebooks] フォルダに保存済みクエリのフォルダを作成します。

次のサンプルデータセットが利用可能です。

tickit

Amazon Redshift ドキュメントのほとんどの例では、`tickit` というサンプルデータを使用します。このデータは、7 個のテーブルで構成されています。そのうち 2 個はファクトテーブル、5 個はディメンションです。このデータをロードすると、スキーマ `tickit` がサンプルデータで更新されます。`tickit` データの詳細については、「Amazon Redshift データベースデベロッパーガイド」の「サンプルデータベース」を参照してください。

tpch

このデータは、決定サポートのベンチマークに使用されます。このデータをロードすると、スキーマ `tpch` がサンプルデータで更新されます。`tpch` データの詳細については、「TPC-H」を参照してください。

tpcds

このデータは、決定サポートのベンチマークに使用されます。このデータをロードすると、スキーマ `tpcds` がサンプルデータで更新されます。`tpcds` データの詳細については、「TPC-DS」を参照してください。

Amazon S3 からデータをロードする

既存のテーブルまたは新しいテーブルに Amazon S3 データをロードできます。

データを既存のテーブルにロードするには

クエリエディタ v2 は、Amazon S3 からデータをロードするために COPY コマンドを使用します。クエリエディタ v2 の [データのロード] ウィザードで生成および使用される COPY コマンドは、Amazon S3 からのコピー時に COPY コマンド構文で使用できる、多くのパラメータをサポートしています。Amazon S3 からデータをロードするための COPY コマンドと使用するオプションの詳細については、Amazon Redshift データベース開発者ガイドの「Amazon S3 からの COPY」を参照してください。

1. データをロードするデータベースに、テーブルがすでに作成されていることを確認します。
2. 次に進む前に、クエリエディタ v2 のツリービューパネルで、ターゲットデータベースに接続していることを確認します。コンテキストメニュー (右クリック) を使用して、データをロードするクラスターまたはワークグループへの接続を作成できます。

   [Load data] (データをロード) をクリックします。
3. [データソース] では、[S3 バケットからロード] を選択します。
4. [S3 URI] で [Browse S3] (S3 をブラウズ) をクリックして、ロードするデータが含まれている Amazon S3 バケットを検索します。
5. 指定した Amazon S3 バケットがターゲットテーブルと同じ AWS リージョン にない場合、[S3 file location] (S3 ファイルの場所) でデータが置かれた AWS リージョン を選択します。
6. Amazon S3 ファイルが実際に複数の Amazon S3 バケット URI を含むマニフェストである場合。[This file is a manifest file] (このファイルはマニフェストファイルです) を選択します。
7. アップロードするファイルの [File format] (ファイル形式) を選択します。サポートされているデータ形式
   は、CSV、JSON、DELIMITER、FIXEDWIDTH、SHAPEFILE、AVRO、PARQUET、ORC です。指定したファイル形式に応じて、それぞれの [File options] (ファイルオプション) を選択できます。また、データが暗号化されており、暗号化に使用する KMS キーの Amazon リソースネーム (ARN) を入力した場合には、[Data is encrypted] (データは暗号化されています) を選択できます。

   CSV または DELIMITER を選択すると、指定した行数が実際には列名であり、ロードするデータではない場合に、[区切り文字] で [ヘッダー行を無視する] を選択することもできます。

8. 圧縮方法を選択してファイルを圧縮します。デフォルトでは圧縮なしです。
9. (オプション) [Advanced settings] (詳細設定) では、さまざまな [Data conversion parameters] (データ変換パラメータ) や [Load operations] (ロードオペレーション) がサポートされています。ファイルの必要に応じて、この情報を入力します。

   データ変換パラメータとデータロードパラメータの詳細については、「Amazon Redshift データベースデベロッパーガイド」の「データ変換パラメータ」と「データのロード操作」を参照してください。
10. [次へ] を選択します。
11. [既存のテーブルをロード] を選択します。
12. [クラスターまたはワークグループ]、[データベース]、[スキーマ]、[テーブル] の名前など、データをロードする先の [ターゲットテーブル] の場所を確認または選択します。
13. Amazon S3 からデータをロードするために必要な許可を持つ [IAM role] (IAM ロール) を選択します。
14. (オプション) 列名を選択して [Column mapping] (列のマッピング) に入力すると、入力データファイルの順序で列がマッピングされます。
15. [Load data] (データをロード) をクリックして、データのロードを開始します。

    ロードが完了すると、データのロードのために生成された COPY コマンドとともに、クエリエディタが表示されます。COPY の [Result] (結果) は次のように表示されます。成功すると、ロードされたテーブルから SQL を使用してデータを選択できるようになります。エラーが発生した場合は、システムビュー STL_LOAD_ERRORS にクエリを実行して、詳細を取得します。COPY コマンドとそのエラーについては、Amazon Redshift データベース開発者ガイドの「STL_LOAD_ERRORS」を参照してください。

新しいテーブルにデータをロードすると、クエリエディタ v2 は最初にデータベースにテーブルを作成してから、同じワークフローの別のアクションとしてデータをロードします。

データを新しいテーブルにロードするには

クエリエディタ v2 は、Amazon S3 からデータをロードするために COPY コマンドを使用します。クエリエディタ v2 の [データのロード] ウィザードで生成および使用される COPY コマンドは、Amazon S3 からのコピー時に COPY コマンド構文で使用できる、多くのパラメータをサポートしています。Amazon S3 からデータをロードするための COPY コマンドと使用するオプションの詳細については、Amazon Redshift データベース開発者ガイドの「Amazon S3 からの COPY」を参照してください。

1. 次に進む前に、クエリエディタ v2 のツリービューパネルで、ターゲットデータベースに接続していることを確認します。コンテキストメニュー (右クリック) を使用して、データをロードするクラスターまたはワークグループへの接続を作成できます。

   [Load data] (データをロード) をクリックします。
2. [データソース] では、[S3 バケットからロード] を選択します。
3. [S3 URI] で [Browse S3] (S3 をブラウズ) をクリックして、ロードするデータが含まれている Amazon S3 バケットを検索します。
4. 指定した Amazon S3 バケットがターゲットテーブルと同じ AWS リージョン にない場合、[S3 file location] (S3 ファイルの場所) でデータが置かれた AWS リージョン を選択します。
5. Amazon S3 ファイルが実際に複数の Amazon S3 バケット URI を含むマニフェストである場合。[This file is a manifest file] (このファイルはマニフェストファイルです) を選択します。
6. アップロードするファイルの [File format] (ファイル形式) を選択します。サポートされているデータ形式
   は、CSV、JSON、DELIMITER、FIXEDWIDTH、SHAPEFILE、AVRO、PARQUET、ORC です。指定したファイル形式に応じて、それぞれの [File options] (ファイルオプション) を選択できます。また、データが暗号化されており、暗号化に使用する KMS キーの Amazon リソースネーム (ARN) を入力した場合には、[Data is encrypted] (データは暗号化されています) を選択できます。

   CSV または DELIMITER を選択すると、指定した行数が実際には列名であり、ロードするデータではない場合に、[区切り文字] で [ヘッダー行を無視する] を選択することもできます。
7. 圧縮方法を選択してファイルを圧縮します。デフォルトでは圧縮なしです。
8. (オプション) [Advanced settings] (詳細設定) では、さまざまな [Data conversion parameters] (データ変換パラメータ) や [Load operations] (ロードオペレーション) がサポートされています。ファイルの必要に応じて、この情報を入力します。

   データ変換パラメータとデータロードパラメータの詳細については、「Amazon Redshift データベースデベロッパーガイド」の「データ変換パラメータ」と「データのロード操作」を参照してください。
9. [次へ] を選択します。
10. [新しいテーブルをロード] を選択します。

テーブルの列は入力データから推測されます。列とテーブルの詳細を追加すると、テーブルスキーマの定義を変更できます。クエリエディタ v2 の推測されたテーブルスキーマに戻すには、[デフォルトに戻す] を選択します。

11. [クラスターまたはワークグループ]、[データベース]、[スキーマ] など、データをロードする先の [ターゲットテーブル] の場所を確認または選択します。作成するテーブルの名前を入力します。
12. Amazon S3 からデータをロードするために必要な許可を持つ [IAM role] (IAM ロール) を選択します。
13. [テーブルを作成] を選択し、表示されている定義を使用してテーブルを作成します。

    テーブル定義のレビューの要約が表示されます。テーブルがデータベースに作成されます。後でテーブルを削除するには、DROP TABLE SQL コマンドを実行します。詳細については、Amazon Redshift データベースデベロッパーガイドの「DROP TABLE」を参照してください。
14. [Load data] (データをロード) をクリックして、データのロードを開始します。

    ロードが完了すると、データのロードのために生成された COPY コマンドとともに、クエリエディタが表示されます。COPY の [Result] (結果) は次のように表示されます。成功すると、ロードされたテーブルから SQL を使用してデータを選択できるようになります。エラーが発生した場合は、システムビュー STL_LOAD_ERRORS にクエリを実行して、詳細を取得します。COPY コマンドとそのエラーについては、Amazon Redshift データベース開発者ガイドの「STL_LOAD_ERRORS」を参照してください。

ローカルファイル設定とワークフローからのデータのロード

ローカルファイルから既存のテーブルまたは新しいテーブルにデータをロードできます。

ローカルファイルからデータをロードするための管理者設定

クエリエディタ v2 の管理者は、[Account settings] (アカウント設定) ウィンドウで共通の Amazon S3 バケットを指定する必要があります。アカウントユーザーは適切な権限で設定されている必要があります。

- 必要な IAM 権限 – ローカルファイルからロードするユーザーには、`s3:ListBucket`、`s3:GetBucketLocation`、`s3:putObject`、`s3:getObject`、および `s3:deleteObject` 権限が必要です。*`optional-prefix`* を指定すると、このバケットのクエリエディタ v2 関連の使用をこのプレフィックスの付いたオブジェクトに制限できます。このオプションは、同じ Amazon S3 バケットをクエリエディタ v2 以外の用途に使用する場合に使用で

きます。バケットとプレフィックスの詳細については、「Amazon Simple Storage Service」ユーザーガイドの「Managing user access to specific folders」(特定のフォルダへのユーザーアクセスの管理) を参照してください。ユーザー間のデータアクセスが許可されないようにするには、クエリエディタ v2 管理者が Amazon S3 バケットポリシーを使用して、aws:userid に基づいてオブジェクトアクセスを制限することをお勧めします。次の例では、*<staging-bucket-name>* に、aws:userid というプレフィックスがある Amazon S3 オブジェクトのみへの読み取り/書き込み Amazon S3 アクセス権限を許可します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:GetBucketLocation"
            ],
            "Resource": [
                "arn:aws:s3:::<staging-bucket-name>"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::<staging-bucket-name>[/<optional-prefix>]/
${aws:userid}/*"
            ]
        }
    ]
}
```

- データの分離 – ユーザーがお互いのデータに (たとえ短時間でも) アクセスできないようにすることをお勧めします。ローカルファイルからのロードでは、クエリエディタ v2 管理者が設定したステージング Amazon S3 バケットを使用します。ステージングバケットのバケットポリシーを設定

して、ユーザー間のデータを分離します。次の例は、*<staging-bucket-name>* のユーザー間でデータを分離するバケットポリシーを示しています。

JSON

```
{
 "Version":"2012-10-17",
    "Statement": [
        {"Sid": "userIdPolicy",
            "Effect": "Deny",
            "Principal": "*",
            "Action": ["s3:PutObject",
                       "s3:GetObject",
                       "s3:DeleteObject"],
            "NotResource": [
                "arn:aws:s3:::<staging-bucket-name>[/<optional-prefix>]/
${aws:userid}/*"
            ]
         }
    ]
}
```

ローカルファイルからのデータのロード

ローカルファイルデータを既存のテーブルにロードするには

クエリエディタ v2 の管理者は、アカウント設定ウィンドウで共通の Amazon S3 バケットを指定する必要があります。クエリエディタ v2 は、アカウントが使用する共通の Amazon S3 バケットにローカルファイルを自動的にアップロードしてから、COPY コマンドを使用してデータをロードします。クエリエディタ v2 のローカルファイルの読み込みウィンドウで生成および実行される COPY コマンドは、Amazon S3 からのコピー時に COPY コマンド構文で使用できる、多くのパラメータをサポートしています。Amazon S3 からデータをロードするための COPY コマンドと使用するオプションの詳細については、「Amazon Redshift データベース開発者ガイド」の「Amazon S3 からの COPY」を参照してください。

1. データをロードするデータベースに、テーブルがすでに作成されていることを確認します。
2. クエリエディタ v2 のツリービューパネルで、ターゲットデータベースに接続していることを確認します。コンテキストメニュー (右クリック) を使用して、データをロードするクラスターまたはワークグループへの接続を作成できます。

3.  [Load data] (データをロード) をクリックします。
4. [Data source] (データソース) で、[Load from local file] (ローカルファイルからロード) を選択します。
5. [ブラウズする] を選択して、データが含まれるファイルを検索してファイルをロードします。デフォルトでは、拡張子 `.csv`、`.avro`、`.parquet`、`.orc` のファイルが表示されますが、他のファイルタイプも選択できます。 ファイルの最大サイズは 100 MB です。
6. アップロードするファイルの [File format] (ファイル形式) を選択します。サポートされているデータ形式
   は、CSV、JSON、DELIMITER、FIXEDWIDTH、SHAPEFILE、AVRO、PARQUET、ORC です。指定したファイル形式に応じて、それぞれの [File options] (ファイルオプション) を選択できます。また、データが暗号化されており、暗号化に使用する KMS キーの Amazon リソースネーム (ARN) を入力した場合には、[Data is encrypted] (データは暗号化されています) を選択できます。

   CSV または DELIMITER を選択すると、指定した行数が実際には列名であり、ロードするデータではない場合に、[区切り文字] で [ヘッダー行を無視する] を選択することもできます。
7. (オプション) [Advanced settings] (詳細設定) では、さまざまな [Data conversion parameters] (データ変換パラメータ) や [Load operations] (ロードオペレーション) がサポートされています。ファイルの必要に応じて、この情報を入力します。

   データ変換パラメータとデータロードパラメータの詳細については、「Amazon Redshift データベースデベロッパーガイド」の「データ変換パラメータ」と「データのロード操作」を参照してください。
8. [次へ] を選択します。
9. [既存のテーブルをロード] を選択します。
10. [クラスターまたはワークグループ]、[データベース]、[スキーマ]、[テーブル] の名前など、データをロードする先の [ターゲットテーブル] の場所を確認または選択します。
11. (オプション) 列名を選択して [Column mapping] (列のマッピング) に入力すると、入力データの順序で列をマッピングすることができます。
12. [Load data] (データをロード) をクリックして、データのロードを開始します。

    ロードが完了すると、ロードが成功したかどうかのメッセージが表示されます。成功すると、ロードされたテーブルから SQL を使用してデータを選択できるようになります。エラーが発生した場合は、システムビュー STL_LOAD_ERRORS にクエリを実行して、詳細を取得しま

す。COPY コマンドとそのエラーについては、Amazon Redshift データベース開発者ガイドの「STL_LOAD_ERRORS」を参照してください。

データをロードするために使用された COPY コマンドテンプレートが [Query history] (クエリ履歴) に表示されます。この COPY コマンドテンプレートには、使用されているパラメータの一部が表示されますが、エディタタブで直接実行することはできません。クエリ履歴の詳細については、「クエリとタブの履歴を表示する」を参照してください。

新しいテーブルにデータをロードすると、クエリエディタ v2 は最初にデータベースにテーブルを作成してから、同じワークフローの別のアクションとしてデータをロードします。

ローカルファイルデータを新しいテーブルにロードするには

クエリエディタ v2 の管理者は、[Account settings] (アカウント設定) ウィンドウで共通の Amazon S3 バケットを指定する必要があります。ローカルファイルは、アカウントが使用する共通の Amazon S3 バケットに自動的にアップロードされ、クエリエディタ v2 は COPY コマンドを使用してデータをロードします。クエリエディタ v2 のローカルファイルの読み込みウィンドウで生成および実行される COPY コマンドは、Amazon S3 からのコピー時に COPY コマンド構文で使用できる、多くのパラメータをサポートしています。Amazon S3 からデータをロードするための COPY コマンドと使用するオプションの詳細については、「Amazon Redshift データベース開発者ガイド」の「Amazon S3 からの COPY」を参照してください。

1. クエリエディタ v2 のツリービューパネルで、ターゲットデータベースに接続していることを確認します。コンテキストメニュー (右クリック) を使用して、データをロードするクラスターまたはワークグループへの接続を作成できます。
2.  [Load data] (データをロード) をクリックします。
3. [Data source] (データソース) で、[Load from local file] (ローカルファイルからロード) を選択します。
4. [ブラウズする] を選択して、データが含まれるファイルを検索してファイルをロードします。デフォルトでは、拡張子 `.csv`、`.avro`、`.parquet`、`.orc` のファイルが表示されますが、他のファイルタイプも選択できます。 ファイルの最大サイズは 100 MB です。
5. アップロードするファイルの [File format] (ファイル形式) を選択します。サポートされているデータ形式は、CSV、JSON、DELIMITER、FIXEDWIDTH、SHAPEFILE、AVRO、PARQUET、ORC です。指定したファイル形式に応じて、それぞれの [File options] (ファイルオプション) を選択で

きます。また、データが暗号化されており、暗号化に使用する KMS キーの Amazon リソースネーム (ARN) を入力した場合には、[Data is encrypted] (データは暗号化されています) を選択できます。

CSV または DELIMITER を選択すると、指定した行数が実際には列名であり、ロードするデータではない場合に、[区切り文字] で [ヘッダー行を無視する] を選択することもできます。

6. (オプション) [Advanced settings] (詳細設定) では、さまざまな [Data conversion parameters] (データ変換パラメータ) や [Load operations] (ロードオペレーション) がサポートされています。ファイルの必要に応じて、この情報を入力します。

   データ変換パラメータとデータロードパラメータの詳細については、「Amazon Redshift データベースデベロッパーガイド」の「データ変換パラメータ」と「データのロード操作」を参照してください。

7. [次へ] を選択します。

8. [新しいテーブルをロード] を選択します。

9. [クラスターまたはワークグループ]、[データベース]、[スキーマ] など、データをロードする先の [ターゲットテーブル] の場所を確認または選択します。作成するテーブルの名前を入力します。

10. [テーブルを作成] を選択し、表示されている定義を使用してテーブルを作成します。

    テーブル定義のレビューの要約が表示されます。テーブルがデータベースに作成されます。後でテーブルを削除するには、DROP TABLE SQL コマンドを実行します。詳細については、Amazon Redshift データベースデベロッパーガイドの「DROP TABLE」を参照してください。

11. [Load data] (データをロード) をクリックして、データのロードを開始します。

    ロードが完了すると、ロードが成功したかどうかを示すメッセージが表示されます。成功すると、ロードされたテーブルから SQL を使用してデータを選択できるようになります。エラーが発生した場合は、システムビュー STL_LOAD_ERRORS にクエリを実行して、詳細を取得します。COPY コマンドとそのエラーについては、Amazon Redshift データベース開発者ガイドの「STL_LOAD_ERRORS」を参照してください。

    データをロードするために使用された COPY コマンドテンプレートが [Query history] (クエリ履歴) に表示されます。この COPY コマンドテンプレートには、使用されているパラメータの一部が表示されますが、エディタタブで直接実行することはできません。クエリ履歴の詳細については、「クエリとタブの履歴を表示する」を参照してください。

Amazon Redshift を使用したクエリの作成

エディタでクエリを入力するか、[クエリ] リストをクリックし [実行] をクリックします。

デフォルトでは、[上限 100] で、結果を 100 行に制限するように設定されています。このオプションをオフにすると、より大きな結果セットが返されます。このオプションをオフにすると、非常に大きな結果セットを避けるために、SQL 文に LIMIT オプションを含めることができます。詳細については、「Amazon Redshift データベースデベロッパーガイド」の「ORDER BY clause」(句の順に表示)を参照してください。

結果領域にクエリプランを表示するには [説明] をオンにします。[Explain graph] (グラフの説明) をオンにしても説明プランのグラフィカルな表現が表示されます。

クエリを [クエリ] フォルダに保存するには [保存]を選択します。

クエリが成功すると、成功メッセージが表示されます。クエリが情報を返す場合、結果が [結果] セクションに移動します。結果の数が表示領域を超えると、結果領域の上部に数値が表示されます。数値を選択して、結果の連続したページを表示することができます。

列ごとに [結果] の絞り込みや並べ替えができます。結果列のヘッダーに絞り込み条件を入力するには、列の上にマウスポインターを置くと、メニュー (☰) で、列を絞り込む条件を入力できます。

クエリにエラーが含まれている場合、クエリエディタ v2 は結果領域にエラーメッセージを表示します。このメッセージは、クエリを修正する方法について説明します。

次のように結果領域で右クリックし、コンテキストニューを表示すると、クエリの結果をエクスポートまたはコピーできます。

- ファイルに選択した行をダウンロードするには、[コピー先] と [JSON] または [CSV] を選択します。
- [行をコピー] を選択して、選択した行をクリップボードにコピーします。
- [行とヘッダーをコピー] を選択して、選択した行と列ヘッダーをクリップボードにコピーします。

結果セットの [エクスポート] を選択し、[JSON] または [CSV] を選択して、行形式の結果のセット全体をファイルにダウンロードすることもできます。結果セットの行数は、[Limit] (制限) オプションまたはクエリ内の SQL `limit` 句で制限できます。ダウンロードされた結果セットの最大サイズは 5 MB です。

ショートカット Ctrl+C (Windows) または Cmd+C (macOS) を使用しても、現在の結果ページからクリップボードにデータをコピーできます。行が選択されていない場合は、フォーカスのあるセルがクリップボードにコピーされます。行が選択されている場合は、選択されている行がクリップボードにコピーされます。

新しいクエリタブを追加するには、
アイコンを選択し、クエリタブのある行に表示されている [Editor] (エディタ) を選択します。クエリタブは `Isolated session` を使用する場合と使用しない場合があります。分離されたセッションでは、あるエディタタブで一時テーブルを作成するなどの SQL コマンドの結果は、別のエディタタブには表示されません。クエリエディタ v2 でエディタタブを開くとき、デフォルトは分離されたセッションです。

クエリを実行する

1. クエリ領域で、以下のいずれかの操作を行います。

   - クエリを入力します。
   - コピーしたクエリを貼り付けます。
   - [クエリ] フォルダを選択し、右クリックで保存したクエリのコンテキストメニューを開いて [クエリを開く] を選択します。

2. 実行する予定の SQL に対して、正しい [Cluster] (クラスター) または [Workgroup] (ワークグループ) と [Database] (データベース) の値を選択したことを確認します。

   ツリービューで [Cluster] (クラスター) または [Workgroup] (ワークグループ) を選択します。ツリービューでは [Database] (データベース) も選択します。

   各エディタタブ内の [クラスター]、[ワークグループ]、[データベース] を、各エディタタブの [分離されたセッション] ヘッダーの近くにあるドロップダウンコントロールを使用して変更できます。

   エディタタブごとに、SQL を分離されたセッションで実行するかどうかを選択します。分離されたセッションには、データベースへの独自の接続があります。これを使用して、他のクエリエディタセッションから分離された SQL を実行します。接続の詳細については、「クエリエディタ v2 を開く」を参照してください。

3. [Run] (実行) を選択します。

   [結果] 領域が開き、クエリ結果が表示されます。

クエリの説明プランを表示する手順は、次のとおりです。

1. クエリを選択します。
2. [説明] をオンにします。

   デフォルトでは [説明グラフ] もオンです。
3. [実行する] を選択します。

   クエリが実行され、説明プランがクエリ [結果] 領域に表示されます。

クエリエディタ v2 では次の機能をサポートしています。

- 1 つのクエリタブで複数の SQL 文を使用してクエリを作成できます。クエリは連続して実行され、各クエリに対して複数の結果タブが開きます。
- セッション変数と一時テーブルを使用してクエリを作成できます。
- ${*parameter*} で指定した置換可能なパラメータを使用してクエリを作成できます。複数の置換可能なパラメータを使用して SQL クエリを作成し、SQL 文の複数の場所で同じパラメータを使用できます。

  クエリが実行されると、パラメータの値を入力するためのウィンドウが表示されます。クエリを実行するたびに、パラメータ値を入力するためのウィンドウが表示されます。

  例については、例:特定のパラメータより大きい売上を参照してください。
- クエリは自動的にバージョン管理されます。実行するクエリの以前のバージョンを選択できます。
- ワークフローを続行する前に、クエリが完了するまで待つ必要はありません。クエリエディタを閉じた場合でも、クエリは引き続き実行されます。
- クエリの作成時には、スキーマ名、テーブル名、列名のオートコンプリートがサポートされています。

SQL エディタでは次の機能をサポートしています。

- SQL で使用される開始と終了の括弧の色は同じです。エディタには、括弧を一致させるのに役立つ縦線が表示されます。
- SQL のセクションは、展開および折りたたむことができます。
- SQL 内のテキストを検索して置換できます。
- ショートカットキーは、一般的な編集作業の一部で使用できます。

- SQL エラーはエディタで強調表示され、問題箇所を簡単に特定できます。

編集機能のデモについては、「Amazon Redshift クエリエディタ v2 で新規に強化された編集エクスペリエンス」の動画をご覧ください。

クエリの例

次に、実行できるクエリのさまざまなタイプの説明をします。

これらのクエリの多くで使用されているデータは tickit サンプルスキーマのものです。tickit サンプルデータのロードの詳細については、「データベースにデータをロードする」を参照してください。tickit サンプルデータの詳細については、「Amazon Redshift データベースデベロッパーガイド」の「サンプルデータベース」を参照してください。

これらのサンプルクエリを実行する際には、エディタで正しいデータベースを選択していることを確認してください (sample_data_devなど) 。

トピック

- 例:セッション変数の設定
- 例:売上合計による上位のイベント
- 例:特定のパラメータより大きい売上
- 例: 一時テーブルの作成
- 例: 一時テーブルからの選択

例:セッション変数の設定

次のコマンドでは、セッションの search_path サーバー設定パラメータを 公開 に設定しています。詳細については、「Amazon Redshift データベースデベロッパーガイド」の 「SET」と「search_path」を参照してください。

```
set search_path to public;
```

例:売上合計による上位のイベント

次のクエリは、売上が最も多いイベントを検索します。

```
select eventname, count(salesid) totalorders, sum(pricepaid) totalsales
from sales, event
```

```
where sales.eventid=event.eventid
group by eventname
order by 3;
```

以下に、結果の一部を示します。

```
eventname           totalorders       totalsales
White Christmas         20              9352
Joshua Radin            38             23469
Beach Boys              58             30383
Linda Ronstadt          56             35043
Rascal Flatts           76             38214
Billy Idol              67             40101
Stephenie Meyer         72             41509
Indigo Girls            57             45399
...
```

例:特定のパラメータより大きい売上

次のクエリは、販売数量が ${numberoforders} で指定された パラメータよりも大きい売上を検索します。パラメータ値が 7 の場合、結果は 60 行です。クエリを実行すると、クエリエディタ v2 は、SQL 文のパラメータの値を収集するための[クエリの実行フォーム] ウィンドウを表示します。

```
select salesid, qtysold
from sales
where qtysold > ${numberoforders}
order by 2;
```

以下に、結果の一部を示します。

```
salesid qtysold
20005 8
21279 8
130232 8
42737 8
74681 8
67103 8
105533 8
91620 8
121552 8
...
```

例: 一時テーブルの作成

次の文は、テンポラリテーブルイベントセールスから情報を選択して[セールス] および [イベント] テーブル。

```
create temporary table eventsalestemp as
select eventname, count(salesid) totalorders, sum(pricepaid) totalsales
from sales, event
where sales.eventid=event.eventid
group by eventname;
```

例: 一時テーブルからの選択

次の文は、一時テーブル [eventsalestemp] から、イベント、総注文数、総売上高を総注文数の順に選択します。

```
select eventname,  totalorders,  totalsales
from eventsalestemp
order by 2;
```

以下に、結果の一部を示します。

```
eventname          totalorders   totalsales
White Christmas         20          9352
Joshua Radin            38         23469
Martina McBride         50         52932
Linda Ronstadt          56         35043
Indigo Girls            57         45399
Beach Boys              58         30383
...
```

Amazon Redshift のノートブック

ノートブックを使用すると、1 つのドキュメント内で複数の SQL クエリを編成、注釈付け、共有できます。ノートブックには、複数の SQL クエリセルと Markdown セルを追加できます。ノートブックは、複数のクエリセルと Markdown セルを使用して、データ分析に関連するクエリと説明を 1 つのドキュメントにグループ化する方法を提供します。Markdown 構文を使用してテキストを追加し、外観の形式を設定して、データ分析タスクのコンテキストと追加情報を提供できます。ノートブックは、チームのメンバーと共有することが可能です。

ノートブックを使用するには、ノートブックの権限を IAM プリンシパル (IAM ユーザーまたは IAM ロール) に追加する必要があります。ベストプラクティスとして、アクセス許可ポリシーを IAM ロールにアタッチし、それを必要に応じてユーザーやグループに割り当てることをお勧めします。詳細については、「Amazon Redshift での Identity and Access Management」を参照してください。権限は、クエリエディタ v2 管理ポリシーのいずれかに追加できます。詳細については、「クエリエディタ v2 へのアクセス」を参照してください。

ノートブックのセルを順番にすべて実行できます。ノートブックの SQL クエリセルには、クエリエディタのタブとほとんど同じ機能があります。詳細については、「Amazon Redshift を使用したクエリの作成」を参照してください。以下は、クエリエディタのタブとノートブックの SQL セルの違いです。

- ノートブックには SQL ステートメントで `Explain` を実行するコントロールはありません。
- ノートブックの SQL セルごとに 1 つのグラフのみを作成できます。

クエリエディタ v2 で作成したファイルにノートブックをエクスポートおよびインポートできます。ファイル拡張子は `.ipynb` で、ファイルサイズは最大 5 MBです。SQL セルと Markdown セルはファイルに保存されます。クラスターまたはワークグループとデータベースは、エクスポートされたノートブックに保存されません。インポートしたノートブックを開くと、クラスターまたはワークグループ、およびそれを実行するデータベースを選択したことになります。SQL セルを実行したら、[Results] (結果) タブで現在の結果ページをグラフとして表示するかどうかを選択できます。クエリの結果セットは、ノートブックには保存されません。

ノートブックの作成

ノートブックを作成すると、1 つのドキュメント内で複数の SQL クエリを編成、注釈付け、共有できます。

ノートブックを作成するには

1. ナビゲーターメニューから、[エディタ] アイコン
   ()
   を選択します。
2. プラスアイコン
   ()
   を選択し、[ノートブック] を選択します。

デフォルトでは、ノートブックに SQL クエリセルが表示されます。

3. SQL クエリセルで、以下のいずれかの操作を行います。

   - クエリを入力します。
   - コピーしたクエリを貼り付けます。

4. (オプション) プラスのアイコン

   ()

   を選択し、次に [Markdown] をクリックして、標準の Markdown 構文を使用して説明または解説のためのテキストの提供が可能な Markdown セルを追加します。

5. (オプション) プラスのアイコン

   ()、[SQ

   の順に選択し、SQL セルを挿入します。

ノートブックの名前は鉛筆

()

アイコンで変更できます。

メニューアイコン

(•••)

から、ノートブックで次のオペレーションを実行することもできます。

- 

  [Share with my team] (チームと共有) — タグで定義されているとおりにノートブックをチームと共有します。ノートブックをチームと共有するには、プリンシパルタグ `sqlworkbench-team` をアカウントの残りのチームメンバーと同じ値に設定します。たとえば、管理者は `accounting-team` の値を会計部門のすべての人のために設定します。例については、「クエリエディタv2 を使用するために必要なアクセス許可」を参照してください。

- 

  [Export] (エクスポート) — ノートブックを `.ipynb` 拡張子の付いたローカルファイルにエクスポートします。

- 

  [クエリのインポート] – ローカルファイルからノートブックのセルにクエリをインポートします。`.sql` および `.txt` 拡張子を持つファイルをインポートできます。

- 

  [Save version] (バージョンを保存) — ノートブックのバージョンを作成します。ノートブックのバージョンを確認するには、保存したノートブックに移動して [Version history] (バージョン履歴) を開きます。
- 

  [Duplicate] (複製) — ノートブックのコピーを作成し、新しいノートブックタブで開きます。
- 

  [Shortcuts] (ショートカット) — ノートブックの作成時に使用できるショートカットを表示します。

ノートブックへのインポート

ノートブック全体または個々の SQL セルをクエリエディタ v2 ノートブックにインポートできます。

ノートブック全体をローカルファイルから [マイノートブック] にインポートするには、
[インポート] を選択し、[ノートブックのインポート] を選択します。ノートブックを含む `.ipynb` ファイルに移動します。ノートブックは、現在開いているノートブックフォルダにインポートされます。これで、ノートブックエディタを使用してノートブックを開くことができます。

ローカルファイルからノートブックの SQL セルにクエリをインポートするには、
[インポート] を選択し、[クエリのインポート] を選択します。[クエリのインポート] ウィンドウで、画面の指示に従って、新しいノートブックまたは既存のノートブックにクエリとしてインポートできるファイルとフォルダを選択します。ファイルには、`.sql` または `.txt` の拡張子が必要です。各クエリは最大 10,000 文字です。既存のノートブックに追加するときは、[保存されたノートブック] リストにあるすべてのノートブックから、どのノートブックに追加するかを選択します。インポートされたクエリは、ノートブックの最後に SQL セルとして追加されます。新しいノートブックを選択するときは、ノートブックの名前を選択すると、現在開いている [保存されたノートブック] フォルダに作成されます。

> **Note**
>
> TextEdit アプリケーションを使用して macOS で `.sql` ファイルを作成すると、ファイルに隠し拡張子が追加される問題が発生する可能性があります。例えば、TextEdit で作成された

Test.sql という名前のファイルが Test.sql.rtf として保存される可能性があります。クエリエディタ v2 は、.rtf 拡張子のファイルをサポートしていません。ただし、TextEdit を使用して .sql ファイルを作成し、プレーンテキストファイルとして保存した場合、ファイルには追加の非表示の .txt 拡張子が付きます。例えば、Text.sql という名前のファイルが Text.sql.txt として保存される場合があります。.rtf 拡張子とは異なり、クエリエディタ v2 は .txt 拡張子を持つファイルをサポートしているため、ノートブックにクエリをインポートするときに Text.sql.txt がサポートされます。

ノートブックのデモについては、「Amazon Redshift SQL Notebooks in query editor v2」(クエリエディタ v2 の Amazon Redshift SQL ノートブック)の動画をご覧ください。

AWS Glue Data Catalog に対するクエリの実行

クエリエディタ v2 を使用して、特定の SQL コマンドを使用し、このセクションで説明されているアクセス許可を付与することで、AWS Glue Data Catalog にカタログ化されたデータをクエリできます。デフォルトでは、AWS Glue Data Catalog は、awsdatacatalog という名前のクエリエディタ v2 のデータベースとして表示されます。AWS Glue Data Catalog に対するクエリの実行は、すべての Amazon Redshift AWS リージョンで利用できるわけではありません。この機能が使用可能かどうかは、SHOW コマンドを使用して確認します。AWS Glue の詳細については、AWS Glue デベロッパーガイド の「What is AWS Glue?」を参照してください。

> **Note**
>
> AWS Glue Data Catalog に対するクエリの実行は、Amazon Redshift RG または RA3 のノードタイプのクラスターと Amazon Redshift Serverless でのみサポートされています。

データウェアハウスを設定し、以下の SQL コマンドを使用してカタログ化された AWS Glue データベースのオブジェクトを表示できます。

- SHOW — 現在接続されているデータウェアハウスに awsdatacatalog がマウントされているかを表示します。例えば、data_catalog_auto_mount パラメータ値を表示するには、次を実行します。

```
SHOW data_catalog_auto_mount;
```

詳細については、「Amazon Redshift データベース開発者ガイド」の「SHOW」を参照してください。

- ALTER SYSTEM — data_catalog_auto_mount のシステムレベルの設定を変更します。例えば、data_catalog_auto_mount パラメータ値を on に変更するには、次を実行します。

```
ALTER SYSTEM SET data_catalog_auto_mount = on;
```

この変更は、プロビジョニングされたクラスターが再起動されるか、サーバーレスワークグループが自動的に一時停止して再開されたときに有効になります。詳細については、「Amazon Redshift データベース開発者ガイド」の「ALTER SYSTEM」を参照してください。

- SHOW SCHEMAS — スキーマのリストを表示します。awsdatacatalog という名前のデータベース内のスキーマは、AWS Glue Data Catalogでカタログ化されている AWS Glue データベースを表します。例えば、これらのスキーマを表示するには、次を実行します。

```
SHOW SCHEMAS FROM DATABASE awsdatacatalog;
```

詳細については、「Amazon Redshift データベース開発者ガイド」の「SHOW SCHEMAS」を参照してください。

- SHOW TABLES — スキーマ内のテーブルを一覧表示します。例えば、スキーマ myglue にある、awsdatacatalog という AWS Glue Data Catalogのデータベース内のテーブルを表示するには、次を実行します。

```
SHOW TABLES FROM SCHEMA awsdatacatalog.myschema;
```

詳細については、「Amazon Redshift データベース開発者ガイド」の「SHOW TABLES」を参照してください。

- SHOW COLUMNS – テーブル内の列を一覧表示します。例えば、スキーマ myglue とテーブル mytable にある、awsdatacatalog という AWS Glue Data Catalogのデータベース内の列を表示するには、次を実行します。

```
SHOW COLUMNS FROM TABLE awsdatacatalog.myglue.mytable;
```

詳細については、「Amazon Redshift データベース開発者ガイド」の「SHOW COLUMNS」を参照してください。

IAM ユーザーまたはロールに対し、AWS Glue Data Catalog にクエリを実行する権限を付与するには、

1. ツリービューペインで、[データベースのユーザー名とパスワード] による認証方法を使用して、プロビジョニングされたクラスターまたはサーバーレスワークグループの初期データベースに接続します。例えば、クラスターまたはワークグループの作成時に使用した管理者ユーザーとパスワードを使用して、dev データベースに接続します。
2. エディタタブで以下の SQL ステートメントを実行して、IAM ユーザーに AWS Glue Data Catalog へのアクセス権を付与します。

   ```
   GRANT USAGE ON DATABASE awsdatacatalog to "IAM:myIAMUser"
   ```

   ここで、*IAM:myIAMUser* は、AWS Glue Data Catalog に USAGE 権限を付与したい IAM ユーザーです。または、USAGE 権限を、IAM ロールの *IAMR:myIAMRole* に付与することもできます。
3. ツリービューペインで、以前に作成したクラスターまたはワークグループへの接続を編集または削除します。次のいずれかの方法で、クラスターまたはワークグループに接続します。
   - クラスターから awsdatacatalog データベースにアクセスするには、認証方法として、[IAM ID を使用した一時的な認証情報] を使用する必要があります。認証方法の詳細については、「Amazon Redshift データベースに接続する」を参照してください。この認証方法を接続ウィンドウに表示するには、クエリエディタ v2 の管理者がアカウントの [アカウント設定] を構成することが必要になる場合があります。
   - ワークグループから awsdatacatalog データベースに接続するには、認証方法として [フェデレーションユーザー] を使用する必要があります。認証方法の詳細については、「Amazon Redshift データベースに接続する」を参照してください。
4. 権限が付与されたら、IAM ID を使用して、ご自分の AWS Glue Data Catalog に対して SQL を実行できます。

接続後、クエリエディタ v2 を使用して、AWS Glue Data Catalog でカタログ化されたデータをクエリできます。クエリエディタ v2 のツリービューペインで、クラスターまたはワークグループと awsdatacatalog データベースを選択します。エディタまたはノートブックペインで、正しいクラスターまたはワークグループが選択されていることを確認します。選択するデータベースは、最初の Amazon Redshift データベース (dev など) でなければなりません。クエリの作成については、「Amazon Redshift を使用したクエリの作成」と「Amazon Redshift のノートブック」を参照してください。指定されたデータベース awsdatacatalog は、アカウント内の外部データカタログデータ

ベースを参照するために予約されています。awsdatacatalog データベースに対するクエリは読み取り専用です。SELECT ステートメントでテーブルを参照するには、3 句構成の表記を使用してください。最初の部分はデータベース名、2 番目の部分は AWS Glue データベース名、3 番目の部分は AWS Glue テーブル名です。

```
SELECT * FROM awsdatacatalog.<aws-glue-db-name>.<aws-glue-table-name>;
```

AWS Glue Data Catalog データを読み取って Amazon Redshift テーブルにデータを入力するさまざまなシナリオを実行できます。

次の SQL 例では、AWS Glue で定義されている 2 つのテーブルを結合します。

```
SELECT pn.emp_id, alias, role, project_name
FROM "awsdatacatalog"."empl_db"."project_name_table" pn,
"awsdatacatalog"."empl_db"."project_alias_table" pa
WHERE pn.emp_id = pa.emp_id;
```

次の SQL 例では、Amazon Redshift テーブルを作成し、2 つの AWS Glue テーブルを結合したデータをそのテーブルに入力します。

```
CREATE TABLE dev.public.glue AS
SELECT pn.emp_id, alias, role, project_name
FROM "awsdatacatalog"."empl_db"."project_name_table" pn,
"awsdatacatalog"."empl_db"."project_alias_table" pa
WHERE pn.emp_id = pa.emp_id;
```

Amazon S3 テーブルに対するクエリの実行 (プレビュー)

クエリエディタ v2 を使用して、AWS Glue Data Catalog にマウントされた Amazon S3 テーブルカタログに保持されているデータをクエリできます。Amazon S3 テーブルカタログは、作成時に AWS Glue Data Catalog にマウントされ、同じアカウントの同じ AWS リージョン 内のすべてのプロビジョニング済みクラスターとサーバーレスワークグループで外部データベースとして自動的に表示されます。Amazon Redshift を使用した Amazon S3 テーブルへのアクセスの詳細については、「Amazon Simple Storage Service ユーザーガイド」の「Accessing Amazon S3 tables with Amazon Redshift」を参照してください。

データレイクのクエリ

このチュートリアルの一連のタスクに従って、Amazon S3 データレイク内のデータをクエリできます。まず、AWS Glue Data Catalog内の外部データベースを参照する外部スキーマを作成します。作成後、Amazon S3 データレイク内のデータをクエリすることができます。

デモ: データレイクをクエリする

データレイクをクエリする方法のデモについては、次の動画をご覧ください。Query your data lake from Amazon Redshift query editor v2 (Amazon Redshift クエリエディタ v2 からデータレイクをクエリする)。

前提条件

クエリエディタ v2 でデータレイクの作業をする前に、お使いの Amazon Redshift 環境で以下が設定済みであることを確認してください。

- AWS Glue を使用して Amazon S3 データをクローリングし、AWS Lake Formation のデータカタログを有効にします。
- AWS Lake Formation に対して AWS Glue が有効化されているデータカタログを使用して、Amazon Redshift の IAM ロールを作成します。この手順の詳細については、「AWS Lake Formation に対し有効化されている AWS Glue Data Catalog を使用して、Amazon Redshift の IAM ロールを作成するには」を参照してください。Redshift Spectrum と Lake Formation の詳細については、「AWS Lake Formation で Redshift Spectrum を使用する」を参照してください。
- Lake Formation データベースでクエリを実行するテーブルの SELECT 権限を付与します。この手順の詳細については、「Lake Formation データベースでクエリを実行するテーブルの SELECT 権限を付与するには」を参照してください。

  Lake Formation コンソール (https://console.aws.amazon.com/lakeformation/) の [許可] セクションの [データレイクの許可] ページで、IAM ロール、AWS Glue データベース、テーブルに適切な権限があることを確認できます。
- 接続しているユーザーが、Amazon Redshift データベースにスキーマを作成してデータレイク内のデータにアクセスする権限を持っていることを確認します。クエリエディタ v2 でデータベースに接続するときは、データベースユーザーまたは IAM ユーザーなどの認証情報を含む認証方法を選択します。接続ユーザーには、`superuser` などの適切な権限とデータベース権限が必要です。クラスターまたはワークグループを作成した Amazon Redshift `admin` ユーザーには `superuser` 権限があり、スキーマを作成して Redshift データベースを管理できます。クエリエディタ v2 でデー

タベースに接続する方法の詳細については、「Amazon Redshift データベースに接続する」を参照してください。

外部スキーマの作成

Amazon S3 データレイク内のデータをクエリするには、まず外部スキーマを作成します。外部スキーマは、AWS Glue Data Catalog内の外部データベースを参照します。

1. クエリエディタ v2 の [エディタ] ビュー
   で、
   [作成] を選択して、[スキーマ] を選択します。
2. [スキーマ名] を入力します。
3. [スキーマタイプ] では、[外部] を選択します。
4. [データカタログ] 詳細内の [リージョン] は、デフォルトで Redshift データベースが置かれている AWS リージョン に設定されます。
5. 外部スキーマのマッピング先で、AWS Glue テーブルへの参照を含む AWS Glue データベースを選択します。
6. Amazon S3 上のデータをクエリするために必要な許可を持つ、Amazon Redshift 用 [IAM ロール]を選択します。
7. オプションとして、データカタログへのアクセス許可を持つ IAM ロールを選択します。
8. [スキーマの作成] を選択します。

   ツリービューパネルのデータベースの下にスキーマが表示されます。

スキーマの作成時に、データベースへのアクセスが拒否されたというエラーが表示された場合は、接続しているユーザーがスキーマを作成するためのデータベース権限を持っているかどうかを確認してください。

Amazon S3 データレイク内のデータのクエリ

前の手順で作成したスキーマを使用します。

1. ツリービューパネルで、スキーマを選択します。
2. テーブル定義を表示するには、テーブルを選択します。テーブルの列とデータ型が表示されます。

3. テーブルをクエリするには、テーブルを選択して、右クリックでコンテキストメニューを表示し、[テーブルの選択] をクリックします。
4. [エディタ] でクエリを実行します。

   次の SQL 例は、flightscsv という名前の AWS Glue テーブルのすべての行をクエリするためにクエリエディタ v2 で生成されました。出力の列と行は、単純にするために切り捨てられています。

```
SELECT * FROM "dev"."mydatalake_schema"."flightscsv";

year    quarter   month   dom  day_of_week   fl_date    unique_carrier  airline_id
  carrier   tail_num   fl_num
2016    4         10      19   3             10/19/16   OO              20304
  OO         N753SK    3086
2016    4         10      19   3             10/19/16   OO              20304
  OO         N753SK    3086
2016    4         10      19   3             10/19/16   OO              20304
  OO         N778SK    3087
2016 4          10      19   3             10/19/16   OO              20304
 OO         N778SK    3087
...
```

データ共有

別のクラスターのユーザーがデータをクエリできるように、データ共有を作成できます。共有するデータを含むクラスターは、プロデューサークラスターと呼ばれます。共有するデータベースオブジェクトのデータ共有は、プロデューサークラスター上に作成します。スキーマ、テーブル、ビュー、および SQL ユーザー定義関数 (UDF) を共有できます。データを共有するクラスターは、コンシューマークラスターと呼ばれます。コンシューマークラスターでは、データ共有からデータベースを作成します。その後コンシューマークラスターのユーザーは、データをクエリできます。詳細については、「Amazon Redshift データベースデベロッパーガイド」の「Amazon Redshift でのデータ共有の開始方法」を参照してください。

データ共有の作成

プロデューサークラスターとして使用するクラスター上に、データ共有を作成します。データ共有に関する考慮事項の詳細については、「Data sharing considerations in Amazon Redshift」 (Amazon Redshift データベースデベロッパーガイド) の「Amazon Redshift Database Developer Guide」 (Amazon Redshift でのデータ共有に関する考慮事項) を参照してください。

1. 使用するプロデューサークラスター上のデータベースを選択します。
2. データ共有を作成します。例:

```
create datashare mysource;
```

3. データ共有に対するアクセス許可を設定します。例:

```
grant alter, share on datashare mysource to admin;
```

4. 共有するデータベースオブジェクトに対するアクセス許可を設定します。例:

```
alter datashare mysource add schema public;
```

```
alter datashare mysource add table public.event;
```

5. データ共有にアクセスするためのコンシューマークラスターの名前空間に対するアクセス許可を設定します。例:

```
grant usage on datashare mysource to namespace '2b12345-1234-5678-9012-
bb1234567890';
```

データ共有の表示

プロデューサークラスターで作成したデータ共有を表示できます。

1. プロデューサークラスターを選択します。
2. データ共有を表示します。例:

```
show datashares;
```

```
share_name share_owner source_database  consumer_database share_type createdate
 is_publicaccessible share_acl producer_account producer_namespace
test_datashare 100  db_producer  NULL   OUTBOUND 2/15/2022  FALSE  admin
 123456789012  p1234567-8765-4321-p10987654321
```

コンシューマーデータベースの作成

コンシューマークラスターでは、データ共有からデータベースを作成します。この手順では、同じアカウント内の 2 つのクラスター間でデータを共有する方法について説明します。AWS アカウント間でのデータ共有の詳細については、「Amazon Redshift データベースデベロッパーガイド」の「AWS アカウント間でのデータ共有」を参照してください。

SQL コマンドまたはクエリエディタ v2 のツリービューパネルを使用して、データベースを作成できます。

SQL を使用するには

1. アカウントのデータ共有とプロデューサークラスターの名前空間からデータベースを作成します。例:

```
create database share_db from datashare mysource of account '123456789012'
 namespace 'p1234567-8765-4321-p10987654321';
```

2. ユーザーがデータベースとスキーマにアクセスできるよう、アクセス許可を設定します。例:

```
grant usage on database share_db to usernames;
```

```
grant usage on schema public to usernames;
```

クエリエディタ v2 のツリービューパネルを使用するには

1. 
   [作成] を選択し、次に [データベース] を選択します。
2. [データベース名] を入力します。
3. (任意) ユーザーとグループ を選択し、[データベースユーザー] を選択します。
4. [Create using a datashare] (データ共有を使用して作成) を選択します。
5. データ共有を選択します。
6. [データベースの作成] を選択します。

   新しい
   

[datashare] (データ共有) のデータベースが、クエリエディタ v2 のツリービューパネルに表示されます。

7. ユーザーがデータベースとスキーマにアクセスできるよう、アクセス許可を設定します。例:

```
grant usage on database share_db to usernames;
```

```
grant usage on schema public to usernames;
```

データ共有オブジェクトのクエリ

コンシューマークラスターでは、オブジェクトのデータベース名、スキーマ名、オブジェクト名の 3 つの部分で表記される完全修飾オブジェクト名を使用して、データ共有オブジェクトをクエリできます。

1. クエリエディタ v2 のツリービューパネルで、スキーマを選択します。
2. テーブル定義を表示するには、テーブルを選択します。

   テーブルの列とデータ型が表示されます。
3. テーブルをクエリするには、テーブルを選択し、右クリックでコンテキストメニューを表示して [Select table] (テーブルを選択) をクリックします。
4. SELECT コマンドを使用してテーブルをクエリします。例:

```
select top 10 * from test_db.public.event;
```

クエリエディタ v2 でスケジュールされたクエリ

Amazon Redshift Query Editor V2 を使用すると、SQL クエリを自動化してスケジュールに従って実行させることができます。スケジュール設定されたクエリとは、指定された時間または間隔で自動的に実行される SQL ステートメントであり、定期的に繰り返されるデータオペレーションや分析タスクを効率的に管理できます。バッチ処理の効率化、定期的なレポートの生成、Amazon Redshift 環境内のデータパイプラインのメンテナンスなどを行う場合は、クエリのスケジュールを設定できます。

スケジュールされたクエリを使用すると、抽出、変換、ロード (ETL) ワークフローの自動化、最新のインサイトでのダッシュボードの更新、さまざまなデータ管理ルーチンの運用が容易になります。次のページでは、Amazon Redshift ワークロードを最適化するためにスケジュール設定されたクエリを作成、設定、管理するプロセスについて詳しく説明します。

クエリエディタ v2 を使用したクエリスケジュールの作成

Amazon Redshift クエリエディタ v2 で SQL ステートメントを実行するスケジュールを作成します。ビジネスニーズに合った時間間隔で SQL ステートメントを実行するスケジュールを作成します。スケジュールされたクエリが実行される時間になると、クエリは、Amazon EventBridge によって開始され、Amazon Redshift Data API を使用します。

SQL ステートメントを実行するスケジュールを作成するには

1. [エディタ]

   ビューで

   [スケジュール] を選択して、SQL ステートメントを実行するスケジュールを作成します。
2. スケジュールを定義する場合は、次の情報を指定します。

   - クエリの実行に必要なアクセス許可を持つ IAM ロール。この IAM ロールは、クラスターまたはワークグループにもアタッチされます。
   - クラスターまたはワークグループへのアクセスを承認する AWS Secrets Manager または一時的な認証情報の認証値。これらの認証方法は Data API でサポートされています。詳細については、「スケジュールされたクエリの認証」を参照してください。
   - データベースが存在するクラスターまたはワークグループ。
   - クエリを実行するデータを含むデータベースの名前。
   - スケジュールされたクエリの名前とその説明。クエリエディタ v2 は、指定のスケジュールされたクエリ名の先頭に「QS2-」を付けます。クエリエディタ v1 は、スケジュールされたクエリ名の先頭に「QS-」を付けます。
   - スケジュールで実行される SQL ステートメント。
   - スケジュールの頻度とリピートオプション、またはスケジュールを定義する cron 形式の値。詳細については、Amazon CloudWatch Events ユーザーガイドの Cron 式を参照してください。
   - オプションで、標準の Amazon SNS 通知を有効にして、スケジュールされたクエリをモニタリングできます。場合によっては、Amazon SNS 通知に提供する E メールアドレスを確認する必要があります。Amazon SNS 通知の E メールアドレスを確認するリンクが送信されるため、E メールをチェックしてください。詳細については、Amazon Simple Notification

Service デベロッパーガイドの「E メール通知」を参照してください。クエリが実行されているにもかかわらず、SNS トピックにパブリッシュされているメッセージが表示されない場合は、Amazon EventBridge ユーザーガイドの「ルールは実行されるが、Amazon SNS トピックにいずれのメッセージもパブリッシュされない」を参照してください。

3. [クエリをスケジュール] を選択してスケジュールを保存して有効にし、スケジュールを[スケジュールされたクエリ] ビューのクエリのリストに追加します。

[スケジュールされたクエリ]



ビューには、クラスターとワークグループのすべてのスケジュールされたクエリが一覧表示されます。このビューでは、スケジュールクエリの詳細の表示、スケジュールの有効化または無効化、スケジュールの編集、およびスケジュールされたクエリの削除を行うことができます。クエリの詳細を表示すると、スケジュールとともにクエリを実行した履歴も表示できます。

> **Note**
>
> スケジュールのクエリは、24 時間の [スケジュール履歴] 内でのみ実行できます。スケジュールに従って実行されるクエリは、クエリエディタ v2 の [クエリ履歴] ビューには表示されません。

クエリのスケジュールのデモ

クエリのスケジュールのデモについては、次のビデオをご覧ください。「Video demo of scheduling a query」(クエリのスケジュールのビデオデモ)。

クエリをスケジュールするアクセス許可の設定

クエリをスケジュールするには、スケジュールを定義する AWS Identity and Access Management (IAM) ユーザーとスケジュールに関連付けられている IAM ロールが Amazon EventBridge と Amazon Redshift Data API を使用する IAM アクセス許可で設定されている必要があります。スケジュールされたクエリから E メールを受信するには、オプションで指定する Amazon SNS 通知も設定する必要があります。

以下では、AWS マネージドポリシーを使用してアクセス許可を付与するタスクについて説明しますが、環境によっては、許可されるアクセス許可の範囲の絞り込みが必要な場合があります。

クエリエディタ v2 にログインしている IAM ユーザーの場合は、IAM コンソール (https://console.aws.amazon.com/iam/) を使用して IAM ユーザーを編集します。

- Amazon Redshift とクエリエディタ v2 のオペレーションを実行するアクセス許可に加えて、IAM ユーザーに AmazonEventBridgeFullAccess および AmazonRedshiftDataFullAccess AWS マネージドポリシーをアタッチします。
- または、ロールにアクセス許可を割り当て、そのロールをユーザーに割り当てます。

  スケジュールされたクエリを定義するときに指定する IAM ロールのリソース ARN に sts:AssumeRole アクセス許可を与えるポリシーをアタッチします。ロールの引き受けについての詳細は、IAM ユーザーガイドの「ロールを切り替えるアクセス許可をユーザーに付与する」を参照してください。

  次の例では、アカウント 123456789012 で IAM ロール myRedshiftRole を引き受けるアクセス許可ポリシーを示します。IAM ロール myRedshiftRole は、スケジュールされたクエリを実行するクラスターまたはワークグループにアタッチする IAM ロールでもあります。

  JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AssumeIAMRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": [
                "arn:aws:iam::123456789012:role/myRedshiftRole"
            ]
        }
    ]
}
```

  クエリのスケジュールに使用する IAM ロールの信頼ポリシーを更新して、IAM ユーザーがロールを引き受けられるようにします。

```
{
            "Sid": "AssumeRole",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:user/myIAMusername"
```

```
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

スケジュールされたクエリの実行を許可するように指定した IAM ロールについては、IAM コンソール (https://console.aws.amazon.com/iam/) を使用して IAM ロールを編集します。

- AmazonRedshiftDataFullAccess および AmazonEventBridgeFullAccess AWS マネージドポリシーを IAM ロールにアタッチします。AmazonRedshiftDataFullAccess マネージドポリシーは、キー RedshiftDataFullAccess でタグ付けされた Redshift Serverless ワークグループに対してのみ redshift-serverless:GetCredentials アクセス許可を付与します。

スケジュールされたクエリの認証

クエリをスケジュールする場合は、SQL の実行時に、次のいずれかの認証方法を使用します。各メソッドでは、クエリエディタ v2 の入力の異なる組み合わせが必要です。これらの認証方法は、SQL ステートメントの実行に使用される Data API によってサポートされています。

クエリを実行するために使用されるデータベースユーザーまたはロールには、適切なデータベース権限が必要です。例えば、テーブル mytable に IAMR:MyRedshiftQEv2Scheduler 権限を付与するには、次の SQL コマンドを実行します。

```
GRANT all ON TABLE mytable TO "IAMR:MyRedshiftQEv2Scheduler";
```

クラスターまたはワークグループ内のデータベースユーザーのリストを表示するには、システムビュー PG_USER_INFO にクエリを実行します。

> **Note**
>
> クエリをスケジュールする対象のすべての Redshift Serverless ワークグループに、キー RedshiftDataFullAccess でタグ付けする必要があります。詳細については、「Amazon Redshift Data API へのアクセスの認可」を参照してください。
> ワークグループにタグを付ける代わりに、redshift-serverless:GetCredentials を許可するインラインポリシーを (スケジュールと一緒に指定する) IAMロールに追加できます。例えば、次のようになります。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "UseTemporaryCredentialsForAllServerlessWorkgroups",
            "Effect": "Allow",
            "Action": "redshift-serverless:GetCredentials",
            "Resource": [
                "arn:aws:redshift-serverless:*:*:workgroup/*"
            ]
        }
    ]
}
```

AWS Secrets Manager

この方法を使用して、AWS Secrets Manager に保存されている secret-arn のシークレット値を指定します。このシークレットには、データベースに接続するための認証情報が含まれます。クラスターまたはワークグループの作成時に、適切な認証情報を使用してシークレットを作成したとします。シークレットにはキー `RedshiftDataFullAccess` のタグを付ける必要があります。タグキーがない場合は、AWS Secrets Manager コンソールを使用して追加します。シークレットの作成方法の詳細については、「データベース接続認証情報のシークレットの作成」を参照してください。

最小のアクセス許可についての詳細は、AWS Secrets Manager ユーザーガイドの「AWS Secrets Manager を使用したシークレットの作成と管理」を参照してください。

一時的な認証情報

このメソッドでは、クラスター内のデータベースに接続するときに、データベース名とデータベースユーザーの値を指定します。データベース名は、ワークグループ内のデータベースに接続するときだけに指定する必要があります。

クラスターに接続する場合、`AmazonRedshiftDataFullAccess` ポリシーは、`redshift_data_api_user` という名前のデータベースユーザーに `redshift:GetClusterCredentials` へのアクセス許可を付与します。別のデータベース

ユーザーを使用して SQL ステートメントを実行する場合は、クラスターにアタッチされた IAM ロールにポリシーを追加して redshift:GetClusterCredentials を許可します。次のポリシー例では、データベースユーザー awsuser と myuser を許可しています。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "UseTemporaryCredentialsForAllDbUsers",
            "Effect": "Allow",
            "Action": "redshift:GetClusterCredentials",
            "Resource": [
                "arn:aws:redshift:*:*:dbuser:*/awsuser",
                "arn:aws:redshift:*:*:dbuser:*/myuser"
            ]
        }
    ]
}
```

スケジュールクエリ履歴を表示するアクセス許可の設定

ユーザーにスケジュールクエリ履歴の表示を許可するには、スケジュールと一緒に指定する IAM ロールの信頼関係を編集してアクセス許可を追加します。

次に示すのは、IAM ユーザー *myIAMusername* にスケジュールクエリの履歴を表示できるようにする IAM ロールの信頼ポリシーの例です。IAM ユーザーに sts:AssumeRole アクセス許可を付与する代わりに、このアクセス許可を IAM ロールに付与するように選択できます。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "redshift.amazonaws.com",
```

```
                    "redshift-serverless.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        },
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "events.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        },
        {
            "Sid": "AssumeRole",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:user/myIAMusername"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

スケジュールされたクエリのモニタリング

E メール通知を送信するように指定した Amazon SNS トピックについては、クエリエディタ v2 を使用して Amazon SNS トピックを作成します。これを行うには、[SNS 通知] セクションに移動し、モニタリングを [オンにする] を選択して、[SNS トピックの作成] でトピックを作成します。クエリエディタ v2 は Amazon SNS トピックを作成し、Amazon EventBridge のアクセスポリシーにサービスプリンシパルを追加します。次に示すのは、Amazon SNS トピックで作成したアクセスポリシーの例です。この例では、AWS リージョン *us-west-2*、AWS アカウント *123456789012*、および Amazon SNS トピック *select-version-pdx-testunload* を使用しています。

JSON

```
{
  "Version":"2012-10-17",
  "Id": "__default_policy_ID",
  "Statement": [
    {
```

```
      "Sid": "Allow_Publish_Events",
      "Effect": "Allow",
      "Principal": {
        "Service": "events.amazonaws.com"
      },
      "Action": "sns:Publish",
      "Resource": "arn:aws:sns:us-west-2:123456789012:select-version-pdx-
testunload"
    }
  ]
}
```

スケジュールされたクエリが実行されると、Amazon SNS は AWS 通知 E メールを送信します。次は、Amazon SNS 通知トピック *may25a-SNS* を使って AWS アカウント *123456789012* の AWS リージョン *eu-north-1* で実行された、スケジュールされたクエリ *QS2-may25a* について、*myemail@example.com* に送信された E メールの例です。

```
{"version":"0","id":"8e4323ec-5258-7138-181b-91290e30ff9b","detail-type":"Scheduled
 Event","source":"aws.events","account":"123456789012","time":"2023-05-25T15:22:00Z",
                    "region":"eu-north-1","resources":["arn:aws:events:eu-
north-1:123456789012:rule/QS2-may25a"],"detail":{}}

--
If you wish to stop receiving notifications from this topic, please click or visit the
 link below to unsubscribe:
https://sns.eu-north-1.amazonaws.com/unsubscribe.html?SubscriptionArn=arn:aws:sns:eu-
north-1:123456789012:may25a-SNS:0c1a3d05-39c2-4507-
bc3d-47250513d7b0&Endpoint=myemail@example.com

Please do not reply directly to this email. If you have any questions or comments
 regarding this email, please contact us at https://aws.amazon.com/support
```

クエリのスケジュールのセットアップに関するトラブルシューティング

クエリのスケジュールに問題がある場合は、次の点を考慮してください。

クエリが実行していない

スケジュールで使用している IAM ロールに、一時的なクラスター認証情報を取得するアクセス許可があるかどうかを確認します。プロビジョニングされたクラスターのアクセス許可

は `redshift:GetClusterCredentialsWithIAM` です。Redshift Serverless ワークグループのアクセス許可は `redshift-serverless:GetCredentials` です。

スケジュールされた履歴が表示されない

AWS コンソールへのログインに使用する IAM ユーザーまたは IAM ロールが、クエリのスケジュールに使用された IAM ロールの信頼ポリシーに追加されていませんでした。

スケジュールされたクエリに AWS Secrets Manager を使用し、接続する場合は、シークレットがキー `RedshiftDataFullAccess` でタグ付けされていることを確認します。

スケジュールされたクエリが AWS Secrets Manager 接続を使用している場合、クエリのスケジュールに使用される IAM ロールには、`SecretsManagerReadWrite` 管理ポリシーと同等のものがロールにアタッチされている必要があります。

クエリ履歴のステータスが **`Failed`** である

クエリが失敗した理由の詳細については、SYS_QUERY_HISTORY システムビューを参照してください。よくある問題は、クエリの実行に使用したデータベースユーザーまたはロールに SQL の実行に必要な権限がないことです。詳細については、「スケジュールされたクエリの認証」を参照してください。

次の SQL は、SYS_QUERY_HISTORY ビューにクエリを実行して、失敗したクエリを返します。

```
SELECT user_id, query_id, transaction_id, session_id, database_name, query_type,
 status, error_message, query_text
FROM sys_query_history
WHERE status = 'failed';
```

失敗した特定のスケジュールされたクエリの詳細を確認するには、「スケジュールされたクエリの結果を AWS CloudShell で表示」を参照してください。

スケジュールされたクエリの結果を AWS CloudShell で表示

AWS CloudShell を使用してスケジュールクエリの詳細を確認できます。次の手順に示す、AWS CLI コマンドを実行するための適切なアクセス許可を持っている必要があります。

スケジュールされたクエリの結果を表示するには

1. AWS コンソールで、AWS CloudShell コマンドプロンプトを開きます。AWS CloudShell の詳細については、「AWS CloudShell ユーザーガイド」の「AWS CloudShell とは」を参照してください。
2. スケジュールされたクエリの IAM ロールを引き受けます。ロールを引き受けるには、スケジュールされたクエリに関連付けられた IAM ロールをクエリエディタ v2 で見つけて、AWS CLI コマンドを AWS CloudShell で使用します。例えば、ロール `scheduler` の場合は、AWS STS コマンドを入力して、スケジュールされたクエリで使用されているロールを引き受けます。

   ```
   aws sts assume-role --role-arn "arn:aws:iam::123456789012:role/scheduler" --role-
   session-name "scheduler-test"
   ```

   返される認証情報は次のようになります。

   ```
   "Credentials": {
   "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
   "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
   "SessionToken": "je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY...",
   "Expiration": "2023-08-18T18:19:44+00:00"
   },
   "AssumedRoleUser": {
   "AssumedRoleId": "AROA35B2NH6WBTP7ONL4E:scheduler-test",
   "Arn": "arn:aws:sts::123456789012:assumed-role/scheduler/scheduler-test"
   }
   }
   ```
3. IAM ロールを引き受けたときに表示される認証情報を使用して、AWS CLI で環境変数を作成します。これらのトークンは、有効期限が切れる前に使用する必要があります。例えば、AWS CloudShell に次の内容を入力します。

   ```
   export AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
   export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
   export AWS_SESSION_TOKEN=je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY...
   ```
4. 失敗したクエリのエラーを表示するには、ステートメントを記述する AWS CLI コマンドを実行します。SQL ステートメントの ID は、クエリエディタ v2 でスケジュールされたクエリの [スケジュール履歴] セクションに表示される [ID] からのものです。

```
aws redshift-data describe-statement --id 130d2620-05d2-439c-b7cf-815d9767f513
```

この例の場合、スケジュールされた SQL `select * from users limit 100` は、`users` テーブルが存在しないという SQL エラーになります。

```
{
"CreatedAt": "2023-08-18T17:39:15.563000+00:00",
"Duration": -1,
"Error": "ERROR: relation \"users\" does not exist",
"HasResultSet": false,
"Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
"QueryString": "select * from users limit 100\n—RequestID=a1b2c3d4-5678-90ab-cdef-
EXAMPLE22222; TraceID=1-633c5642-4039308d03f3a0ba53dbdf6f",
"RedshiftPid": 1073766651,
"RedshiftQueryId": 0,
"ResultRows": -1,
"ResultSize": -1,
"Status": "FAILED",
"UpdatedAt": "2023-08-18T17:39:16.116000+00:00",
"WorkgroupName": "default"
}
```

クエリ結果の視覚化

クエリを実行して結果が表示された後 [Chart] (チャート) をオンにすると、結果をグラフィックで視覚化して表示することができます。以下のコントロールを使用して、グラフの内容、構造、および外観を定義できます。

+

トレース

グラフ内の関連するグラフィックマークのセットを表します。グラフには複数のトレースを定義できます。

タイプ

トレースタイプを定義して、データを以下のいずれかとして表すことができます。

- 散布プロットまたはバブルチャートのための散布図。
- 垂直または水平の棒でデータのカテゴリを表す棒グラフ。

- 塗りつぶされた領域を定義する面グラフ。
- 棒を使用して度数分布を表すヒストグラム。
- 各スライスが全体の割合を表す、データの円形表現のための円グラフ。
- プロセスのさまざまな段階のデータを表すファネルグラフまたはファネル面グラフ。
- X 軸 (通常、時間間隔を表す) に沿って始値、高値、安値、および終値を表す財務データによく使用される OHLC (open-high-low-close) グラフ。
- 時系列に沿ってカテゴリの値の範囲を表すローソク足チャート。
- 一連の中間値を介して初期値がどのように増減するかを表すウォーターフォールグラフ。値は、時間間隔またはカテゴリを表すことができます。
- 経時的な値の変化を表す折れ線グラフ。

X 軸

X 軸に沿ってプロットする値が含まれるテーブル列を指定します。記述的な値が含まれる列は、多くの場合、ディメンションデータを表します。定量的な値が含まれる列は、多くの場合、事実データを表します。

Y 軸

Y 軸に沿ってプロットする値が含まれるテーブル列を指定します。記述的な値が含まれる列は、多くの場合、ディメンションデータを表します。定量的な値が含まれる列は、多くの場合、事実データを表します。

サブプロット

グラフデータの追加の表現を定義できます。

変換

変換を定義して、トレースデータをフィルタリングすることができます。分割変換は、単一のソーストレースから複数のトレースを表示するために使用します。集計変換は、トレースを平均値または最小値として表示するために使用します。ソート変換は、トレースをソートするために使用します。

全体的な外観

背景色、余白の色、パレットをデザインするためのカラースケール、テキストのスタイルとサイズ、タイトルのスタイルとサイズ、およびモードバーのデフォルトを設定できます。ドラッグ、クリック、およびホバーの操作を定義できます。メタテキストを定義できます。トレース、軸、凡例、および注釈のデフォルト外観を定義できます。

チャートを作成するには

1. クエリを実行し、結果を取得します。
2. [チャート] をオンにします。
3. [トレース] を選択し、データの視覚化を開始します。
4. チャートスタイルは次のいずれかを選択します。
   - 散布図
   - 棒グラフ
   - 面グラフ
   - ヒストグラム
   - 円グラフ
   - ファネルグラフ
   - ファネル面グラフ
   - OHLC (open-high-low-close) グラフ
   - ローソク足チャート
   - ウォーターフォールグラフ
   - 線グラフ
5. [スタイル] をクリックして、色、軸、凡例、注釈などの外観をカスタマイズします。テキスト、図形、画像を追加できます。
6. [注釈] をクリックして、テキスト、図形、画像を追加します。
7. チャートの表示を更新するには [更新] をクリックします。[全画面表示] を選択すると、チャート表示が展開します。

例: 円グラフを作成してクエリ結果を視覚化する

次の例では、サンプルデータベースの 売上 テーブルを使用します。詳細については、「Amazon Redshift データベースデベロッパーガイド」の「サンプルデータベース」を参照してください。

以下は、円グラフのデータを提供するために実行するクエリです。

```
select top 5 eventname, count(salesid) totalorders, sum(pricepaid) totalsales
from sales, event
where sales.eventid=event.eventid group by eventname
```

```
order by 3;
```

売上合計で上位のイベントの円グラフを作成するには

1. クエリを実行します。
2. クエリ結果領域で [チャート] をオンにします。
3. [トレース] を選択します。
4. [タイプ] で、[円グラフ] を選択します。
5. [値] で [totalsales] を選択します。
6. [ラベル] で、[eventname] を選択します。
7. [スタイル] の次に [一般] を選択します。
8. [Colorscales] (カラースケール) で [Categorical]、[Pastel2] を選択します。



例: 収益と売上を比較するための複合グラフを作成する

この例の手順を実行して、収益データの棒グラフと売上データの折れ線グラフを組み合わせたグラフを作成します。以下の例は、tickit サンプルデータベースの Sales テーブルを使用します。詳細については、「Amazon Redshift データベースデベロッパーガイド」の「サンプルデータベース」を参照してください。

以下は、グラフのデータを提供するために実行するクエリです。

```
select eventname, total_price, total_qty_sold
from  (select eventid, total_price, total_qty_sold, ntile(1000) over(order by
 total_price desc) as percentile
```

```
       from (select eventid, sum(pricepaid) total_price, sum(qtysold) total_qty_sold
             from   tickit.sales
             group by eventid)) Q, tickit.event E
       where Q.eventid = E.eventid
       and percentile = 1
order by total_price desc;
```

収益と売上を比較するための複合グラフを作成する

1. クエリを実行します。
2. クエリ結果領域で [チャート] をオンにします。
3. [トレース o] で、[タイプ] に [Bar] (棒グラフ) を選択します。
4. [X] には eventname を選択します。
5. [Y] には total_price を選択します。

   棒グラフの X 軸に沿ってイベント名が表示されます。
6. [Style] (スタイル) で [Traces] (トレース) を選択します。
7. [Name] (名前) に Revenue と入力します。
8. [Style] (スタイル) で、[Axes] (軸) を選択します。
9. [Titles] (タイトル) には、[Y] を選択して Revenue と入力します。

   左側の Y 軸に、ラベル Revenue が表示されます。
10. [Structure] (構造) で [Trace] (トレース) を選択します。
11. ＋

    [Trace] (トレース) を選択します。

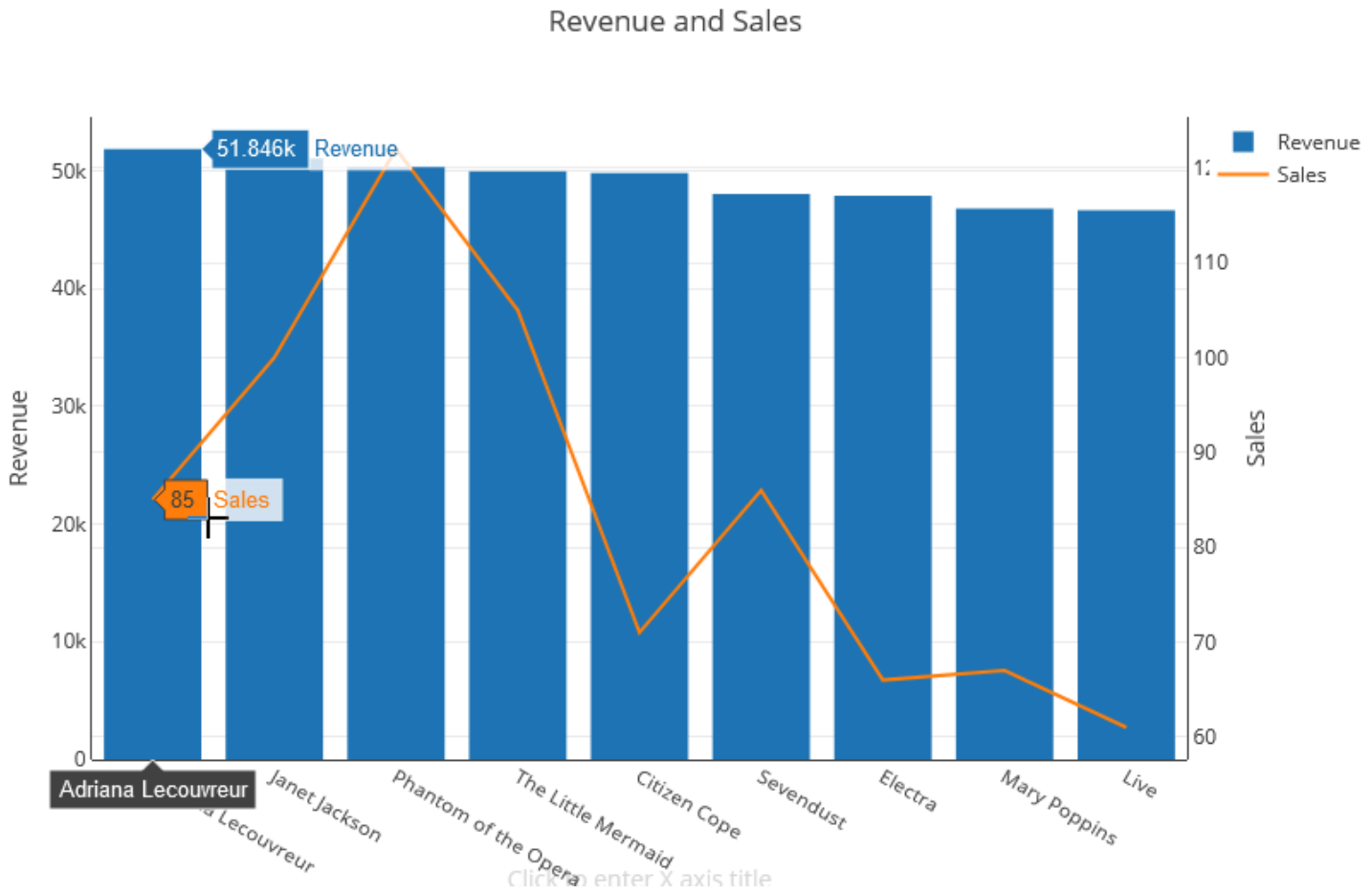
    トレース 1 のオプションが表示されます。
12. [タイプ] には [Line] (線グラフ) を選択します。
13. [X] には eventname を選択します。
14. [Y] には total_qty_sold を選択します。
15. [Axes To Use] (使用する軸) で、[Y Axis] (Y 軸) に

    ＋

    を選択します。

    [Y Axis] (Y 軸) が Y2 を表示します。

16. [Style] (スタイル) で、[Axes] (軸) を選択します。
17. [TITLE] (タイトル) で [Y2] を選択します。
18. [Name] (名前) に Sales と入力します。
19. [LINES] (線) で、Y:Sales を選択します。
20. [Axis Line] (軸線) で [Show] (表示) を選択し、[Position] (位置) に [Right] (右) を選択します。



デモ: Amazon Redshift クエリエディタ v2 を使用してビジュアライゼーションを構築する

ビジュアライゼーションの構築方法のデモについては、次の動画をご覧ください。Build visualizations using Amazon Redshift query editor v2 (Amazon Redshift クエリエディタ v2 を使用してビジュアライゼーションを構築する)。

チームとしての共同作業と共有

クエリをチームと共有できます。

チームは、クエリエディタ v2 リソースを共同作業し、共有する一連のユーザーに対して定義されます。管理者は IAM ロールにタグを追加することでチームを作成できます。詳細については、「クエリエディタv2 を使用するために必要なアクセス許可」を参照してください。

クエリの保存と閲覧

クエリをチームと共有する前に、クエリを保存してください。保存されたクエリを表示および削除できます。

クエリを保存するには

1. クエリを準備し [保存] を選択します。
2. クエリのタイトルを入力します。
3. [保存] を選択します。

保存されたクエリを閲覧するには

1. ナビゲーションペインから [クエリ] を選択します。
2. [マイクエリ]、[Shared by me] (私が共有したクエリ) または [Shared to my team] (チームで共有したクエリ) を表示することができます。これらのクエリは、個別のクエリとして表示することも、作成したフォルダ内に表示することもできます。

クエリの共有

チーム内でクエリを共有できます。保存されたクエリの履歴の表示や、クエリのバージョンを管理することもできます。

クエリをチームと共有するには、プリンシパルタグ `sqlworkbench-team`をアカウントの残りのチームメンバーと同じ値に設定します。たとえば、管理者は `accounting-team` の値を会計部門のすべての人のために設定します。例については、「クエリエディタv2 を使用するために必要なアクセス許可」を参照してください。

チームとクエリを共有するには

1. ナビゲーションペインから [クエリ] を選択します。
2. 共有したいクエリを右クリックしてコンテキストメニューを開き、[チームと共有] を選択します。
3. クエリを共有する 1 つまたは複数のチームを選択し、[共有オプションの保存] を選択します。

クエリのバージョン管理

SQL クエリを保存するたびに、クエリエディタ v2 は新しいバージョンとして保存します。以前のバージョンのクエリの閲覧、クエリのコピーと保存、クエリの復元ができます。

クエリバージョンを管理するには

1. ナビゲーションペインから [クエリ] を選択します。
2. 操作したいクエリを右クリックして、コンテキストメニューを開きます。
3. [バージョン履歴] を選択して、クエリのバージョンの一覧を開きます。
4. [バージョン履歴] ページでは、以下のことが実行できます。
   - [Revert to selected] (選択済みに戻す) — 選択したバージョンに戻して、このバージョンで作業を続行します。
   - [Save selected as] (選択済みを名前を付けて保存) — エディタで新しいクエリを作成します。

Amazon Redshift クエリエディタ v1 を使用したデータベースのクエリの実行

Amazon Redshift クラスターによってホストされているデータベースでクエリを実行するには、クエリエディタを使用することが最も簡単な方法です。クラスターを作成した後、Amazon Redshift コンソールでクエリエディタを使用することで、すぐにクエリを実行できます。

> **Note**
>
> Amazon Redshift サーバーレスでは、この元のクエリエディタを使用してデータをクエリすることはできません。Amazon Redshift クエリエディタ v2 を代わりに使用してください。

2021 年 2 月、更新されたクエリエディターがデプロイされ、クエリエディターを使用する権限が変更されました。新しいクエリエディターは、Amazon Redshift データ API を使用してクエリを実行します。AWS マネージド AWS Identity and Access Management (IAM) ポリシーである `AmazonRedshiftQueryEditor` ポリシーが更新され、必要なアクセス許可が含まれるようになりました。カスタム IAM ポリシーがある場合は、必ず更新してください。`AmazonRedshiftQueryEditor` をガイドとして使用してください。`AmazonRedshiftQueryEditor` への変更には、以下が含まれます。

- クエリエディタのステートメント結果を管理する権限には、ステートメント所有者ユーザーが必要です。
- Secrets Manager を使用してデータベースに接続する権限が追加されました。

詳細については、「Amazon Redshift コンソールのクエリエディタを使用するために必要なアクセス許可」を参照してください。

新しいクエリエディターからクラスターに接続する場合、次の 2 つの認証方法のいずれかを使用できます。

クエリエディタを使用して、以下の操作を実行できます。

- 単一の SQL ステートメントクエリを実行する。
- 最大 100 MB の結果セットをカンマ区切り値 (CSV) ファイルにダウンロードする。
- 再利用するためにクエリを保存する。ヨーロッパ (パリ) リージョン、アジアパシフィック (大阪) リージョン、アジアパシフィック (香港) リージョン、中東 (バーレーン) リージョンでは、クエリを保存できません。
- ユーザー定義テーブルのクエリ実行の詳細を表示する。
- 将来の時間に実行するようにクエリをスケジュールします。
- クエリエディタで作成したクエリの履歴を表示します。
- 拡張 VPC ルーティングを使用してクラスターに対してクエリを実行します。

クエリエディタの考慮事項

クエリエディタを使用するときは、クエリを操作する方法について次の操作を行います。

- クエリの最大期間は 24 時間です。
- クエリ結果の最大サイズは 100 MB です。100 MB を超えるレスポンスデータが返されると、その呼び出しは終了します。
- クエリ結果の最大保持時間は 24 時間です。
- クエリステートメントの最大サイズは 100 KB です。
- クラスターは、Amazon VPC サービスに基づいて Virtual Private Cloud (VPC) で作成する必要があります。
- クエリエディタでトランザクションを使用することはできません。トランザクションの詳細については、Amazon Redshift データベースデベロッパーガイドの「BEGIN」を参照してください。

- クエリは最大 3,000 文字保存することができます。

SQL クライアントツールを使用して Amazon Redshift データウェアハウスクラスターに接続する

Java Database Connectivity (JDBC)、Python、および Open Database Connectivity (ODBC) 接続を介して、SQL クライアントツールから Amazon Redshift データウェアハウスに接続できます。Amazon Redshift では、SQL クライアントツールまたはライブラリを提供またはインストールしません。これらのツールまたはライブラリを使用してデータウェアハウス内のデータを操作するには、それらをクライアントコンピュータまたは Amazon EC2 インスタンスにインストールします。JDBC、Python、または ODBC ドライバーをサポートするほとんどの SQL クライアントツールが使用可能です。

このトピックの最後にあるセクションのリストでは、JDBC、Python、または ODBC による接続を使用するために、ご自分のクライアントコンピュータまたは Amazon EC2 インスタンスを設定するプロセスを順を追って説明します。また、サーバーへのクライアント接続に関連するセキュリティオプションについても説明します。さらに、Amazon Redshift RSQL などの SQL クライアントツールの設定と接続に関する情報も以下に記載されています。ビジネスインテリジェンスツールをまだ使用していない場合は、これらのツールを試すことができます。また、このセクションでは、データに接続する方法を理解することもできます。最後に、データウェアハウスに接続しようとして問題が発生した場合は、トラブルシューティング情報を参照して解決策を特定できます。

クライアントツールとの接続に関する推奨事項

IP アドレスを使用して Redshift クラスターに接続する場合、停止または接続が切断され、クラスターが新しいアベイラビリティーゾーン (AZ) でオンラインになる際に、別のダウンタイムが発生する可能性があります。それでも IP アドレスを使用してアプリケーションを Redshift に接続する場合は、クラスターの仮想プライベートクラウド (VPC) エンドポイントにアタッチされたプライベート IP アドレスを使用してください。これは、[プロパティ] タブの [ネットワークとセキュリティ] のクラスターの詳細に表示されます。

> **Note**
>
> アプリケーションがリーダーノードの IP アドレスを使用して Redshift クラスターにアクセスする場合、推奨されるベストプラクティスは、クラスターエンドポイント URL を使用す

るように変更することです。詳細については、「Amazon Redshift での接続の設定」を参照してください。

トピック

- Amazon Redshift での接続の設定
- メタデータ検出のベストプラクティス
- 接続のセキュリティオプションを設定する
- クライアントツールおよびコードからの接続
- 認証プロファイルを使用して Amazon Redshift に接続する
- Amazon Redshift での接続の問題のトラブルシューティング

Amazon Redshift での接続の設定

次のセクションでは、SQL クライアントツールからクラスターに接続するために、JDBC、Python、および ODBC での接続を設定する方法について説明します。このセクションでは、JDBC、Python、ODBC での接続を設定する方法について説明します。また、Secure Sockets Layer (SSL) とサーバー証明書を使用して、クライアントとサーバー間の通信を暗号化する方法についても説明します。

Amazon Redshift 用の JDBC、Python、および ODBC ドライバー

クラスター内のデータを操作するには、クライアントコンピュータまたはインスタンスから接続できるようにするために、JDBC、Python、または ODBC のドライバーが必要です。JDBC、Python、もしくは ODBC サポートする SQL クライアントツールを使用するために、JDBC、Python、ODBC のデータアクセス API オペレーションのいずれかを使用するように、アプリケーションを記述します。

Amazon Redshift では JDBC、Python、および ODBC ドライバーをダウンロードできます。これらのドライバーは サポート によってサポートされています。PostgreSQL ドライバーはテストされておらず、Amazon Redshift チームによってサポートされていません。Amazon Redshift クラスターに接続するときは、Amazon Redshift 固有のドライバーを使用してください。Amazon Redshift ドライバーには以下の利点があります。

- IAM、SSO、およびフェデレーション認証のサポート。
- 新しい Amazon Redshift データ型のサポート。

- 認証プロファイルのサポート。
- Amazon Redshift の機能強化に伴うパフォーマンスの向上。

JDBC および ODBC ドライバーをダウンロードし、クラスターへの接続を設定する方法の詳細については、Amazon Redshift での JDBC ドライバーバージョン 2.x による接続の構成、Amazon Redshift Python コネクタおよび Amazon Redshift での ODBC ドライバーのバージョン 2.x による接続の設定 を参照してください。

IAM ロールのベストプラクティスを含む IAM ID の管理の詳細については、「Amazon Redshift での Identity and Access Management」を参照してください。

クラスター接続文字列を検索する

SQL クライアントツールでクラスターに接続するには、クラスター接続文字列が必要です。クラスター接続文字列は、Amazon Redshift コンソールのクラスターの詳細ページにあります。

クラスターの接続文字列を見つけるには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [クラスター] を選択し、リストからクラスター名を選択してその詳細を開きます。
3. JDBC URL および ODBC URL 接続文字列は、追加の詳細とともに、[General information (一般情報)] セクションで手に入れることができます。各文字列はクラスターが実行される AWS リージョンに基づき決まります。適切な接続文字列の横にあるアイコンをクリックしてコピーします。

クラスターエンドポイントに接続するには、DescribeClusters API リクエストのクラスターエンドポイント URL を使用できます。クラスターエンドポイント URL の例は次のとおりです。

```
mycluster.cmeaswqeuae.us-east-2.redshift.amazonaws.com
```

クラスターのカスタムドメイン名を設定している場合は、それを使用してクラスターに接続することもできます。カスタムドメイン名の作成の詳細については、「カスタムドメイン名の設定」を参照してください。

> **Note**
>
> 接続するときは、クラスターノードの IP アドレスや VPC エンドポイントの IP アドレスを使用しないでください。不必要な停止を避けるため、必ず Redshift エンドポイントを使用してください。例外として、カスタムドメイン名を使用する場合に限り、エンドポイント URL を使用しません。詳細については、「クライアント接続にカスタムドメイン名を使用する」を参照してください。

Amazon Redshift での JDBC ドライバーバージョン 2.x による接続の構成

JDBC ドライバーバージョン 2.x による接続を使用すると、多くのサードパーティー SQL クライアントツールから Amazon Redshift クラスターに接続できます。Amazon Redshift JDBC コネクタは、オープンソースのソリューションを提供します。ソースコードの閲覧、機能拡張のリクエスト、問題の報告、およびコントリビューションを行うことができます。

JDBC ドライバーの変更に関する最新情報については、「変更ログ」を参照してください。

デフォルトでは、Amazon Redshift JDBC ドライバーは TCP キープアライブを使用して接続がタイムアウトしないように設定されています。ドライバーがキープアライブパケットの送信を開始するタイミングを指定することや、この機能を無効にするために、接続 URL 内に関連するプロパティを設定することができます。接続 URL の構文の詳細については、接続 URL の構築を参照してください。

プロパティ	説明
`TCPKeepAlive`	TCP キープアライブをオフにするには、このプロパティを `FALSE` に設定します。

トピック

- Amazon Redshift JDBC ドライバーのバージョン 2.x をダウンロードする
- Amazon Redshift JDBC ドライバーバージョン 2.x のインストール
- JDBC URL の取得
- 接続 URL の構築
- Apache Maven を使用して JDBC 接続を設定する
- 認証と SSL の設定

- ログ記録の構成
- データ型変換
- プリペアドステートメントのサポートを使用する
- JDBC ドライバーの 2.x と 1.x バージョンの違い
- JDBC ドライバーバージョン 2.x 用の初期化 (.ini) ファイルの作成
- JDBC ドライバーバージョン 2.x の設定オプション
- JDBC ドライバーバージョン 2.x の以前のバージョン

Amazon Redshift JDBC ドライバーのバージョン 2.x をダウンロードする

> **Note**
>
> Amazon Redshift JDBC 2.x ドライバーはスレッドセーフな設計ではありません。同じ接続を同時に使用しようとするスレッドが 2 つ以上あると、デッドロック、エラー、誤った結果、またはその他の予期しない動作が発生する可能性があります。
> マルチスレッドアプリケーションを使用している場合は、同時アクセスを避けるために、ドライバーへのアクセスを同期することをお勧めします。

Amazon Redshift には、JDBC 4.2 API と互換性のあるツール用のドライバーが用意されています。このドライバーのクラス名は `com.amazon.redshift.Driver` です。

JDCBC ドライバーをインストールする方法、JDBC ドライバーライブラリを参照する方法、ドライバークラスを登録する方法の詳細については、以下のトピックを参照してください。

Amazon Redshift JDBC ドライバーバージョン 2.x を使用するコンピュータごとに、Java ランタイム環境 (JRE) 8.0 がインストールされていることを確認します。

データベース認証用に Amazon Redshift JDBC ドライバーを使用する場合、Java クラスパスに AWS SDK for Java 2.40.5 以降があることを確認してください。AWS SDK for Java をインストールしていない場合は、JDBC 4.2 互換のドライバーならびに AWS SDK 用のドライバー依存ライブラリが格納された、ZIP ファイルをダウンロードします。

- JDBC 4.2 互換ドライバーバージョン 2.x および AWS SDK ドライバー依存ライブラリ

  この ZIP ファイルには、JDBC 4.2 互換ドライバーバージョン 2.xと、AWS SDK for Java 2.x のドライバー依存ライブラリファイルが含まれています。依存 jar ファイルを JDBC ドライバーと同じ場所に解凍します。CLASSPATH に含める必要があるのは、JDBC ドライバのみです。

この ZIP ファイルには、AWSSDK for Java 2.x のすべては集録されていません。ただしこれには、AWS Identity and Access Management(IAM) データベース認証に必要な、AWSSDK for Java 2.x のドライバ依存ライブラリが含まれています。

IAM データベース認証に必要な AWS SDK では、この Amazon Redshift JDBC ドライバーを使用します。

AWS SDK for Java 2.x の完全版をインストールするには、AWS SDK for Java デベロッパーガイドの「AWS SDK for Java 2.x」を参照してください。

- JDBC 4.2 互換ドライバーバージョン 2.x (AWS SDK なし)

JDBC ドライバーバージョン 2.x の、ソフトウェアライセンスと変更ログファイルを確認します。

- JDBC ドライバーバージョン 2.x ライセンス
- JDBC ドライバーバージョン 2.x 変更ログ

JDBC ドライバーバージョン 1.2.27.1051 以降では Amazon Redshift のストアドプロシージャをサポートしています。詳細については、Amazon Redshift データベースデベロッパーガイドの Amazon Redshift でのストアドプロシージャの作成を参照してください。

Amazon Redshift JDBC ドライバーバージョン 2.x のインストール

AWS SDK 用の、Amazon Redshift JDBC 4.2 互換ドライバーバージョン 2.x とドライバー依存ライブラリをインストールするには、ZIP アーカイブからファイルを抽出し任意のディレクトリに格納します。

Amazon Redshift JDBC 4.2 互換ドライバーバージョン 2.x を (AWS SDKを含めずに) インストールするには、JAR ファイルを任意のディレクトリにコピーします。

Amazon Redshift JDBC ドライバーを使用して Amazon Redshift データストアにアクセスするには、以下に説明するように設定を実行する必要があります。

トピック

- JDBC ドライバーライブラリの参照
- ドライバークラスの登録

JDBC ドライバーライブラリの参照

データへの接続に使用する JDBC アプリケーションまたは Java コードは、ドライバー JAR ファイルにアクセスする必要があります。アプリケーションまたはコードで、ZIP アーカイブから抽出したすべての JAR ファイルを指定します。

JDBC アプリケーションでのドライバーの使用

JDBC アプリケーションは、通常、ドライバーライブラリファイルのリストを追加するための一連の設定オプションを提供します。提供されたオプションを使用して、ZIP アーカイブのすべての JAR ファイルをアプリケーションのドライバー設定の一部として含めます。詳細については、JDBC アプリケーションのドキュメントを参照してください。

Java コードでのドライバーの使用

すべてのドライバーライブラリファイルをクラスパスに含める必要があります。これは、Java ランタイム環境がクラスおよびその他のリソースファイルを検索するパスです。オペレーティングシステムでクラスパス設定をする場合は、適切な Java SE ドキュメントを参照し詳細を確認してください。

- Windows の場合: https://docs.oracle.com/javase/7/docs/technotes/tools/windows/classpath.html
- Linux および Solaris: https://docs.oracle.com/javase/7/docs/technotes/tools/solaris/classpath.html
- macOS: デフォルトの macOS クラスパスは、JDBC ドライバーがインストールされているディレクトリです。

ドライバークラスの登録

アプリケーションに適したクラスを登録していることを確認してください。Amazon Redshift JDBC ドライバーを Amazon Redshift データストアに接続するには、次のクラスを使用します。

- `Driver` クラスは を拡張します。`java.sql.Driver`
- `DataSource` クラスは `javax.sql.DataSource` と `javax.sql.ConnectionPoolDataSource` を拡張します。

ドライバーは、JDBC バージョンに依存しない次の完全修飾クラス名をサポートします。

- `com.amazon.redshift.jdbc.Driver`
- `com.amazon.redshift.jdbc.DataSource`

次の例は、DriverManager クラスを使用して JDBC 4.2 用の接続を確立する方法を示しています。

```
            private static Connection connectViaDM() throws Exception
{
Connection connection = null;
connection = DriverManager.getConnection(CONNECTION_URL);
return connection;
}
```

次の例では、DataSourceクラスを使用して接続を確立する方法を示しています。

```
 private static Connection connectViaDS() throws Exception
{
Connection connection = null;
11
Amazon Redshift JDBC Driver Installation and Configuration Guide
DataSource ds = new com.amazon.redshift.jdbc.DataSource
();
ds.setURL(CONNECTION_URL);
connection = ds.getConnection();
return connection;
}
```

JDBC URL の取得

SQL クライアントツールから Amazon Redshift クラスターに接続する前に、クラスターの JDBC URL を確認しておく必要があります。JDBC の URL 形式を次に示します。`jdbc:redshift://`*`endpoint`*`:`*`port`*`/`*`database`*

前述の形式のフィールドには、次の値があります。

フィールド	値
`jdbc`	接続用のプロトコルです。
`redshift`	データベースに接続するために Amazon Redshift ドライバーを使用するよう指定するサブプロトコルです。

フィールド	値
endpoint	Amazon Redshift クラスターのエンドポイント。
port	クラスターを起動したときに指定したポート番号。ファイアウォールがある場合、このポートが開いていて使用できることを確認します。
database	クラスター用に作成したデータベース。

JDBC URL の例を次に示します。`jdbc:redshift://examplecluster.abc123xyz789.us-west-2.redshift.amazonaws.com:5439/dev`

URL 値に次の URI 予約文字のいずれかが含まれている場合、値は URL エンコードされている必要があります。

- ;
- +
- {
- }
- [
-]
- &
- =
- ?
- 空のスペース

例えば、PWD 値が `password:password` の場合、その値を使用する接続 URL は次のようになります。

```
jdbc:redshift://redshift.company.us-west-1.redshift.amazonaws.com:9000/dev;UID=amazon;PWD=password%3Apassword
```

JDBC 接続の取得方法については、「クラスター接続文字列を検索する」を参照してください。

クライアントコンピューターがデータベースに接続できない場合、考えられる問題をトラブルシューティングすることができます。詳細については、「Amazon Redshift での接続の問題のトラブルシューティング」を参照してください。

接続 URL の構築

接続 URL を使用して、アクセスしているデータストアに接続情報を指定します。以下は、Amazon Redshift JDBC ドライバーバージョン 2.x で使用する接続 URL の形式です。ここで、[Host] は Amazon Redshift サーバーのエンドポイントであり、[Port] は、サーバーがクライアントリクエストをリッスンするために使用する Transmission Control Protocol (TCP) のポート番号です。

```
jdbc:redshift://[Host]:[Port]
```

以下は、いくつかのオプション設定を指定する接続 URL の形式です。

```
jdbc:redshift://[Host]:[Port]/[database];[Property1]=[Value];
[Property2]=[Value];
```

URL 値に次の URI 予約文字のいずれかが含まれている場合、値は URL エンコードされている必要があります。

- ;
- +
- {
- }
- [
-]
- &
- =
- ?
- 空のスペース

例えば、PWD 値が password:password の場合、その値を使用する接続 URL は次のようになります。

```
jdbc:redshift://redshift.company.us-west-1.redshift.amazonaws.com:9000/
dev;UID=amazon;PWD=password%3Apassword
```

例えば、AWSの米国西部 (北カリフォルニア) リージョンにある、Amazon Redshift クラスターのポート 9000 に接続するとします。また、dev という名前のデータベースにアクセスし、データベースのユーザー名とパスワードを使用して接続を認証することもできます。この場合、次の接続 URL を使用します。

```
jdbc:redshift://redshift.company.us-west-1.redshift.amazonaws.com:9000/
dev;UID=amazon;PWD=amazon
```

次の文字を使用して、残りの URL 文字列内の設定オプションを区切ることができます。

- ;
- ?

例えば、以下の URL 文字列はお互いに等価です。

```
jdbc:redshift://my_host:5439/dev;ssl=true;defaultRowFetchSize=100
```

```
jdbc:redshift://my_host:5439/dev?ssl=true;defaultRowFetchSize=100
```

次の文字を使用して、URL 文字列内で互いに設定オプションを区切ることができます。

- ;
- &

例えば、以下の URL 文字列はお互いに等価です。

```
jdbc:redshift://my_host:5439/dev;ssl=true;defaultRowFetchSize=100
```

```
jdbc:redshift://my_host:5439/dev;ssl=true&defaultRowFetchSize=100
```

次の URL の例では、ログレベル 6 とログのためのパスを指定しています。

```
jdbc:redshift://redshift.amazonaws.com:5439/dev;DSILogLevel=6;LogPath=/home/user/logs;
```

接続 URL ではプロパティを複製しないでください。

指定できる構成オプションの完全なリストについては、「JDBC ドライバーバージョン 2.x の設定オプション」を参照してください。

> **Note**
>
> 接続するときは、クラスターノードの IP アドレスや VPC エンドポイントの IP アドレスを使用しないでください。不必要な停止を避けるため、必ず Redshift エンドポイントを使用してください。例外として、カスタムドメイン名を使用する場合に限り、エンドポイント URL を使用しません。詳細については、「クライアント接続にカスタムドメイン名を使用する」を参照してください。

Apache Maven を使用して JDBC 接続を設定する

Apache Maven は、ソフトウェアプロジェクトを管理する包括的なツールです。AWS SDK for Java では、Apache Maven プロジェクトをサポートしています。詳細については、AWS SDK for Javaデベロッパーガイドの[Apache Maven での SDK の使用]を参照してください。

Apache Maven を使用する場合、Amazon Redshift JDBC ドライバーを使用した Amazon Redshift クラスターへの接続を使用するようにプロジェクトを設定および構築できます。これを行うには、プロジェクトの `pom.xml` ファイルに JDBC ドライバーを依存関係として追加する必要があります。Maven を使用してプロジェクトを構築し JDBC 接続を使用するには、このセクションのステップに従ってください。

Maven 依存関係として JDBC ドライバーを設定するには

1. Amazon リポジトリまたは Maven Central リポジトリのいずれかを `pom.xml` ファイルのリポジトリセクションに追加します。

   > **Note**
   >
   > 以下のコードの例の URL は、ブラウザーで使用するとエラーを返します。この URL は、Maven プロジェクトのコンテキストでのみ使用してください。

   接続に Secure Sockets Layer (SSL) を使用するには、`pom.xml`ファイルに次のリポジトリを追加します。

   ```
   <repositories>
   ```

```
    <repository>
      <id>redshift</id>
      <url>https://s3.amazonaws.com/redshift-maven-repository/release</url>
    </repository>
</repositories>
```

Maven Central のリポジトリの場合は、pom.xmlファイルに次を追加します。

```
<repositories>
    <repository>
      <id>redshift</id>
      <url>https://repo1.maven.org/maven2</url>
    </repository>
</repositories>
```

2. pom.xml ファイルの依存関係セクションで、使用するドライバーのバージョンを宣言します。

   Amazon Redshift には、JDBC 4.2 API と互換性のあるツール用のドライバーが用意されています。これらのドライバーでサポートされる機能については、Amazon Redshift JDBC ドライバーのバージョン 2.x をダウンロードするを参照してください。

   以下の例では、*driver-version*は実際のドライバーのバージョン (例えば 2.1.0.1) に置き換えます。JDBC 4.2 互換ドライバの場合は、以下を使用します。

   ```
   <dependency>
      <groupId>com.amazon.redshift</groupId>
      <artifactId>redshift-jdbc42</artifactId>
      <version>driver-version</version>
   </dependency>
   ```

   このドライバーのクラス名は com.amazon.redshift.Driver です。

IAM データベース認証を使用する場合、Amazon Redshift Maven ドライバーには、次のオプションの依存関係が必要です。

```
<dependency>
      <groupId>com.amazonaws</groupId>
      <artifactId>aws-java-sdk-core</artifactId>
      <version>1.12.23</version>
      <scope>runtime</scope>
      <optional>true</optional>
```

```
</dependency>
    <dependency>
      <groupId>com.amazonaws</groupId>
      <artifactId>aws-java-sdk-redshift</artifactId>
      <version>1.12.23</version>
      <scope>runtime</scope>
      <optional>true</optional>
</dependency>
<dependency>
      <groupId>com.amazonaws</groupId>
      <artifactId>aws-java-sdk-sts</artifactId>
      <version>1.12.23</version>
      <scope>runtime</scope>
      <optional>true</optional>
</dependency>
```

Amazon Redshift JDBC ドライバーを最新バージョンにアップグレードまたは変更するには、まず、依存関係のバージョンセクションをドライバーの最新バージョンに変更します。その後、以下で示すとおり、Maven Clean Plugin でプロジェクトをクリーニングします。

```
mvn clean
```

認証と SSL の設定

不正なアクセスからデータを保護するために、Amazon Redshift データストアでは、すべての接続をユーザー認証情報を使用して認証する必要があります。一部のデータストアでは、一方向認証の有無にかかわらず、Secure Sockets Layer (SSL) プロトコル経由で接続を確立する必要があります。

Amazon Redshift JDBC ドライバーバージョン 2.x では、これらの認証プロトコルが完全にサポートされています。

ドライバーがサポートする SSL バージョンは、使用している JVM のバージョンによって異なります。Java の各バージョンでサポートされている SSL バージョンの詳細については、Java プラットフォームグループの製品管理ブログから [TLS、SSL、および HTTPS の診断]を参照してください。

接続に使用される SSL バージョンは、ドライバーとサーバーの両方でサポートされている最も高いバージョンであり、接続時に決定されます。

接続先の Amazon Redshift サーバーのセキュリティ要件に従って接続を認証するように、Redshift JDBC ドライバーバージョン 2.x を設定します。

接続を認証するには、常に Redshift ユーザー名とパスワードを入力する必要があります。SSL が有効で、サーバー上で必要であるかどうかに応じて、SSL 経由で接続するようにドライバーを設定する必要もあります。または、クライアント (ドライバー自体) がサーバーの身元を確認するために、一方向の SSL 認証を使用する場合もあります。

接続 URL でドライバーに構成情報を提供します。接続 URL の構文の詳細については、接続 URL の構築を参照してください。

SSL は、Transport Layer Security と Secure Sockets Layer の両方の TLS/SSL を示します。このドライバーは、業界標準バージョンの TLS/SSL をサポートしています。

IAM 認証の設定

IAM 認証を使用して Amazon Redshift サーバーに接続する場合は、データソース接続文字列の一部として次のプロパティを設定します。

IAM 認証の詳細については、「Amazon Redshift での Identity and Access Management」 を参照してください。

IAM 認証を使用するには、以下のいずれかの接続文字列形式を使用します。

接続文字列	説明
`jdbc:redshift:iam:// [host]:[port]/[db]`	通常の接続文字列。ドライバーは、ホストから ClusterID とリージョンを推測します。
`jdbc:redshift:iam:// [cluster-id]: [region]/[db]`	ドライバーは、ClusterID とリージョンを指定して、ホストの情報を取得します。
`jdbc:redshift:iam:// [host]/[db]`	デフォルトでドライバーはポート 5439 に設定され、ホストから ClusterID とリージョンを推測します。クラスターの作成、変更、または移行時に選択したポートに応じて、選択したポートへのアクセスを許可します。

プロファイルの指定

IAM 認証を使用している場合は、プロファイル名の下で、必須またはオプションの追加接続プロパティを指定できます。これにより、接続文字列に特定の情報を直接入力することを回避できます。プロファイルのプロパティを使用して、接続文字列でプロファイル名を指定します。

プロファイルは AWS 認証情報ファイルに追加できます。このファイルのデフォルトの場所は、`~/.aws/credentials`です。

次の環境変数 `AWS_CREDENTIAL_PROFILES_FILE` でパスを設定することにより、デフォルト値を変更できます。

プロファイルの詳細については、AWS SDK for Javaの「Working with AWS Credentials」を参照してください。

インスタンスプロファイルの認証情報の使用

IAM ロールに関連付けられている Amazon EC2 インスタンスでアプリケーションを実行している場合は、インスタンスプロファイルの認証情報を使用して接続することができます。

これを行うには、前のテーブルで IAM 接続文字列形式のいずれかを使用し、dbuser 接続プロパティを、接続に使用する Amazon Redshift ユーザー名に設定します。

インスタンスプロファイルの詳細については、IAM ユーザーガイドの「アクセス管理」を参照してください。

認証情報プロバイダーの使用

ドライバーは、次のサービスからの認証情報プロバイダーのプラグインもサポートします。

- AWS IAM アイデンティティーセンター
- Active Directory フェデレーションサービス (ADFS)
- JSON ウェブトークン (JWT) サービス
- Microsoft Azure アクティブディレクトリ (AD) サービスと Browser Microsoft Azure Active Directory (AD) サービス
- Okta サービス
- PingFederate サービス
- Okta、Ping、ADFS などの SAML サービス用のブラウザ SAML

これらのサービスのいずれかを使用する場合は、接続 URL で次のプロパティを指定する必要があります。

- Plugin_Name – 認証情報プロバイダーのプラグインクラスの完全修飾クラスパスです。

- IdP_Host: – Amazon Redshift による認証に使用しているサービスのホストです。
- IdP_Port – 認証サービスのホストがリッスンするポートです。Okta には必要ありません。
- ユーザー – idp_host サーバーのユーザー名。
- パスワード – idp_host ユーザー名に関連付けられたパスワード。
- DbUser – 接続している Amazon Redshift ユーザー名。
- SSL_Insecure – IDP サーバー証明書を検証する必要があるかどうかを示します。
- Client_ID – Azure AD ポータルのユーザー名に関連付けられているクライアント ID。Azure AD にのみ使用されます。
- Client_Secret – Azure AD ポータルのクライアント ID に関連付けられたクライアントシークレットです。Azure AD にのみ使用されます。
- IdP_Tenant – Amazon Redshift アプリケーションの Azure AD テナント ID です。Azure AD にのみ使用されます。
- App_ID – Amazon Redshift アプリケーションの Okta アプリケーション ID です。Okta にのみ使用されます。
- App_Name – Amazon Redshift アプリケーションのオプションの Okta アプリケーション名です。Okta にのみ使用されます。
- Partner_SPID – オプションのパートナー SPID (サービスプロバイダー ID) 値です。PingFederate にのみ使用されます。
- Idc_Region – AWS IAM アイデンティティセンターインスタンスが配置されている AWS リージョン リージョン。AWS IAM アイデンティティセンターでのみ使用されます。
- Issuer_Url – AWS IAM アイデンティティセンターサーバーのインスタンスエンドポイント。AWS IAM アイデンティティセンターでのみ使用されます。

これらのサービスのいずれかにブラウザプラグインを使用している場合は、接続 URL に以下を含めることもできます。

- Login_URL – ブラウザのプラグインを介して Security Assertion Markup Language (SAML) または Azure AD サービスを使用するための、ID プロバイダーのウェブサイト上のリソース URL です。このパラメータは、ブラウザプラグインを使用している場合に必要です。

- Listen_Port – ブラウザのプラグインを介して SAML、Azure AD サービスまたは AWS IAM アイデンティティセンターのサービスを使用する場合に、ID プロバイダーから SAML レスポンスを取得するためにドライバーが使用するポートです。

- IdP_Response_Timeout – ブラウザのプラグインを介して SAML、Azure AD または AWS IAM アイデンティティセンターのサービスを使用する場合に、ドライバーが ID プロバイダーからの SAML レスポンスを待機する時間 (秒単位) です。

追加された接続文字列のプロパティの詳細については、JDBC ドライバーバージョン 2.x の設定オプションを参照してください。

ユーザー名とパスワードのみを使用する

接続先のサーバーが SSL を使用しない場合は、Redshift ユーザー名とパスワードのみを入力して接続を認証できます。

Redshift ユーザー名とパスワードのみを使用して認証を設定するには

1. UID プロパティを Amazon Redshift サーバーにアクセスするための Redshift ユーザー名に設定します。
2. PWD プロパティを Redshift ユーザー名に対応するパスワードに設定します。

本人確認なしで SSL を使用する

接続しているサーバーが SSL を使用していて、ID 検証を必要としない場合には、検証されていない SSL ファクトリを使用するようにドライバーを設定できます。

本人確認を行わずに SSL 接続を構成するには

1. UID プロパティを Amazon Redshift サーバーにアクセスするための Redshift ユーザー名に設定します。
2. PWD プロパティを Redshift ユーザー名に対応するパスワードに設定します。
3. SSLFactory プロパティを com.amazon.redshift.ssl.NonValidatingFactory に設定します。

一方向 SSL 認証の使用

接続しているサーバーが SSL を使用し、証明書がある場合は、一方向認証を使用してサーバーの ID を検証するようにドライバーを構成できます。

一方向認証では、サーバーの ID を検証するために、署名された信頼できる SSL 証明書が必要です。特定の証明書を使用するようにドライバーを構成するか、適切な証明書を含む TrustStore にアクセスするようにドライバーを構成することができます。証明書または TrustStore を指定しない場合、

ドライバーは、デフォルトの Java TrustStore (通常は `jssecacerts` または `cacerts`) を使用します。

一方向 SSL 認証を構成するには

1. UID プロパティを Amazon Redshift サーバーにアクセスするための Redshift ユーザー名に設定します。
2. PWD プロパティを Redshift ユーザー名に対応するパスワードに設定します。
3. SSL プロパティを true に設定します。
4. SSLRootCert プロパティを、ルート CA 証明書がある場所に設定します。
5. デフォルトの Java TrustStores のいずれかを使用していない場合は、次のいずれかの操作を行います。
   - サーバー証明書を指定するには、SSLRootCert プロパティを証明書のフルパスに設定します。
   - TrustStore を指定するには、以下の操作を行います。
     a. keytool プログラムを使用して、使用する TrustStore にサーバー証明書を追加します。
     b. ドライバーを使用して Java アプリケーションを起動するときに使用する TrustStore とパスワードを指定します。次に例を示します。

```
-Djavax.net.ssl.trustStore=[TrustStoreName]
-Djavax.net.ssl.trustStorePassword=[TrustStorePassword]
-Djavax.net.ssl.trustStoreType=[TrustStoreType]
```

6. 1 つを選択してください。
   - 証明書を検証するには、SSLMode プロパティを verify-ca に設定します。
   - 証明書を検証し、証明書のホスト名を検証するには、SSLMode プロパティに verify-full を設定します。

ログ記録の構成

ドライバーのログ記録を有効にすると、問題の診断に利用することができます。

ドライバーに関する情報は、次の方法を使用してログに記録できます。

- ログに記録された情報を .log ファイルに保存するには、ログファイルの使用を参照してください。

- ログに記録された情報を DriverManager で指定された LogStream または LogWriter に送信するには、LogStream または LogWriter の使用を参照してください。

接続 URL でドライバーに構成情報を提供します。接続 URL の構文の詳細については、接続 URL の構築を参照してください。

ログファイルの使用

問題をキャプチャするのに十分な時間だけログ記録を起動します。ログ記録によってパフォーマンスが低下し、ディスク容量が大量に消費される可能性があります。

接続 URL での LogLevel キーを設定して、ログファイルに含まれる詳細情報の量を指定しながら、ログ記録を有効にします。次の表に、Amazon Redshift JDBC ドライバーバージョン 2.x で提供されるログレベルを、冗長性の低いものから高いものの順に示します。

LogLevel 値	説明
1	ドライバーの中断につながる重大なエラーイベントを記録します。
2	ドライバーの実行を続行するよう許可する可能性があるエラーイベントを記録します。
3	アクションが実行されなかった場合にエラーが発生する可能性のあるイベントをログに記録します。このレベルのロギングと、このレベルを超えるロギングのレベルでは、ユーザーのクエリもログに記録されます。
4	ドライバーの進行状況を示す全般的な情報を記録します。
5	ドライバーのデバッグに役立つ詳細な情報を記録します。
6	すべてのドライバーアクティビティを記録します。

ログファイルを使用するログを設定するには

1. LogLevel プロパティを、ログファイルに含める必要があるレベルの情報に設定します。
2. LogPath プロパティを、ログファイルを保存するフォルダへのフルパスに指定します。

   例えば、次の接続 URL はログレベル 3 を有効にし、ログファイルを C:\temp フォルダ `jdbc:redshift://redshift.company.us-west-1.redshift.amazonaws.com:9000/Default;DSILogLevel=3;LogPath=C:\temp` に保存します。
3. 新しい設定が有効になっていることを確認するには、JDBC アプリケーションを再起動し、サーバーに再接続します。

   Amazon Redshift JDBC ドライバーは、LogPath プロパティで指定された場所に次のログファイルを生成します。

   - 接続に固有ではないドライバーのアクティビティをログに記録する redshift_jdbc.log ファイルです。
   - データベースへの接続ごとに redshift_jdbc_connection_[Number].log ファイル。[Number] は各ログファイルを識別する番号です。このファイルは、接続に固有のドライバーアクティビティをログに記録します。

LogPath 値が無効な場合、ドライバーはログに記録された情報を標準出力ストリームに送信します (`System.out`)。

LogStream または LogWriter の使用

問題をキャプチャするのに十分な時間だけログ記録を起動します。ログ記録によってパフォーマンスが低下し、ディスク容量が大量に消費される可能性があります。

接続 URL で LogLevel キーを設定し、DriverManager で指定された LogStream または LogWriter に送信される詳細情報の量を指定しながら、ログ記録を有効にします。

LogStream または LogWriter を使用するログ記録を有効にするには

1. ドライバーの進行状況を示す一般情報をログに記録するようにドライバーを構成するには、LogLevel プロパティを 1 または INFO に設定します。
2. 新しい設定が有効になっていることを確認するには、JDBC アプリケーションを再起動し、サーバーに再接続します。

データ型変換

Amazon Redshift JDBC ドライバーバージョン 2.x では、Amazon Redshift、SQL、および Java データ型の変換を行う多くの一般的なデータ形式がサポートされています。

サポートされているデータ型のマッピングを以下の表に示しています。

Amazon Redshift のタイプ	SQL タイプ	Java 型
BIGINT	SQL_BIGINT	Long
BOOLEAN	SQL_BIT	ブール値
CHAR	SQL_CHAR	文字列
DATE	SQL_TYPE_DATE	java.sql.Date
DECIMAL	SQL_NUMERIC	BigDecimal
DOUBLE PRECISION	SQL_DOUBLE	Double
GEOMETRY	SQL_LONGVARBINARY	byte[]
INTEGER	SQL_INTEGER	整数
OID	SQL_BIGINT	Long
SUPER	SQL_LONGVARCHAR	文字列
REAL	SQL_REAL	浮動小数点
SMALLINT	SQL_SMALLINT	ショート
TEXT	SQL_VARCHAR	文字列
TIME	SQL_TYPE_TIME	java.sql.Time
TIMETZ	SQL_TYPE_TIME	java.sql.Time
TIMESTAMP	SQL_TYPE_TIMESTAMP	java.sql.Timestamp
TIMESTAMPTZ	SQL_TYPE_TIMESTAMP	java.sql.Timestamp

Amazon Redshift のタイプ	SQL タイプ	Java 型
VARCHAR	SQL_VARCHAR	文字列

プリペアドステートメントのサポートを使用する

Amazon Redshift JDBC ドライバーは、プリペアドステートメントをサポートしています。プリペアドステートメントを使用すると、同じ接続中に複数回実行する必要がある、パラメータ化されたクエリのパフォーマンスを向上できます。

プリペアドステートメントは、サーバー側でコンパイルされますが、すぐには実行されない SQL ステートメントです。コンパイルされたステートメントは、オブジェクトまたは接続を閉じるまで、PreparedStatement オブジェクトとしてサーバーに保存されます。そのオブジェクトが存在している間は、ステートメントを再度コンパイルすることなく、異なるパラメータ値を使用して必要な回数だけプリペアドステートメントを実行できます。これによってオーバーヘッドが軽減され、一連のクエリをより迅速に実行することが可能になります。

プリペアドステートメントの詳細については、Oracle からの JDBC の基礎チュートリアルの「プリペアドステートメントの使用」を参照してください。

複数のクエリを含むステートメントを準備できます。例えば、次のプリペアドステートメントには、2 つの INSERT クエリが含まれています。

```
PreparedStatement pstmt = conn.prepareStatement("INSERT INTO
MyTable VALUES (1, 'abc'); INSERT INTO CompanyTable VALUES
(1, 'abc');");
```

これらのクエリは、同じプリペアドステートメント内で指定された他のクエリの結果に依存しないように注意してください。準備ステップの間はクエリが実行されないため、結果はまだ返されておらず、同じ準備済みステートメント内の他のクエリでは使用できません。

例えば、テーブルを作成してから、新しく作成されたテーブルに値を挿入する次のプリペアドステートメントは許可されていません。

```
PreparedStatement pstmt = conn.prepareStatement("CREATE
TABLE MyTable(col1 int, col2 varchar); INSERT INTO myTable
VALUES (1, 'abc');");
```

このステートメントを準備しようとすると、サーバーは送信先のテーブル (myTable) がまだ存在していないことを示すエラーを返します。INSERT クエリを準備する前に、CREATE クエリを実行する必要があります。

JDBC ドライバーの 2.x と 1.x バージョンの違い

このセクションでは、JDBC ドライバーの 2.x および 1.x バージョンによって返される情報の違いについて説明します。JDBC ドライバーのバージョン 1.x の使用は終了されています。

次のテーブルでは、JDBC ドライバーの各バージョンについて、getDatabaseProductName() 関数および getDatabaseProductVersion() 関数によって返される DatabaseMetadata 情報を表示します。JDBC ドライバーのバージョン 2.x は、接続の確立中に値を取得します。JDBC ドライバーのバージョン 1.x は、クエリの結果として値を取得します。

JDBC ドライバーのバージョン	getDatabaseProductName() の結果	getDatabaseProductVersion() の結果
2.x	Redshift	8.0.2
1.x	PostgreSQL	08.00.0002

次のテーブルは、JDBC ドライバーのバージョンごとに getTypeInfo 関数によって返される DatabaseMetadata 情報を示します。

JDBC ドライバーのバージョン	getTypeInfo 結果
2.x	Redshift データ型との整合性
1.x	PostgreSQL データ型と一貫性がある

JDBC ドライバーバージョン 2.x 用の初期化 (.ini) ファイルの作成

Amazon Redshift JDBC ドライバーバージョン 2.x で初期化 (.ini) ファイルを使用すると、システムレベルの設定パラメータを指定できます。例えば、フェデレーション IdP 認証パラメータなどは、アプリケーションごとに異なっている場合があります。.ini ファイルにより、SQL クライアントに必要な構成パラメータを集約することができます。

JDBC ドライバーバージョン 2.x 用に初期化 (.ini) ファイルを作成して、その中に SQL クライアントの構成オプションを記述します。このファイルのデフォルトの名前は `rsjdbc.ini` です。JDBC ドライバーバージョン 2.x は、以下に示す場所から.ini ファイルを探します (優先順位に従いリストされています)。

- `IniFile`SQL クライアントの接続 URL または 接続プロパティダイアログボックス内にある パラメータ。`IniFile` パラメータには、.ini ファイルへのファイル名を含む完全なパスが記述されている必要があります。`IniFile` パラメータの詳細については、「IniFile」を参照してください。`IniFile` パラメータで.ini ファイルの場所を誤って指定した場合には、エラーが表示されます。
- ファイル名を含む完全なパスを指定する環境変数 (AMAZON_REDSHIFT_JDBC_INI_FILE など)。`rsjdbc.ini` を使用するか、ファイル名を指定することができます。AMAZON_REDSHIFT_JDBC_INI_FILE 環境変数で、.ini ファイルの場所が正しく指定されていない場合は、エラーが表示されます。
- ドライバの JAR ファイルが置かれているディレクトリ。
- ユーザーのホームディレクトリ。
- システムの一時ディレクトリ。

.ini ファイルは、[DRIVER] などのセクションにより整理することが可能です。各セクションには、さまざまな接続パラメータを指定するキーと値のペアが含まれます。.ini ファイル内のセクションは、`IniSection`パラメータによりを指定します。`IniSection` パラメータの詳細については、「IniSection」を参照してください。

次に、[DRIVER]、[DEV]、[QA]、および [PRODD] のセクションを使用する .ini ファイル形式の例を示します。[DRIVER] セクションは任意の接続に適用できます。

```
[DRIVER]
key1=val1
key2=val2

[DEV]
key1=val1
key2=val2

[QA]
key1=val1
key2=val2
```

```
[PROD]
key1=val1
key2=val2
```

JDBC ドライバーバージョン 2.x は、次の場所から設定パラメータをロードします (優先順位に従いリストされています)。

- アプリケーションコード内のデフォルト設定パラメータ。
- INIファイルの [DRIVER] セクションのプロパティ (含まれている場合)。
- カスタムセクションの設定パラメータ (`IniSection` オプションが、SQL クライアントの接続 URL または 接続プロパティダイアログボックスで指定されている場合)。
- `getConnection` 呼び出しの中で指定された、接続プロパティオブジェクトに含まれるプロパティ。
- 接続 URL 内で指定されている設定パラメータ。

JDBC ドライバーバージョン 2.x の設定オプション

以下に、Amazon Redshift JDBC ドライバーのバージョン 2.x で指定できるオプションの説明を示します。設定オプションでは、大文字と小文字を区別しません。

接続 URL を使用して、構成プロパティを設定できます。詳細については、「接続 URL の構築」を参照してください。

トピック

- AccessKeyID
- AllowDBUserOverride
- App_ID
- App_Name
- ApplicationName
- AuthProfile
- AutoCreate
- Client_ID
- Client_Secret
- ClusterID
- 圧縮

- connectTimeout
- connectionTimezone
- databaseMetadataCurrentDbOnly
- DbUser
- DbGroups
- DBNAME
- defaultRowFetchSize
- DisableIsValidQuery
- enableFetchRingBuffer
- enableMultiSqlSupport
- EnableTableTypes
- fetchRingBufferSize
- ForceLowercase
- groupFederation
- HOST
- IAMDisableCache
- IAMDuration
- Idc_Client_Display_Name
- Idc_Region
- IdP_Host
- IdP_Partition
- IdP_Port
- IdP_Tenant
- IdP_Response_Timeout
- IniFile
- IniSection
- isServerless
- Issuer_Url
- listen_port

- Login_URL
- loginTimeout
- loginToRp
- LogLevel
- LogPath
- OverrideSchemaPatternType
- Partner_SPID
- パスワード
- Plugin_Name
- PORT
- preferPQ
- Preferred_Role
- プロファイル
- PWD
- queryGroup
- readOnly
- リージョン
- reWriteBatchedInserts
- reWriteBatchedInsertsSize
- roleArn
- roleSessionName
- scope
- SecretAccessKey
- SessionToken
- serverlessAcctId
- serverlessWorkGroup
- socketFactory
- socketTimeout
- SSL

- SSL_Insecure
- SSLCert
- SSLFactory
- SSLKey
- SSLMode
- SSLPassword
- SSLRootCert
- StsEndpointUrl
- tcpKeepAlive
- トークン
- token_type
- UID
- ユーザー
- webIdentityToken

AccessKeyID

- デフォルト値 - なし
- データ型 - 文字列

このパラメータを指定して、ユーザーまたはロールの IAM アクセスキーを入力できます。通常、既存の文字列またはユーザープロファイルを確認することで、キーを見つけることができます。このパラメータを指定する場合は、SecretAccessKeyパラメータも指定する必要があります。JDBC URL で渡す場合は、AccessKeyID を URL エンコードする必要があります。

このパラメータはオプションです。

AllowDBUserOverride

- デフォルト値: 0
- データ型 – 文字列

このオプションは、ドライバーが SAML アサーションの DbUser 値を使用するか、接続 URL の DbUser 接続プロパティで指定された値を使用するかを指定します。

このパラメータはオプションです。

1

ドライバーは SAML アサーションの DbUser 値を使用します。

SAML アサーションで DBUser の値が指定されていない場合、ドライバーは接続プロパティ DBUser で指定された値を使用します。この接続プロパティでも値が指定されていない場合、ドライバーは接続プロファイルで指定された値を使用します。

0

ドライバーは、DBUser接続プロパティで指定された DBUser 値を使用します。

接続プロパティ DBUser で値が指定されていない場合、ドライバーは接続プロファイルで指定された値を使用します。接続プロファイルでも値が指定されていない場合、ドライバーは SAML アサーションの値を使用します。

App_ID

- デフォルト値 – なし
- データ型 – 文字列

Okta が提供する Amazon Redshift アプリケーションに関連付けられた一意の ID。

このパラメータは、Okta サービスを介して認証する場合に必要です。

App_Name

- デフォルト値 – なし
- データ型 – 文字列

Amazon Redshift への接続を認証するために使用する Okta アプリケーションの名前。

このパラメータはオプションです。

ApplicationName

- デフォルト値 – null

- データ型 - 文字列

監査目的のために Amazon Redshift に渡すアプリケーションの名前。

このパラメータはオプションです。

AuthProfile

- デフォルト値 - なし
- データ型 – 文字列

Amazon Redshift への接続に使用する認証プロファイルの名前。

このパラメータはオプションです。

AutoCreate

- デフォルト値 – false
- データ型 – ブール型

このオプションは、指定されたユーザーが存在しない場合に、ドライバーが新しいユーザーを作成するかどうかを指定します。

このパラメータはオプションです。

true

DBUser または固有 ID (UID) のいずれかで指定されたユーザーが存在していない場合は、その名前の新しいユーザーが作成されます。

false

ドライバーが新しいユーザーを作成することはありません。指定したユーザーが存在しない場合、認証は失敗します。

Client_ID

- デフォルト値 – なし
- データ型 – 文字列

Azure AD サービスを使用して接続を認証する際に使用するクライアント ID。

このパラメータは、Azure AD サービスを介して認証する場合に必要です。

Client_Secret

- デフォルト値 – なし
- データ型 – 文字列

Azure AD サービスを使用して接続を認証するときに使用するクライアントシークレット。

このパラメータは、Azure AD サービスを介して認証する場合に必要です。

ClusterID

- デフォルト値 - なし
- データ型 – 文字列

接続する Amazon Redshift クラスターの名前。ドライバーは、指定されたホストからこのパラメータを検出しようとします。Network Load Balancer (NLB) を使用しており IAM を介して接続する場合、ドライバーはそれを検出できないため、この接続オプションを使用して設定することができます。

このパラメータはオプションです。

圧縮

- デフォルト値 - オフ
- データ型 - 文字列

Amazon Redshift サーバーとクライアントまたはドライバー間のワイヤプロトコル通信に使用される圧縮方法。

このパラメータはオプションです。

次の値を指定できます。

- lz4

Amazon Redshift とのワイヤプロトコル通信に使用される圧縮方法を lz4 に設定します。

- off

  Amazon Redshift とのワイヤプロトコル通信には圧縮を使用しません。

connectTimeout

- デフォルト値 – 10
- データ型 – 整数

ソケット接続操作に使用するタイムアウト値。Amazon Redshift 接続の確立に必要な時間がこの値を超えると、接続は利用不可であると見なされます。タイムアウトは秒単位で指定されます。0 の値は、タイムアウトが指定されていないことを意味します。

このパラメータはオプションです。

connectionTimezone

- デフォルト値 – LOCAL
- データ型 - 文字列

セッションレベルのタイムゾーン。

このパラメータはオプションです。

次の値を指定できます。

LOCAL

  セッションレベルのタイムゾーンを LOCAL JVM タイムゾーンに設定します。

SERVER

  セッションレベルのタイムゾーンを、Amazon Redshift サーバー上のユーザーに設定されたタイムゾーンに設定します。次のコマンドを使用して、ユーザーのセッションレベルのタイムゾーンを設定できます。

```
ALTER USER
[...]
```

```
SET TIMEZONE TO [...];
```

databaseMetadataCurrentDbOnly

- デフォルト値 – true
- データ型 – ブール型

このオプションは、メタデータ API が、アクセス可能なすべてのデータベースからデータを取得するか、接続しているデータベースからのみデータを取得するかを指定します。

このパラメータはオプションです。

次の値を指定できます。

true

アプリケーションは、単一のデータベースからメタデータを取得します。

false

アプリケーションは、アクセス可能なすべてのデータベースからメタデータを取得します。

DbUser

- デフォルト値 - なし
- データ型 – 文字列

Amazon Redshift アカウントで使用するユーザー ID。AutoCreate プロパティを有効にしている場合は、現在存在しない ID を使用できます。

このパラメータはオプションです。

DbGroups

- デフォルト値 – PUBLIC
- データ型 – 文字列

DBUser が現在のセッションで参加する、既存のデータベースグループ名のコンマ区切りリスト。

このパラメータはオプションです。

DBNAME

- デフォルト値 – null
- データ型 - 文字列

接続先のデータベースの名前です。このオプションは、JDBC 接続 URL にデータベース名を指定するために使用できます。

このパラメータは必須です。データベース名は、接続 URL、またはクライアントアプリケーションの接続プロパティで指定する必要があります。

defaultRowFetchSize

- デフォルト値: 0
- データ型 – 整数

このオプションは、getFetchSize のデフォルト値を指定します。

このパラメータはオプションです。

次の値を指定できます。

0

1 回のオペレーションですべての行をフェッチします。

正の整数

ResultSet のフェッチを繰り返すごとに、データベースからフェッチする行の数。

DisableIsValidQuery

- デフォルト値 – False
- データ型 – ブール型

このオプションは、Connection.isValid() メソッドを使用してデータベース接続がアクティブかどうかを判断するときに、ドライバーが新しいデータベースクエリを送信するかどうかを指定します。

このパラメータはオプションです。

true

Connection.isValid() を使用してデータベース接続がアクティブかどうかを判断する場合、ドライバーはクエリを送信しません。これにより、データベースサーバーが予期せずシャットダウンした場合、ドライバーがデータベース接続をアクティブとして誤って識別することがあります。

false

Connection.isValid () を使用してデータベース接続がアクティブかどうかを判断する場合、ドライバーはクエリを送信します。

enableFetchRingBuffer

- デフォルト値 – true
- データ型 – ブール型

このオプションは、ドライバーが別のスレッドのリングバッファを使用して行をフェッチすることを指定します。fetchRingBufferSize パラメータは、リングバッファのサイズを指定します。

リングバッファは JDBC に自動メモリ管理を実装し、データ取り出しオペレーション中のメモリ不足 (OOM) エラーを防止します。リングバッファは、バッファされたデータの実際のサイズをリアルタイムでモニタリングし、ドライバーの合計メモリ使用量が設定された制限内に収まるようにします。バッファ容量に達すると、ドライバーはデータフェッチ操作を一時停止し、手動操作を必要とせずにメモリのオーバーフローを防止します。この組み込みのセーフガードにより、OOM エラーが自動的に排除され、ユーザーによる設定は必要ありません。

トランザクションがセミコロンで区切られた複数の SQL コマンドを含むステートメントを検出した場合、そのトランザクションのフェッチリングバッファは false に設定されます。enableFetchRingBuffer の値は変わりません。

このパラメータはオプションです。

> **Note**
>
> リングバッファが無効になっており、フェッチサイズが正しく設定されていない場合、メモリ不足 (OOM) の問題が発生する可能性があります。フェッチサイズの設定の詳細については、こちらを参照してください。

enableMultiSqlSupport

- デフォルト値 – true
- データ型 – ブール型

このオプションは、ステートメント内でセミコロンで区切られた複数の SQL コマンドを処理するかどうかを指定します。

このパラメータはオプションです。

次の値を指定できます。

true

ドライバーは、Statement オブジェクト内でセミコロンで区切られた複数の SQL コマンドを処理します。

false

ドライバーは、単一のステートメントで複数の SQL コマンドに対してエラーを返します。

EnableTableTypes

- デフォルト値 – true
- データ型 – ブール型

このオプションは、`DatabaseMetaData` の `getTables` および `getTableTypes` メソッドの結果で、ドライバーがデータソースから詳細なテーブルタイプ情報を認識するかどうかを指定します。デフォルトでは、ドライバーは詳細なテーブルタイプを認識します。

このパラメータはオプションです。ドライバーバージョン 2.2.8 以降で有効です。

true

ドライバーは次のテーブルタイプを認識します。TABLE、VIEW、SYSTEM TABLE、SYSTEM VIEW、EXTERNAL TABLE、LOCAL TEMPORARY。

false

ドライバーは、詳細なテーブルタイプ情報を一般的な TABLE および VIEW テーブルタイプに正規化します。

fetchRingBufferSize

- デフォルト値 – 1G
- データ型 – 文字列

このオプションは、結果セットの取得時に使用するリングバッファのサイズを指定します。サイズはバイト単位で指定できます。例えば、1 KB の場合は 1K、5,000 バイトの場合は 5000、1 MB の場合は 1M、1 GB の場合は 1G などです。ヒープメモリの割合を指定することもできます。ドライバーは、制限に達すると行のフェッチを停止します。アプリケーションが行を読み込み、リングバッファのスペースを解放すると、フェッチが再開されます。

このパラメータはオプションです。

ForceLowercase

- デフォルト値 – false
- データ型 – ブール型

このオプションは、シングルサインオン認証を使用するときに ID プロバイダーから Amazon Redshift に送信されたすべてのデータベースグループ (DbGroups) をドライバーが小文字にするかどうかを指定します。

このパラメータはオプションです。

true

ドライバーは、ID プロバイダーから送信されるすべてのデータベースグループを小文字にします。

false

ドライバーは、データベースグループを変更しません。

groupFederation

- デフォルト値 – false
- データ型 – ブール型

このオプションは、Amazon Redshift IDP グループを使用するかどうかを指定します。これは GetClusterCredentialsV2 API でサポートされています。

このパラメータはオプションです。

true

Amazon Redshift アイデンティティプロバイダー (IDP) グループを使用します。

false

ユーザーフェデレーションに STS API と GetClusterCredentials を使用し、接続に DbGroups を明示的に指定します。

HOST

- デフォルト値 – null
- データ型 - 文字列

接続先の Amazon Redshift サーバーのホスト名。このオプションは、JDBC 接続 URL にホスト名を指定するために使用できます。

このパラメータは必須です。ホスト名は、接続 URL、またはクライアントアプリケーションの接続プロパティで指定する必要があります。

IAMDisableCache

- デフォルト値 – false
- データ型 – ブール型

このオプションは、IAM 認証情報をキャッシュするかどうかを指定します。

このパラメータはオプションです。

true

IAM 認証情報はキャッシュされません。

false

IAM 認証情報はキャッシュされます。これにより、API ゲートウェイへのリクエストが抑制された場合のパフォーマンスが向上します。

IAMDuration

- デフォルト値 – 900
- データ型 – 整数

一時的な IAM 認証情報が期限切れになるまでの時間 (秒単位)。

- 最小値 – 900
- 最大値 – 3,600

このパラメータはオプションです。

Idc_Client_Display_Name

- デフォルト値 - Amazon Redshift JDBC ドライバー
- データ型 - 文字列

BrowserIdcAuthPlugin を使用しているクライアントに使用される表示名。

このパラメータはオプションです。

Idc_Region

- デフォルト値 - なし
- データ型 - 文字列

IAM アイデンティティセンターインスタンスが配置されている AWS リージョン。

このパラメータは、plugin_name 設定オプションで `BrowserIdcAuthPlugin` を使用して認証する場合にのみ必要です。

IdP_Host

- デフォルト値 – なし
- データ型 – 文字列

Amazon Redshift への認証に使用する IdP (ID プロバイダー) のホスト。これは、接続文字列またはプロファイルのいずれかで指定できます。

このパラメータはオプションです。

IdP_Partition

- デフォルト値 - なし
- データ型 - 文字列

ID プロバイダー (IdP) が設定されているクラウドパーティションを指定します。これにより、ドライバーが接続する IdP 認証エンドポイントが決まります。

このパラメータを空白のままにすると、ドライバーはデフォルトで商用パーティションを使用します。可能な値は以下のとおりです。

- us-gov: IdP が Azure Government で設定されている場合は、この値を使用します。例えば、Azure AD Government はエンドポイント `login.microsoftonline.us` を使用します。
- cn: IdP が中国のクラウドパーティションで設定されている場合は、この値を使用します。例えば、Azure AD China はエンドポイント `login.chinacloudapi.cn` を使用します。

このパラメータはオプションです。

IdP_Port

- デフォルト値 – なし
- データ型 – 文字列

IdP (アイデンティティプロバイダー) が使用するポート。ポートは、接続文字列またはプロファイルで指定できます。デフォルトのポート番号は 5439 です。クラスターの作成、変更、または移行時に選択したポートに応じて、選択したポートへのアクセスを許可します。

このパラメータはオプションです。

IdP_Tenant

- デフォルト値 – なし
- データ型 – 文字列

Amazon Redshift アプリケーションの Azure AD テナント ID です。

このパラメータは、Azure AD サービスを介して認証する場合に必要です。

IdP_Response_Timeout

- デフォルト値 – 120
- データ型 – 整数

ブラウザのプラグインを介して SAML または Azure AD サービスを使用する場合に、ドライバーが ID プロバイダーからの SAML 応答を待機する時間 (秒単位) です。

このパラメータはオプションです。

IniFile

- デフォルト値 - なし
- データ型 – 文字列

.ini ファイルのファイル名を含む完全なパス。例:

```
IniFile="C:\tools\rsjdbc.ini"
```

.ini ファイルの詳細については、「JDBC ドライバーバージョン 2.x 用の初期化 (.ini) ファイルの作成」を参照してください。

このパラメータはオプションです。

IniSection

- デフォルト値 - なし
- データ型 – 文字列

構成オプションを含む.iniファイル内のセクションの名前。.ini ファイルの詳細については、「JDBC ドライバーバージョン 2.x 用の初期化 (.ini) ファイルの作成」を参照してください。

次の例では、.ini ファイル内の [Prod] セクションを指定しています。

```
IniSection="Prod"
```

このパラメータはオプションです。

isServerless

- デフォルト値 – false
- データ型 – ブール型

このオプションは、Amazon Redshift エンドポイントホストがサーバーレス インスタンスであるかどうか指定します。ドライバーは、指定されたホストからこのパラメータを検出しようとします。Network Load Balancer (NLB) を使用している場合、ドライバーはそれを検出できないため、ここで設定できます。

このパラメータはオプションです。

true

Amazon Redshift エンドポイントホストはサーバーレスインスタンスです。

false

Amazon Redshift エンドポイント ホストはプロビジョニングされたクラスターです。

Issuer_Url

- デフォルト値 - なし
- データ型 - 文字列

AWS IAM アイデンティティセンターサーバーのインスタンスエンドポイントを指します。

このパラメータは、plugin_name 設定オプションで `BrowserIdcAuthPlugin` を使用して認証する場合にのみ必要です。

listen_port

- デフォルト値 - 7890
- データ型 – 整数

ブラウザのプラグインを介して SAML、Azure AD または AWS アイデンティティセンターのサービスを使用する場合に、ID プロバイダーから SAML レスポンスを受信するためにドライバーが使用するポートです。

このパラメータはオプションです。

Login_URL

- デフォルト値 – なし
- データ型 – 文字列

ブラウザのプラグインを介して SAML または Azure AD サービスを使用する場合に、ID プロバイダーのウェブサイト上のリソース URL です。

このパラメータは、ブラウザプラグインを介して SAML または Azure AD サービスで認証する場合に必要です。

loginTimeout

- デフォルト値: 0
- データ型 – 整数

サーバーに接続して認証するときにタイムアウトするまで待つ秒数。接続を確立するときにこのしきい値よりも時間がかかると、接続は中止されます。

このプロパティが 0 に設定されている場合、接続はタイムアウトしません。

このパラメータはオプションです。

loginToRp

- デフォルト値 – `urn:amazon:webservices`
- データ型 – 文字列

AD FS 認証の種類に使用する証明書利用者の信頼。

このパラメータはオプションです。

LogLevel

- デフォルト値: 0
- データ型 – 整数

このプロパティを使用して、ログファイルに含まれる詳細情報の量を指定しながら、ドライバーのログ記録を有効または無効にします。

問題をキャプチャするのに十分な時間だけログを有効にします。ログ記録によってパフォーマンスが低下し、ディスク容量が大量に消費される可能性があります。

このパラメータはオプションです。

パラメータを次のいずれかの値に設定します。

0

すべてのログ記録を無効にします。

1

FATAL レベルでログ記録を有効にします。これにより、ドライバーの中止につながる非常に重大なエラーイベントがログに記録されます。

2

ERROR レベルでのログ記録を有効にし、ドライバーの実行を続行できるように許可する可能性があるエラーイベントがログに記録されます。

3

WARNING レベルでのログ記録を有効にします。これにより、アクションを実行しないとエラーが発生する可能性のあるイベントがログに記録されます。

4

INFO レベルでログ記録を有効にし、ドライバーの進行状況を示す全般的な情報をログに記録します。

5

DEBUG レベルでログ記録を有効にします。このレベルでは、ドライバーのデバッグに役立つ詳細な情報がログに記録されます。

6

TRACE レベルでのログ記録を有効にします。これにより、すべてのドライバーのアクティビティがログに記録されます。

ログ記録を有効にすると、ドライバーは LogPath プロパティで指定された場所に次のログファイルを生成します。

- **redshift_jdbc.log** – 接続に固有ではないドライバーのアクティビティを ログに記録するファイル。
- **redshift_jdbc_connection_[Number].log** – データベースに対して行われた各接続のファイル。ここで、[Number]は各ログファイルを他のログファイルと区別する番号です。このファイルは、接続に固有のドライバーアクティビティを ログに記録します。

LogPath 値が無効な場合、ドライバーは ログに記録された情報を標準出力ストリーム System.out に送信します。

LogPath

- デフォルト値 – 現在の作業ディレクトリです。
- データ型 – 文字列

DSILogLevel プロパティが有効になっている場合に、ドライバーが ログファイルを保存するフォルダへのフルパスです。

接続 URL がすべての JDBC アプリケーションと互換性があることを確認するには、別のバックスラッシュを入力して、ファイルパス内のバックスラッシュ (\) を エスケープすることをお勧めします。

このパラメータはオプションです。

OverrideSchemaPatternType

- デフォルト値 – null
- データ型 – 整数

このオプションによって、GetTables コールで使用されるクエリのタイプを無視するかどうかを指定します。

0

スキーマユニバーサルクエリなし

1

ローカルスキーマクエリ

2

外部スキーマクエリ

このパラメータはオプションです。

Partner_SPID

- デフォルト値 – なし
- データ型 – 文字列

PingFederate サービスを使用して接続を認証するときに使用するパートナー SPID (サービスプロバイダー ID) の値。

このパラメータはオプションです。

パスワード

- デフォルト値 - なし
- データ型 – 文字列

IDP を介して IAM 認証を使用して接続する場合、これは IDP_Host サーバーのパスワードです。標準認証を使用する場合、PWD の代わりに Amazon Redshift データベースのパスワードにこれを使用できます。

このパラメータはオプションです。

Plugin_Name

- デフォルト値 – なし
- データ型 – 文字列

特定の認証情報プロバイダーのプログインを実装する完全修飾クラス名。

このパラメータはオプションです。

以下のプロバイダーオプションがサポートされています。

- **`AdfsCredentialsProvider`** – Active Directory フェデレーションサービス。

- **AzureCredentialsProvider** – Microsoft Azure Active Directory (AD) サービス。
- **BasicJwtCredentialsProvider** – JSON ウェブトークン (JWT) サービス。
- **BasicSamlCredentialsProvider** – Security Assertion Markup Language (SAML) 認証情報。多数の SAML サービスプロバイダーで使用できます。
- **BrowserAzureCredentialsProvider** – Microsoft Azure Active Directory (AD) サービスのブラウザ。
- **BrowserAzureOAuth2CredentialsProvider** – ネイティブ認証用の Microsoft Azure Active Directory (AD) サービスのブラウザ。
- **BrowserIdcAuthPlugin** – AWS IAM アイデンティティセンターを使用する認可プラグイン。
- **BrowserSamlCredentialsProvider** – Okta、Ping、ADFS などの SAML サービス用のブラウザ SAML。
- **IdpTokenAuthPlugin** — AWS IAM アイデンティティセンターのトークン、または AWS IAM アイデンティティセンターにリンクされているウェブ ID プロバイダーからの OpenID Connect (OIDC) JSON ベースの ID トークン (JWT) を受け入れる認可プラグイン。
- **OktaCredentialsProvider** – Okta サービス。
- **PingCredentialsProvider** – PingFederate サービス。

PORT

- デフォルト値 – null
- データ型 – 整数

接続先の Amazon Redshift サーバーのポート。このオプションは、JDBC 接続 URL にポート名を指定するために使用できます。

このパラメータはオプションです。

preferPQ

- デフォルト値 – true
- データ型 – ブール型

このオプションは、ドライバーが TLS 1.3 ClientHello でポスト量子ハイブリッドキー交換グループをアドバタイズするかどうかを制御します。ポスト量子グループが提供され、サーバーもそれらをサ

ポートする場合、サーバーはその中から 1 つを選択します。それ以外の場合、ハンドシェイクは従来のキー交換に透過的にフォールバックします。

このパラメータはオプションです。ドライバーバージョン 2.2.8 以降で有効です。

true

ランタイムでサポートされている場合、ドライバーはポスト量子ハイブリッド TLS キー交換グループをアドバタイズします。

false

ドライバーはポスト量子グループをアドバタイズしません。ドライバーが提供するポスト量子キー交換グループがネットワークミドルボックスに認識されず、TLS ハンドシェイクがブロックされる場合は、この設定を使用します。

Preferred_Role

- デフォルト値 – なし
- データ型 – 文字列

Amazon Redshift への接続中に引き受ける IAM ロール。

このパラメータはオプションです。

プロファイル

- デフォルト値 - なし
- データ型 – 文字列

IAM 認証に使用するプロファイルの名前。このプロファイルには、接続文字列で指定されていない追加の接続プロパティが含まれます。

このパラメータはオプションです。

PWD

- デフォルト値 - なし
- データ型 - 文字列

プロパティ UID を使用して指定した Amazon Redshift ユーザー名に対応するパスワード。

このパラメータはオプションです。

queryGroup

- デフォルト値 – null
- データ型 – 文字列

このオプションは、適切なクエリグループにクエリを割り当てることで、実行時にクエリをキューに割り当てます。セッションに対してクエリグループが設定されます。接続で実行されるすべてのクエリは、このクエリグループに属します。

このパラメータはオプションです。

readOnly

- デフォルト値 – false
- データ型 – ブール型

このプロパティは、ドライバーが読み込み専用モードであるかどうかを指定します。

このパラメータはオプションです。

true

接続は読み込み専用モードであり、データストアに書き込むことはできません。

false

接続は読み込み専用モードではなく、データストアに書き込むことができます。

リージョン

- デフォルト値 – null
- データ型 – 文字列

このオプションは、クラスターが配置されている AWS リージョンを指定します。StsEndPoint オプションを指定すると、[Region (リージョン)] オプションは無視されます。Redshift `GetClusterCredentials` API オペレーションは、リージョンオプションも使用します。

このパラメータはオプションです。

reWriteBatchedInserts

- デフォルト値 – false
- データ型 – ブール型

このオプションは、互換性のある INSERT ステートメントを書き換えてバッチに統合する最適化を有効にします。

このパラメータはオプションです。

reWriteBatchedInsertsSize

- デフォルト値 – 128
- データ型 – 整数

このオプションは、互換性のある INSERT ステートメントを書き換えてバッチに統合する最適化を有効にします。この値は、2 の累乗だけ指数関数的に増加する必要があります。

このパラメータはオプションです。

roleArn

- デフォルト値 - なし
- データ型 – 文字列

ロールの Amazon リソースネーム (ARN)。Plugin_Name オプションに BasicJwtCredentialsProvider を指定する場合、必ずこのパラメータを指定してください。ARN は次の形式で指定します。

`arn:partition:service:region:account-id:resource-id`

Plugin_Name オプションに BasicJwtCredentialsProvider を指定する場合、このパラメータは必須です。

roleSessionName

- デフォルト値 – jwt_redshift_session
- データ型 – 文字列

引き受けたロールセッションの識別子。通常は、アプリケーションのユーザーに関連付けられている名前または識別子を渡します。アプリケーションが使用する一時的なセキュリティ認証情報は、そのユーザーに関連付けられます。Plugin_Name オプションに BasicJwtCredentialsProvider を指定する場合、このパラメータを指定できます。

このパラメータはオプションです。

scope

- デフォルト値 - なし
- データ型 - 文字列

ユーザーが同意可能なスコープをスペースで区切ったリストです。このパラメータを指定すると、Microsoft Azure アプリケーションが呼び出す API の同意を得られるようになります。Plugin_Name オプションに BrowserAzureOAuth2CredentialsProvider を指定する場合、このパラメータを指定できます。

このパラメータは、BrowserAzureOAuth2CredentialsProvider プラグインでは必須です。

SecretAccessKey

- デフォルト値 - なし
- データ型 – 文字列

ユーザーまたはロールの IAM アクセスキー。これを指定する場合は、AccessKeyID も指定する必要があります。JDBC URL で渡す場合は、SecretAccessKey を URL エンコードする必要があります。

このパラメータはオプションです。

SessionToken

- デフォルト値 - なし
- データ型 – 文字列

認証に使用している IAM ロールに関連付けられた一時的な IAM セッショントークン。JDBC URL で渡される場合、一時的な IAM セッショントークンは URL エンコードされている必要があります。

このパラメータはオプションです。

serverlessAcctId

- デフォルト値 – null
- データ型 - 文字列

Amazon Redshift Serverless アカウント ID。ドライバーは、指定されたホストからこのパラメータを検出しようとします。Network Load Balancer (NLB) を使用している場合、ドライバーはそれを検出できないため、ここで設定できます。

このパラメータはオプションです。

serverlessWorkGroup

- デフォルト値 – null
- データ型 - 文字列

Amazon Redshift Serverless ワークグループ名。ドライバーは、指定されたホストからこのパラメータを検出しようとします。Network Load Balancer (NLB) を使用している場合、ドライバーはそれを検出できないため、ここで設定できます。

このパラメータはオプションです。

socketFactory

- デフォルト値 – null
- データ型 - 文字列

このオプションは、ソケット作成用のソケットファクトリを指定します。

このパラメータはオプションです。

socketTimeout

- デフォルト値: 0
- データ型 – 整数

タイムアウトする前にソケット読み込みオペレーションを待つ秒数。このオペレーションにこのしきい値よりも時間がかかると、接続は閉じます。このプロパティが 0 に設定されていると、接続はタイムアウトしません。

このパラメータはオプションです。

SSL

- デフォルト値 – TRUE
- データ型 – 文字列

このプロパティを使用して、SSL による接続を有効または無効にします。

このパラメータはオプションです。

次の値を指定できます。

TRUE

ドライバーは SSL 経由でサーバーに接続します。

FALSE

ドライバーは SSL を使用せずにサーバーに接続します。このオプションは IAM 認証ではサポートされていません。

または、AuthMech プロパティを構成することもできます。

SSL_Insecure

- デフォルト値 – true
- データ型 – 文字列

このプロパティは、IDP ホストサーバー証明書を検証する必要があるかどうかを示します。

このパラメータはオプションです。

次の値を指定できます。

true

ドライバーは IDP サーバー証明書の信頼性を確認しません。

false

ドライバーは、IDP サーバー証明書の信頼性をチェックします。

SSLCert

- デフォルト値 - なし
- データ型 – 文字列

SSL を使用するときに Amazon Redshift サーバーインスタンスを確認するための信頼できる追加 CA 証明書を含む .pem または .crt ファイルのフルパス。

このパラメータは、SSLKey を指定した場合に必要です。

SSLFactory

- デフォルト値 - なし
- データ型 – 文字列

サーバー証明書を使用せずに TLS/SSL 経由でサーバーに接続するときに使用する SSL ファクトリ。

SSLKey

- デフォルト値 - なし
- データ型 – 文字列

SSLCert で指定された証明書を検証するための PKCS8 キーファイルを含む .der ファイルのフルパス。

このパラメータは、SSLCert を指定した場合に必要です。

SSLMode

- デフォルト値 – verify-ca
- データ型 – 文字列

このプロパティを使用して、TLS/SSL が有効になっている場合にドライバーが証明書を検証する方法を指定します。

このパラメータはオプションです。

次の値を指定できます。

verify-ca

ドライバーは、証明書が信頼できる認定権限 (CA) から送信されていることを確認します。

verify-full

ドライバは、証明書が信頼できる CA から取得されていること、および証明書のホスト名が接続 URL で指定されたホスト名と一致することを確認します。

> **Note**
>
> 可能な限り、SSL 接続に `verify-full` を使用することをお勧めします。

SSLPassword

- デフォルト値: 0
- データ型 – 文字列

SSLKey で指定された暗号化キーファイルのパスワード。

SSLKey が指定され、キーファイルが暗号化されている場合、このパラメータは必須です。

SSLRootCert

- デフォルト値 - なし
- データ型 – 文字列

SSL を使用するときに Amazon Redshift サーバーインスタンスを確認するためのルート CA 証明書を含む .pem または .crt ファイルのフルパス。

StsEndpointUrl

- デフォルト値 – null
- データ型 – 文字列

AWS Security Token Service(AWS STS) エンドポイントを指定することができます。このオプションを指定すると、リージョンオプションは無視されます。このエンドポイントにはセキュアプロトコル (HTTPS) のみを指定できます。

tcpKeepAlive

- デフォルト値 – TRUE
- データ型 – 文字列

このプロパティを使用して、TCP キープアライブを有効または無効にします。

このパラメータはオプションです。

次の値を指定できます。

TRUE

接続のタイムアウトを防ぐために TCP キープアライブを使用するようにドライバーを設定します。

FALSE

ドライバーは TCP キープアライブを使用しません。

トークン

- デフォルト値 - なし
- データ型 - 文字列

AWS IAM アイデンティティセンターが提供するアクセストークン、または AWS IAM アイデンティティセンターにリンクされているウェブ ID プロバイダーが提供する OpenID Connect (OIDC) JSON ウェブトークン (JWT)。アプリケーションは、このトークンを生成するために、AWS IAM アイデンティティセンターまたは AWS IAM アイデンティティセンターにリンクされている ID プロバイダーにより、アプリケーションのユーザーを認証する必要があります。

このパラメータは `IdpTokenAuthPlugin` で機能します。

token_type

- デフォルト値 - なし

- データ型 - 文字列

IdpTokenAuthPlugin で使用しているトークンのタイプ。

次の値を指定できます。

ACCESS_TOKEN

AWS IAM アイデンティティセンターが提供するアクセストークンを使用する場合は、これを入力します。

EXT_JWT

AWS IAM アイデンティティセンターと統合されているウェブベースの ID プロバイダーが提供する OpenID Connect (OIDC) JSON ウェブトークン (JWT) を使用する場合は、これを入力します。

このパラメータは IdpTokenAuthPlugin で機能します。

UID

- デフォルト値 - なし
- データ型 - 文字列

データベースへのアクセスに使用するデータベースユーザー名。

このパラメータは必須です。

ユーザー

- デフォルト値 - なし
- データ型 - 文字列

IDP を介して IAM 認証を使用して接続する場合、これは idp_host サーバーのユーザー名です。標準認証を使用する場合、これを Amazon Redshift データベースユーザー名として使用できます。

このパラメータはオプションです。

webIdentityToken

- デフォルト値 - なし

- データ型 - 文字列

ID プロバイダによって提供される OAuth 2.1 アクセストークンまたは OpenID Connect ID トークン。アプリケーションは、ウェブアイデンティティのプロバイダーを使用してアプリケーションのユーザーを認証することによって、このトークンを取得する必要があります。Plugin_Name オプションに BasicJwtCredentialsProvider を指定する場合、必ずこのパラメータを指定してください。

Plugin_Name オプションに BasicJwtCredentialsProvider を指定する場合、このパラメータは必須です。

JDBC ドライバーバージョン 2.x の以前のバージョン

ツールに特定のバージョンのドライバーが必要な場合にのみ、Amazon Redshift JDBC ドライバーバージョン 2.x の以前のバージョンをダウンロードします。

以前の JDBC 4.2 互換 JDBC ドライバーバージョン 2.x のドライバーを以下に示します。

- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.2.8/redshift-jdbc42-2.2.8.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.2.7/redshift-jdbc42-2.2.7.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.2.6/redshift-jdbc42-2.2.6.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.2.5/redshift-jdbc42-2.2.5.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.2.4/redshift-jdbc42-2.2.4.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.2.3/redshift-jdbc42-2.2.3.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.2.2/redshift-jdbc42-2.2.2.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.2.1/redshift-jdbc42-2.2.1.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.2.0/redshift-jdbc42-2.2.0.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.34/redshift-jdbc42-2.1.0.34.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.33/redshift-jdbc42-2.1.0.33.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.32/redshift-jdbc42-2.1.0.32.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.30/redshift-jdbc42-2.1.0.30.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.29/redshift-jdbc42-2.1.0.29.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.28/redshift-jdbc42-2.1.0.28.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.26/redshift-jdbc42-2.1.0.26.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.25/redshift-jdbc42-2.1.0.25.zip

- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.24/redshift-jdbc42-2.1.0.24.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.23/redshift-jdbc42-2.1.0.23.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.22/redshift-jdbc42-2.1.0.22.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.21/redshift-jdbc42-2.1.0.21.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.20/redshift-jdbc42-2.1.0.20.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.19/redshift-jdbc42-2.1.0.19.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.18/redshift-jdbc42-2.1.0.18.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.17/redshift-jdbc42-2.1.0.17.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.16/redshift-jdbc42-2.1.0.16.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.15/redshift-jdbc42-2.1.0.15.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.14/redshift-jdbc42-2.1.0.14.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.13/redshift-jdbc42-2.1.0.13.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.12/redshift-jdbc42-2.1.0.12.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.11/redshift-jdbc42-2.1.0.11.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.10/redshift-jdbc42-2.1.0.10.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.9/redshift-jdbc42-2.1.0.9.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.8/redshift-jdbc42-2.1.0.8.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.7/redshift-jdbc42-2.1.0.7.zip

Amazon Redshift Python コネクタ

Python 用 Amazon Redshift コネクタを使用することで、AWSSDK for Python (Boto3)、pandasと Numerical Python (NumPy) との連携作業が可能になります。pandas の詳細については、pandas GitHub のレポジトリを参照してください。NumPy の詳細については、NumPy GitHub のレポジトリを参照してください。

Amazon Redshift Python コネクタは、オープンソースのソリューションを提供します。ソースコードの閲覧、機能拡張のリクエスト、問題の報告、およびコントリビューションを行うことができます。

Amazon Redshift Python コネクタを使用するには、Python バージョン 3.6 以降が必要です。詳細については、[Amazon Redshift Python ドライバーライセンス契約]を参照してください。

Amazon Redshift Python コネクタには、以下の機能があります。

- AWS Identity and Access Management(IAM) 認証。詳細については、「Amazon Redshift での Identity and Access Management」を参照してください。
- フェデレーション API アクセスを使用した、アイデンティティプロバイダー認証 フェデレーション API アクセスは、次のような企業のアイデンティティプロバイダーに対応しています。
  - Azure AD 詳細については、AWSビッグデータブログ投稿[Amazon Redshift アクセスを Microsoft Azure AD シングルサインオンでフェデレーションする]を参照してください。
  - アクティブディレクトリフェデレーションサービス 詳細については、AWS ビッグデータブログ投稿「Active Directory フェデレーションサービス (AD FS) を使用した Amazon Redshift クラスターへのアクセスをフェデレーションする: パート 1」を参照してください。
  - Okta 詳細については、AWS ビッグデータブログ投稿「ID プロバイダーとして Okta を使用して Amazon Redshift アクセスをフェデレーションする」を参照してください。
  - PingFederate 詳細については、[PingFederate サイト]を参照してください。
  - JumpCloud 詳細については、[JumpCloud のサイト]を参照してください。
- Amazon Redshift のデータ型

Amazon Redshift Python コネクタは、Python データベース API 仕様 2.0 を実装しています。詳細については、Pythonのウェブサイトの[PEP 249-Python データベース API 仕様 v2.0]を参照してください。

トピック

- Amazon Redshift Python コネクタのインストール
- Amazon Redshift Python コネクタの設定オプション
- Python コネクタのインポート
- Python コネクタと NumPy の連携
- Python コネクタとpandasの連携
- アイデンティティプロバイダープラグインの使用
- Amazon Redshift Python コネクタの使用例
- Amazon Redshift Python コネクタの API リファレンス

Amazon Redshift Python コネクタのインストール

以下のいずれかの方法で、Amazon Redshift Python コネクタをインストールすることができます。

- Python パッケージインデックス (PyPI)

- Conda
- GitHub リポジトリのクローン作成

PyPI からの Python コネクタのインストール

Python パッケージインデックス (PyPI) から Python コネクタをインストールするには、pip を使用します。これを行うには、以下のコマンドを実行します。

```
>>> pip install redshift_connector
```

コネクタは、仮想環境内にインストールできます。これを行うには、以下のコマンドを実行します。

```
>>> pip install redshift_connector
```

オプションとして、コネクタと一緒に pandas と NumPy をインストールすることができます。

```
>>> pip install 'redshift_connector[full]'
```

pip の詳細については、[pip site]を参照してください。

Conda からの Python コネクタのインストール

Python コネクタは Anaconda.org からインストールできます。

```
>>>conda install -c conda-forge redshift_connector
```

AWSから GitHub リポジトリをクローンして Python コネクターをインストールします。

Python コネクタをソースからインストールするには、AWSから GitHub リポジトリをクローンします。Python と virtualenv をインストールしたら、環境を設定し、次のコマンドを実行して必要な依存関係をインストールします。

```
$ git clone https://github.com/aws/amazon-redshift-python-driver.git
$ cd amazon-redshift-python-driver
$ virtualenv venv
$ . venv/bin/activate
$ python -m pip install -r requirements.txt
$ python -m pip install -e .
```

```
$ python -m pip install redshift_connector
```

Amazon Redshift Python コネクタの設定オプション

以下に、Amazon Redshift Python コネクタに指定できるオプションの説明を示します。以下のオプションは、特に指定がない限り、利用可能な最新のコネクタバージョンに適用されます。

access_key_id

- デフォルト値 - なし
- データ型 - 文字列

IAM データベース認証用に設定された IAM ロールまたはユーザーのアクセスキー。

このパラメータはオプションです。

allow_db_user_override

- デフォルト値 – false
- データ型 – ブール型

正

コネクタが、DbUserSecurity Assertion Markup Language (SAML) アサーションの値を使用することを指定します。

誤

DbUser接続パラメータの値を使用することを指定します。

このパラメータはオプションです。

app_name

- デフォルト値 - なし
- データ型 - 文字列

認証に使用する アイデンティティ プロバイダー (IdP) アプリケーションの名前。

このパラメータはオプションです。

application_name

- デフォルト値 - なし
- データ型 - 文字列

監査目的のために Amazon Redshift に渡すクライアントアプリケーションの名前。指定したアプリケーション名は、SYS_CONNECTION_LOG テーブルの「application_name」列に表示されます。これにより、問題をデバッグするときに接続ソースを追跡およびトラブルシューティングできます。

このパラメータはオプションです。

auth_profile

- デフォルト値 - なし
- データ型 - 文字列

JSON としての接続プロパティを持つ Amazon Redshift 認証プロファイルの名前。接続パラメータの名前付けの詳細については、RedshiftPropertyクラスを参照してください。RedshiftPropertyクラスは、エンドユーザーから提供された接続パラメータや、必要に応じてIAM 認証プロセスで生成された接続パラメータ (一時的な IAM 認証情報など) を格納します。詳細については、[RedShiftProperty クラス]を参照してください。

このパラメータはオプションです。

auto_create

- デフォルト値 – false
- データ型 – ブール型

ユーザーが存在しない場合にユーザーを作成するかどうかを示す値。

このパラメータはオプションです。

client_id

- デフォルト値 - なし
- データ型 - 文字列

Azure IdP からのクライアント ID。

このパラメータはオプションです。

client_secret

- デフォルト値 - なし
- データ型 - 文字列

Azure IdP からのクライアントシークレット。

このパラメータはオプションです。

cluster_identifier

- デフォルト値 - なし
- データ型 - 文字列

Amazon Redshift クラスターのクラスター識別子。

このパラメータはオプションです。

credentials_provider

- デフォルト値 - なし
- データ型 - 文字列

Amazon Redshift での認証に使用される IdP。以下の値が有効です。

- `AdfsCredentialsProvider`
- `AzureCredentialsProvider`
- `BrowserAzureCredentialsProvider`
- `BrowserAzureOAuth2CredentialsProvider`
- `BrowserIdcAuthPlugin` - AWS IAM アイデンティティセンターを使用する認可プラグイン。
- `BrowserSamlCredentialsProvider`
- `IdpTokenAuthPlugin` - AWS IAM アイデンティティセンターのトークン、または AWS IAM アイデンティティセンターにリンクされているウェブ ID プロバイダーからの OpenID Connect (OIDC) JSON ベースの ID トークン (JWT) を受け入れる認可プラグイン。

- PingCredentialsProvider
- OktaCredentialsProvider

このパラメータはオプションです。

データベース

- デフォルト値 - なし
- データ型 - 文字列

接続先のデータベースの名前。

このパラメータは必須です。

database_metadata_current_db_only

- デフォルト値 – true
- データ型 – ブール型

アプリケーションがマルチデータベースデータ共有カタログをサポートしているかどうかを示す値。デフォルト値の True は、下位互換性のためにアプリケーションがマルチデータベースデータ共有カタログをサポートしていないことを示します。

このパラメータはオプションです。

db_groups

- デフォルト値 - なし
- データ型 - 文字列

DbUser で示されるユーザーが現在のセッションで参加している既存のデータベースグループ名のコンマ区切りのリスト。

このパラメータはオプションです。

db_user

- デフォルト値 - なし

- データ型 - 文字列

Amazon Redshift で使用するユーザー ID。

このパラメータはオプションです。

enable_table_types

- デフォルト値 – true
- データ型 – ブール型

コネクタが `get_tables` および `get_table_types` メタデータメソッドの結果で、データソースから詳細なテーブルタイプ情報を認識するかどうかを指定します。デフォルトでは、コネクタは詳細なテーブルタイプを認識します。

- True: コネクタは次のテーブルタイプを認識します。TABLE、VIEW、SYSTEM TABLE、SYSTEM VIEW、EXTERNAL TABLE、LOCAL TEMPORARY。
- False: コネクタは、詳細なテーブルタイプ情報を一般的な TABLE および VIEW テーブルタイプに正規化します。

このパラメータはオプションです。コネクタバージョン 2.1.16 以降で使用できます。

endpoint_url

- デフォルト値 - なし
- データ型 - 文字列

Amazon Redshift エンドポイント URL。このオプションは、AWS内部使用の場合にのみ利用できます。

このパラメータはオプションです。

group_federation

- デフォルト値 – false
- データ型 – ブール型

このオプションは、Amazon Redshift IDP グループを使用するかどうかを指定します。

このパラメータはオプションです。

true

Amazon Redshift アイデンティティプロバイダー (IDP) グループを使用します。

false

ユーザーフェデレーションに STS API と GetClusterCredentials を使用して、接続に db_groups を指定します。

ホスト

- デフォルト値 - なし
- データ型 - 文字列

Amazon Redshift クラスターのホスト名。

このパラメータはオプションです。

iam

- デフォルト値 – false
- データ型 – ブール型

IAM 認証が有効になっています。

このパラメータは必須です。

iam_disable_cache

- デフォルト値 – false
- データ型 – ブール型

このオプションは、IAM 認証情報をキャッシュするかどうかを指定します。デフォルトでIAM 認証情報はキャッシュされます。これにより、API gateway へのリクエストがスロットルされた場合のパフォーマンスが向上します。

このパラメータはオプションです。

idc_client_display_name

- デフォルト値 — Amazon Redshift Python コネクタ
- データ型 - 文字列

BrowserIdcAuthPlugin を使用しているクライアントに使用される表示名。

このパラメータはオプションです。

idc_region

- デフォルト値 - なし
- データ型 - 文字列

AWS IAM アイデンティティセンターインスタンスが配置されている AWS リージョン。

このパラメータは、credentials_provider 設定オプションで `BrowserIdcAuthPlugin` を使用して認証する場合にのみ必要です。

idp_partition

- デフォルト値 - なし
- データ型 - 文字列

ID プロバイダー (IdP) が設定されているクラウドパーティションを指定します。これにより、ドライバーが接続する IdP 認証エンドポイントが決まります。

このパラメータを空白のままにすると、ドライバーはデフォルトで商用パーティションを使用します。可能な値は以下のとおりです。

- `us-gov`: IdP が Azure Government で設定されている場合は、この値を使用します。例えば、Azure AD Government はエンドポイント `login.microsoftonline.us` を使用します。
- `cn`: IdP が中国のクラウドパーティションで設定されている場合は、この値を使用します。例えば、Azure AD China はエンドポイント `login.chinacloudapi.cn` を使用します。

このパラメータはオプションです。

idpPort

- デフォルト値— 7890
- データ型 – 整数

IdP が SAML アサーションを送信する先のリッスンポートです。

このパラメータは必須です。

idp_response_timeout

- デフォルト値 – 120
- データ型 – 整数

IdP から SAML アサーションを取得する際のタイムアウトです。

このパラメータは必須です。

idp_tenant

- デフォルト値 - なし
- データ型 - 文字列

IdP テナント。

このパラメータはオプションです。

issuer_url

- デフォルト値 - なし
- データ型 - 文字列

AWS IAM アイデンティティセンターサーバーのインスタンスエンドポイントを指します。

このパラメータは、credentials_provider 設定オプションで `BrowserIdcAuthPlugin` を使用して認証する場合にのみ必要です。

listen_port

- デフォルト値— 7890

- データ型 – 整数

ブラウザのプラグインを介して SAML、Azure AD または AWS IAM アイデンティティセンターのサービスを使用する場合に、ID プロバイダーから SAML レスポンスを受信するためにドライバーが使用するポートです。

このパラメータはオプションです。

login_url

- デフォルト値 - なし
- データ型 – 文字列

IdP のシングルサインオン URL。

このパラメータはオプションです。

max_prepared_statements

- デフォルト値 – 1000
- データ型 – 整数

接続ごとにキャッシュされる準備済みステートメントの最大数です。このパラメータを 0 に設定すると、キャッシュメカニズムが無効になります。このパラメータに負の数を入力すると、デフォルト値に設定されます。

このパラメータはオプションです。

numeric_to_float

- デフォルト値 – false
- データ型 – ブール型

このオプションは、コネクタが数値データ型の値を decimal.Decimal から float に変換するかどうかを指定します。デフォルトでは、コネクタは数値データ型の値を decimal.Decimal で受け取り、変換しません。

精度が求められるユースケースでは、結果が丸められる場合もあるため、numeric_to_float を有効にすることは推奨されません。

decimal.Decimal および decimal.Decimal と float 間のトレードオフの詳細については、Python ウェブサイトの「decimal — Decimal fixed point and floating point arithmetic」(10 進数 — 固定小数点数と浮動小数点数の演算) を参照してください。

このパラメータはオプションです。

partner_sp_id

- デフォルト値 - なし
- データ型 - 文字列

Ping による認証に使用されるパートナー SP IDです。

このパラメータはオプションです。

password

- デフォルト値 - なし
- データ型 - 文字列

認証に使用されるパスワード。

このパラメータはオプションです。

ポート

- デフォルト値 — 5439
- データ型 – 整数

Amazon Redshift クラスターのポートナンバー。

このパラメータは必須です。

preferred_role

- デフォルト値 - なし
- データ型 - 文字列

現在の接続に優先される IAM ロール。

このパラメータはオプションです。

principal_arn

- デフォルト値 - なし
- データ型 - 文字列

ポリシーを生成する対象のユーザーまたは IAM ロールの Amazon リソースネーム (ARN)。ロールにポリシーをアタッチし、そのロールをユーザーに割り当ててアクセスを許可することをお勧めします。

このパラメータはオプションです。

profile

- デフォルト値 - なし
- データ型 - 文字列

AWS認証情報を含むAWS認証情報ファイル内のプロファイルの名前。

このパラメータはオプションです。

provider_name

- デフォルト値 - なし
- データ型 - 文字列

Redshift ネイティブ認証プロバイダの名前です。

このパラメータはオプションです。

リージョン

- デフォルト値 - なし
- データ型 - 文字列

クラスターが配置されているAWS リージョン。

このパラメータはオプションです。

role_arn

- デフォルト値 - なし
- データ型 - 文字列

発信者が想定しているロールの Amazon リソースネーム (ARN)です。このパラメーターは、JwtCredentialsProviderで示されるプロバイダーが使用します。

JwtCredentialsProviderプロバイダーの場合、このパラメータは必須です。それ以外の場合、このパラメータはオプションです。

role_session_name

- デフォルト値 – jwt_redshift_session
- データ型 - 文字列

引き受けたロールセッションの識別子。通常は、アプリケーションを使用するユーザーに関連付けられている名前または識別子を渡します。アプリケーションが使用する一時的なセキュリティ認証情報は、そのユーザーに関連付けられます。このパラメータは、JwtCredentialsProviderで示されるプロバイダーが使用します。

このパラメータはオプションです。

scope

- デフォルト値 - なし
- データ型 - 文字列

ユーザーが同意可能なスコープをスペースで区切ったリストです。このパラメータを指定すると、アプリケーションが呼び出す API の同意を得られるようになります。credentials_provider オプションに BrowserAzureOAuth2CredentialsProvider を指定する場合、このパラメータを指定できます。

このパラメータは、BrowserAzureOAuth2CredentialsProvider プラグインでは必須です。

secret_access_key_id

- デフォルト値 - なし
- データ型 - 文字列

IAM データベース認証用に設定された IAM ロールまたはユーザーのシークレットアクセスキー。

このパラメータはオプションです。

session_token

- デフォルト値 - なし
- データ型 - 文字列

IAM データベース認証用に設定された IAM ロールまたはユーザーのアクセスキー。このパラメータは、一時的なAWS認証情報を使用している場合に必要です。

このパラメータはオプションです。

serverless_acct_id

- デフォルト値 - なし
- データ型 - 文字列

Amazon Redshift Serverless アカウント ID。

このパラメータはオプションです。

serverless_work_group

- デフォルト値 - なし
- データ型 - 文字列

Amazon Redshift Serverless ワークグループ名。

このパラメータはオプションです。

ssl

- デフォルト値 – true
- データ型 – ブール型

Secure Sockets Layer (SSL) が有効になっています。

このパラメータは必須です。

ssl_insecure

- デフォルト値 – false
- データ型 – ブール型

IdP ホストのサーバー SSL 証明書の検証を無効にするかどうかを指定する値。このパラメータを True に設定すると、IdP ホストのサーバー SSL 証明書の検証が無効になります。本番環境では、デフォルト値の False のままにしておくことを推奨します。

このパラメータはオプションです。

sslmode

- デフォルト値 – verify-ca
- データ型 - 文字列

Amazon Redshift への接続のセキュリティ。次の値のいずれかを指定できます。

- verify-ca
- verify-full

> **Note**
>
> 可能な限り、SSL 接続に `verify-full` を使用することをお勧めします。

このパラメータは必須です。

tcp_keepalive

- デフォルト値 – true
- データ型 – ブール型

TCP キープアライブを使用して接続がタイムアウトしないようにするかどうか。次の値を指定できます。

- True: ドライバーは、TCP キープアライブを使用して、接続がタイムアウトしないようにします。
- False: ドライバーは TCP キープアライブを使用しません。

このパラメータはオプションです。

tcp_keepalive_count

- デフォルト値 - なし
- データ型 – 整数

接続を非アクティブと見なす前に送信する認知されていないプローブ数。例えば、値を 3 に設定すると、ドライバーは接続がアクティブでないと判断する前に、未応答のキープアライブパケットを 3 つ送信します。

このパラメータを指定しない場合、Amazon Redshift はシステムのデフォルト値を使用します。

このパラメータはオプションです。

tcp_keepalive_interval

- デフォルト値 - なし
- データ型 – 整数

ドライバーがプローブの確認応答を受信しなかった場合の、後続のキープアライブプローブの秒単位の間隔。このパラメータを指定する場合、正の整数である必要があります。

このパラメータを指定しない場合、Amazon Redshift はシステムのデフォルト値を使用します。

このパラメータはオプションです。

tcp_keepalive_idle

- デフォルト値 - なし
- データ型 – 整数

ドライバーが最初のキープアライブプローブを送信した後の非アクティブ時間 (秒単位)。例えば、値を 120 に設定すると、ドライバーは最初のキープアライブパケットを送信する前に 2 分間の非アクティブ状態を維持します。このパラメータを指定する場合、正の整数である必要があります。

このパラメータを指定しない場合、Amazon Redshift はシステムのデフォルト値を使用します。

このパラメータはオプションです。

タイムアウト

- デフォルト値 - なし
- データ型 – 整数

サーバーへの接続がタイムアウトするまで待つ秒数。

このパラメータはオプションです。

トークン

- デフォルト値 - なし
- データ型 - 文字列

AWS IAM アイデンティティセンターが提供するアクセストークン、または AWS IAM アイデンティティセンターにリンクされているウェブ ID プロバイダーが提供する OpenID Connect (OIDC) JSON ウェブトークン (JWT)。アプリケーションは、このトークンを生成するために、AWS IAM アイデンティティセンターまたは AWS IAM アイデンティティセンターにリンクされている ID プロバイダーにより、アプリケーションのユーザーを認証する必要があります。

このパラメータは `IdpTokenAuthPlugin` で機能します。

token_type

- デフォルト値 - なし
- データ型 - 文字列

`IdpTokenAuthPlugin` で使用しているトークンのタイプ。

次の値を指定できます。

ACCESS_TOKEN

AWS IAM アイデンティティセンターが提供するアクセストークンを使用する場合は、これを入力します。

EXT_JWT

AWS IAM アイデンティティセンターと統合されているウェブベースの ID プロバイダーが提供する OpenID Connect (OIDC) JSON ウェブトークン (JWT) を使用する場合は、これを入力します。

このパラメータは IdpTokenAuthPlugin で機能します。

ユーザー

- デフォルト値 - なし
- データ型 - 文字列

認証に使用するユーザー名。

このパラメータはオプションです。

web_identity_token

- デフォルト値 - なし
- データ型 - 文字列

ID プロバイダによって提供される OAuth 2.0 アクセストークンまたは OpenID Connect ID トークン。アプリケーションを使用しているユーザーをウェブアイデンティティプロバイダーで認証することで、アプリケーションがこのトークンを取得するようにします。JwtCredentialsProviderで示されるプロバイダーはこのパラメータを使用します。

JwtCredentialsProviderプロバイダーの場合、このパラメータは必須です。それ以外の場合、このパラメータはオプションです。

Python コネクタのインポート

Python コネクタをインポートするには、次のコマンドを実行します。

```
>>> import redshift_connector
```

AWS認証情報を使用して Amazon Redshift クラスターに接続するには、以下のコマンドを実行します。

```
conn = redshift_connector.connect(
    host='examplecluster.abc123xyz789.us-west-1.redshift.amazonaws.com',
    port=5439,
    database='dev',
    user='awsuser',
    password='my_password'
 )
```

Python コネクタと NumPy の連携

以下は、Python コネクタと NumPy を連携する例です。

```
>>>  import numpy
#Connect to the cluster
>>> import redshift_connector
>>> conn = redshift_connector.connect(
     host='examplecluster.abc123xyz789.us-west-1.redshift.amazonaws.com',
     port=5439,
     database='dev',
     user='awsuser',
     password='my_password'
  )

# Create a Cursor object
>>> cursor = conn.cursor()

# Query and receive result set
cursor.execute("select * from book")

result: numpy.ndarray = cursor.fetch_numpy_array()
print(result)
```

結果は、以下のとおりです。

```
[['One Hundred Years of Solitude' 'Gabriel García Márquez']
['A Brief History of Time' 'Stephen Hawking']]
```

Python コネクタとpandasの連携

以下は、Python コネクタとpandasを統合する例です。

```
>>> import pandas
```

```
#Connect to the cluster
>>> import redshift_connector
>>> conn = redshift_connector.connect(
     host='examplecluster.abc123xyz789.us-west-1.redshift.amazonaws.com',
     port=5439,
     database='dev',
     user='awsuser',
     password='my_password'
  )

# Create a Cursor object
>>> cursor = conn.cursor()

# Query and receive result set
cursor.execute("select * from book")
result: pandas.DataFrame = cursor.fetch_dataframe()
print(result)
```

アイデンティティプロバイダープラグインの使用

ID プロバイダープラグインの使用方法については、「IAM 認証情報の提供オプション」を参照してください。IAM ロールのベストプラクティスを含む IAM ID の管理の詳細については、「Amazon Redshift での Identity and Access Management」を参照してください。

ADFS アイデンティティプロバイダープラグインを使用した認証

次に、アクティブディレクトリフェデレーションサービス (ADFS) アイデンティティプロバイダープラグインを使用して、Amazon Redshift データベースに接続するユーザーを認証する例を示します。

```
>>> con = redshift_connector.connect(
    iam=True,
    database='dev',
    host='my-testing-cluster.abc.us-east-2.redshift.amazonaws.com',
    cluster_identifier='my-testing-cluster',
    credentials_provider='AdfsCredentialsProvider',
    user='brooke@myadfshostname.com',
    password='Hunter2',
    idp_host='myadfshostname.com'
)
```

Azure アイデンティティプロバイダープラグインを使用した認証

以下は、Azure のアイデンティティプロバイダプラグインを使用した認証の例です。以下のように、Azure エンタープライズアプリケーションのclient_idおよびclient_secretの値を作成することができます。

```
>>>  con = redshift_connector.connect(
    iam=True,
    database='dev',
    host='my-testing-cluster.abc.us-east-2.redshift.amazonaws.com',
    cluster_identifier='my-testing-cluster',
    credentials_provider='AzureCredentialsProvider',
    user='brooke@myazure.org',
    password='Hunter2',
    idp_tenant='my_idp_tenant',
    client_id='my_client_id',
    client_secret='my_client_secret',
    preferred_role='arn:aws:iam:123:role/DataScientist'
)
```

AWS IAM アイデンティティセンターのプロバイダープラグインを使用した認証

以下は、AWS IAM アイデンティティセンターのプロバイダープラグインを使用した認証の例です。

```
with redshift_connector.connect(
credentials_provider='BrowserIdcAuthPlugin',
host='my-testing-cluster.abc.us-east-2.redshift.amazonaws.com',
database='dev',
idc_region='us-east-1',
issuer_url='https://identitycenter.amazonaws.com/ssoins-790723ebe09c86f9',
idp_response_timeout=60,
listen_port=8100,
idc_client_display_name='Test Display Name',
# port value of 5439 is specified by default
)
```

Azure ブラウザのアイデンティティプロバイダープラグインを使用した認証

次に、Azure ブラウザアイデンティティプロバイダープラグインを使用して、Amazon Redshift データベースに接続するユーザーを認証する例を示します。

多要素認証は、ユーザーから提供されたサインイン認証情報を使用して、ブラウザで行われます。

```
>>>con = redshift_connector.connect(
    iam=True,
    database='dev',
    host='my-testing-cluster.abc.us-east-2.redshift.amazonaws.com',
    cluster_identifier='my-testing-cluster',
    credentials_provider='BrowserAzureCredentialsProvider',
    idp_tenant='my_idp_tenant',
    client_id='my_client_id',
)
```

Okta アイデンティティプロバイダープラグインを使用した認証

以下は、Oktaのアイデンティティプロバイダプラグインを使用した認証の例です。Okta アプリケーションを使用してidp_host、app_idおよびapp_nameの値を取得できます。

```
>>> con = redshift_connector.connect(
    iam=True,
    database='dev',
    host='my-testing-cluster.abc.us-east-2.redshift.amazonaws.com',
    cluster_identifier='my-testing-cluster',
    credentials_provider='OktaCredentialsProvider',
    user='brooke@myazure.org',
    password='hunter2',
    idp_host='my_idp_host',
    app_id='my_first_appetizer',
    app_name='dinner_party'
)
```

汎用 SAML ブラウザ アイデンティティプロバイダープラグインで JumpCloud を使用した認証

以下に、認証用の汎用 SAML ブラウザ アイデンティティプロバイダープラグインで JumpCloud を使用する例を示します。

パスワードパラメータは必須です。ただし、ブラウザで多要素認証が行われるため、このパラメータを入力する必要はありません。

```
>>> con = redshift_connector.connect(
    iam=True,
    database='dev',
    host='my-testing-cluster.abc.us-east-2.redshift.amazonaws.com',
    cluster_identifier='my-testing-cluster',
```

```
    credentials_provider='BrowserSamlCredentialsProvider',
    user='brooke@myjumpcloud.org',
    password='',
    login_url='https://sso.jumpcloud.com/saml2/plustwo_melody'
)
```

Amazon Redshift Python コネクタの使用例

Amazon Redshift Python コネクタの使用例を次に示します。これを実行するには、まず Python コネクタをインストールする必要があります。Amazon Redshift Python コネクタのインストールの詳細については、「Amazon Redshift Python コネクタのインストール」を参照してください。Python コネクタで使用できる設定オプションの詳細については、「Amazon Redshift Python コネクタの設定オプション」を参照してください。

トピック

- AWS 認証情報を使用した Amazon Redshift クラスターへの接続およびクエリ
- オートコミットの有効化
- カーソルのパラメータスタイルの設定
- Amazon S3 バケットからデータをコピーする COPY と Amazon S3 バケットへデータを書き込む UNLOAD の使用

AWS 認証情報を使用した Amazon Redshift クラスターへの接続およびクエリ

以下の例は、AWS 認証情報を使用して Amazon Redshift クラスターに接続し、次にテーブルをクエリしてクエリ結果を取得する手順を示しています。

```
#Connect to the cluster
>>> import redshift_connector
>>> conn = redshift_connector.connect(
     host='examplecluster.abc123xyz789.us-west-1.redshift.amazonaws.com',
     database='dev',
     port=5439,
     user='awsuser',
     password='my_password'
  )

# Create a Cursor object
>>> cursor = conn.cursor()

# Query a table using the Cursor
```

```
>>> cursor.execute("select * from book")

#Retrieve the query result set
>>> result: tuple = cursor.fetchall()
>>> print(result)
 >> (['One Hundred Years of Solitude', 'Gabriel García Márquez'], ['A Brief History of
 Time', 'Stephen Hawking'])
```

オートコミットの有効化

オートコミットプロパティは、Python データベース API 仕様に従って、デフォルトではオフになっています。ロールバックコマンドの実行後に以下のコマンドを使用して、接続の自動コミットプロパティをオンにして、トランザクションが進行中でないことを確認できます。

```
#Connect to the cluster
>>> import redshift_connector
>>> conn = redshift_connector.connect(...)

# Run a rollback command
>>>  conn.rollback()

# Turn on autocommit
>>>  conn.autocommit = True
>>>  conn.run("VACUUM")

# Turn off autocommit
>>>  conn.autocommit = False
```

カーソルのパラメータスタイルの設定

カーソルのパラメータスタイルは cursor.paramstyle で変更できます。デフォルトとして使用されるパラメータスタイルは `format` です。パラメータスタイルの有効な値は、`qmark`、`numeric`、`named`、`format`、および `pyformat` です。

さまざまなパラメータスタイルを使用してパラメータをサンプル SQL ステートメントに渡す例を以下に示します。

```
# qmark
redshift_connector.paramstyle = 'qmark'
sql = 'insert into foo(bar, jar) VALUES(?, ?)'
cursor.execute(sql, (1, "hello world"))
```

```
# numeric
redshift_connector.paramstyle = 'numeric'
sql = 'insert into foo(bar, jar) VALUES(:1, :2)'
cursor.execute(sql, (1, "hello world"))

# named
redshift_connector.paramstyle = 'named'
sql = 'insert into foo(bar, jar) VALUES(:p1, :p2)'
cursor.execute(sql, {"p1":1, "p2":"hello world"})

# format
redshift_connector.paramstyle = 'format'
sql = 'insert into foo(bar, jar) VALUES(%s, %s)'
cursor.execute(sql, (1, "hello world"))

# pyformat
redshift_connector.paramstyle = 'pyformat'
sql = 'insert into foo(bar, jar) VALUES(%(bar)s, %(jar)s)'
cursor.execute(sql, {"bar": 1, "jar": "hello world"})
```

Amazon S3 バケットからデータをコピーする COPY と Amazon S3 バケットへデータを書き込む UNLOAD の使用

以下の例は、Amazon S3 バケットからテーブルにデータをコピーしてから、テーブルからバケットにアンロードする方法を示しています。

以下のデータを含む category_csv.txt という名前のテキストファイルをAmazon S3 バケットにアップロードします。

```
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"
```

次に、最初に Amazon Redshift データベースに接続する Python コードの例を示します。次に、categoryという名前のテーブルを作成し、S3 バケットから CSV データをテーブルにコピーします。

```
#Connect to the cluster and create a Cursor
>>> import redshift_connector
>>> with redshift_connector.connect(...) as conn:
```

```
>>> with conn.cursor() as cursor:

#Create an empty table
>>>     cursor.execute("create table category (catid int, cargroup varchar, catname
 varchar, catdesc varchar)")

#Use COPY to copy the contents of the S3 bucket into the empty table
>>>     cursor.execute("copy category from 's3://testing/category_csv.txt' iam_role
 'arn:aws:iam::123:role/RedshiftCopyUnload' csv;")

#Retrieve the contents of the table
>>>     cursor.execute("select * from category")
>>>     print(cursor.fetchall())

#Use UNLOAD to copy the contents of the table into the S3 bucket
>>>     cursor.execute("unload ('select * from category') to 's3://testing/
unloaded_category_csv.txt'  iam_role 'arn:aws:iam::123:role/RedshiftCopyUnload' csv;")

#Retrieve the contents of the bucket
>>>     print(cursor.fetchall())
 >> ([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'])
```

autocommit を true に設定していない場合は、execute() ステートメントの実行後に conn.commit() でコミットします。

データは S3 バケット内にあるファイル unloaded_category_csv.text0000_part00 にアンロードされ、次のコンテンツが含まれます。

```
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"
```

Amazon Redshift Python コネクタの API リファレンス

Amazon Redshift Python コネクタ API オペレーションの説明は次のとおりです。

redshift_connector

redshift_connectorAPI オペレーションの説明は次のとおりです。

`connect(user, database, password[, port, …])`

Amazon Redshift クラスターへの接続を確立します。この関数は、ユーザー入力を検証し、必要に応じて アイデンティティプロバイダープラグインを使用して認証し、接続オブジェクトを作成します。

`apilevel`

現在サポートされている DBAPI レベルは「2.0」です。

`paramstyle, str(object='') -> str str(bytes_or_buffer[, encoding[, errors]]) -> str`

グローバルに使用するデータベース API パラメータスタイル。

接続

Amazon Redshift Python コネクタの接続 API オペレーションの説明は次のとおりです。

`__init__(user, password, database[, host, …])`

生の接続オブジェクトを初期化します。

`cursor`

この接続にバインドされたカーソルオブジェクトを作成します。

`commit`

現在のデータベーストランザクションをコミットします。

`rollback`

現在のデータベーストランザクションをロールバックします。

`close`

データベース接続を閉じます。

`execute(cursor, operation, vals)`

指定された SQL コマンドを実行します。`redshift_connector.paramstyle`の値に応じて、パラメータをシーケンスまたはマッピングとして提供することができます。

`run(sql[, stream])`

指定された SQL コマンドを実行します。オプションで、COPY コマンドで使用するストリーミングを指定することができます。

`xid(format_id, global_transaction_id, …)`

トランザクション ID を作成します。`global_transaction_id`パラメータのみが postgres で使用されます。format_id と branch_qualifier は postgres では使用されません。`global_transaction_id`は、タプル(`format_id`、`global_transaction_id`、`branch_qualifier`)を返す postgres がサポートする任意の文字列識別子とすることができます。

`tpc_begin(xid)`

フォーマット ID、グローバルトランザクション ID、ブランチクオリファイアで構成されるトランザクション ID `xid`で、TPC トランザクションを開始します。

`tpc_prepare`

.tpc_begin で開始されたトランザクションの最初のフェーズを実行します。

`tpc_commit([xid])`

引数なしで呼び出された場合、.tpc_commit は、以前に.tpc_prepare () で準備された TPC トランザクションをコミットします。

`tpc_rollback([xid])`

引数なしで呼び出されると、.tpc_rollback は TPC トランザクションをロールバックします。

`tpc_recover`

.tpc_commit (xid) または.tpc_rollback (xid) での使用に適した保留中のトランザクション ID のリストを返します。

カーソル

カーソル API オペレーションの説明は次のとおりです。

`__init__(connection[, paramstyle])`

生のカーソルオブジェクトを初期化します。

`insert_data_bulk(filename, table_name, parameter_indices, column_names, delimiter, batch_size)`

一括 INSERT ステートメントを実行します。

`execute(operation[, args, stream, …])`

データベースオペレーションを実行します。

`executemany(operation, param_sets)`

データベースオペレーションを準備し、指定されたすべてのパラメータシーケンスまたはマッピングに対して実行します。

`fetchone`

クエリの結果セットの次の行を取得します。

`fetchmany([num])`

クエリ結果の次の行のセットを取得します。

`fetchall`

クエリ結果の残りのすべての行を取得します。

`close`

今すぐカーソルを閉じます。

`__iter__`

カーソルオブジェクトを反復処理して、クエリから行を取得できます。

`fetch_dataframe([num])`

最後のクエリ結果のデータフレームを返します。

`write_dataframe(df, table)`

同じ構造データフレームを Amazon Redshift データベースに書き込みます。

`fetch_numpy_array([num])`

最後のクエリ結果の NumPy 配列を返します。

`get_catalogs`

Amazon Redshift は、1 つの接続で複数のカタログをサポートしていません。Amazon Redshift は現在のカタログのみを返します。

`get_tables([catalog, schema_pattern, …])`

システム内でユーザー定義された固有のパブリックテーブルを返します。

`get_columns([catalog, schema_pattern, …])`

Amazon Redshift データベース内の特定のテーブルのすべての列のリストを返します。

AdfsCredentials プロバイダープラグイン

Amazon Redshift Python コネクタの AdfsCredentials プロバイダープラグイン API オペレーションの構文を次に示します。

```
redshift_connector.plugin.AdfsCredentialsProvider()
```

AzureCredentials プロバイダープラグイン

Amazon Redshift Python コネクタの AzureCredentials プロバイダープラグイン API オペレーションの構文を次に示します。

```
redshift_connector.plugin.AzureCredentialsProvider()
```

ブラウザ AzureCredentials プロバイダープラグイン

Amazon Redshift Python コネクタのBrowserAzureCredentials プロバイダープラグイン API オペレーションの構文を以下に示します。

```
redshift_connector.plugin.BrowserAzureCredentialsProvider()
```

BrowserSamlCredentials プロバイダープラグイン

Amazon Redshift Python コネクタの BrowserSamlCredentials プロバイダープラグイン API オペレーションの構文は以下のとおりです。

```
redshift_connector.plugin.BrowserSamlCredentialsProvider()
```

Oktacredentials プロバイダープラグイン

Amazon Redshift Python コネクタの OktaCredentials プロバイダープラグイン API オペレーションの構文を次に示します。

```
redshift_connector.plugin.OktaCredentialsProvider()
```

PingCredentials プロバイダープラグイン

Amazon Redshift Python コネクタの PingCredentials プロバイダー プラグイン API オペレーションの構文を次に示します。

```
redshift_connector.plugin.PingCredentialsProvider()
```

SamlCredentials プロバイダープラグイン

Amazon Redshift Python コネクタの SamlCredentials プロバイダープラグイン API オペレーションの構文を次に示します。

```
redshift_connector.plugin.SamlCredentialsProvider()
```

Apache Spark 用の Amazon Redshift の統合

Apache Spark は、機械学習、ストリーム処理、グラフ分析に役立つ分散処理フレームワークおよびプログラミングモデルです。Apache Spark は、Apache Hadoop と同様に、ビッグデータのワークロードを処理するために一般的に使用されているオープンソースの分散処理システムです。Spark には最適化された有向非循環グラフ (DAG) 実行エンジンが搭載されており、データをメモリ内に積極的にキャッシュします。これにより、特に特定のアルゴリズムやインタラクティブクエリのパフォーマンスを向上させることができます。

この統合により、Amazon Redshift と Amazon Redshift Serverless のデータを読み書きする Apache Spark アプリケーションを構築するために使用できる Spark コネクタが提供されます。これらのアプリケーションは、アプリケーションのパフォーマンスやデータのトランザクションの一貫性を損なうことはありません。この統合は自動的に Amazon EMR および AWS Glue に含まれるため、データインジェストと変換パイプラインの一部として Amazon Redshift のデータにアクセスしてロードする Apache Spark ジョブをすぐに実行できます。

現在、この統合では Spark のバージョン 3.3.x、3.4.x、3.5.x、4.0.x、および 4.1.x を使用できます。

この統合により、次のことが可能になります。

- AWS Identity and Access Management (IAM) 認証。詳細については、「Amazon Redshift での Identity and Access Management」を参照してください。
- パフォーマンスを向上させるための述語とクエリのプッシュダウン
- Amazon Redshift のデータ型
- Amazon Redshift および Amazon Redshift Serverless への接続

Spark コネクタを使用する際の考慮事項と制限事項

- tempdir URI は Amazon S3 の場所を指します。この一時ディレクトリは自動的にはクリーンアップされないため、追加コストが発生する可能性があります。Amazon Simple Storage Service ユーザーガイドの Amazon S3 ライフサイクルポリシーを使用して、Amaxon S3 バケットの保持ルールを定義することをお勧めします。
- デフォルトでは、S3 バケットと Redshift クラスターが異なる AWS リージョンにある場合、Amazon S3 と Redshift 間のコピーは機能しません。個別の AWS リージョンを使用するには、`tempdir_region` パラメータを、`tempdir` に使用する S3 バケットのリージョンに設定します。
- `tempformat` パラメータを使用して Parquet データを書き込む場合、S3 と Redshift の間でクロスリージョンの書き込みが行われます。
- Amazon S3 サーバー側の暗号化を使用して、使用する Amazon S3 バケットを暗号化することをお勧めします。
- Amazon S3 バケットへのパブリックアクセスをブロックすることをお勧めします。
- Amazon Redshift クラスターがパブリックにアクセスできないようにすることをお勧めします。
- Amazon Redshift 監査ログ作成を有効にすることをお勧めします。
- Amazon Redshift 保管時の暗号化を有効にすることをお勧めします。
- Amazon EMR 上の Spark から Amazon Redshift への JDBC 接続に対して SSL を有効にすることをお勧めします。
- Amazon Redshift 認証パラメータのパラメータ `aws_iam_role` を使用して IAM ロールを渡すことをお勧めします。

Spark コネクタによる認証

次の図は、Amazon S3、Amazon Redshift、Spark ドライバー、および Spark エグゼキューター間の認証を示しています。

Redshift と Spark 間の認証

Amazon Redshift が提供する JDBC ドライバーバージョン 2.x を使用し、サインイン認証情報を指定して Spark コネクタで Amazon Redshift に接続できます。IAM を使用するには、IAM 認証を使用するように JDBC URL を設定します。Amazon EMR または AWS Glue から Redshift クラスターに接続するには、IAM ロールに一時的な IAM 認証情報を取得するために必要なアクセス権限があることを確認します。次のリストは、IAM ロールが認証情報を取得して Amazon S3 オペレーションを実行するために必要なすべての権限を示しています。

- Redshift:GetClusterCredentials (プロビジョニングされた Redshift クラスターの場合)
- Redshift:DescribeClusters (プロビジョニングされた Redshift クラスターの場合)
- Redshift:GetWorkgroup (Amazon Redshift Serverless ワークグループの場合)
- Redshift:GetCredentials (Amazon Redshift Serverless の場合、ワークグループ)
- s3:ListBucket
- s3:GetBucket
- s3:GetObject
- s3:PutObject
- s3:GetBucketLifecycleConfiguration

GetClusterCredentials の詳細については、「GetClusterCredentials のIAM ポリシー」を参照してください。

また、COPY および UNLOAD オペレーション中に Amazon Redshift が IAM ロールを引き継げるようにする必要があります。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "redshift.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
```

```
        }
    ]
}
```

最新の JDBC ドライバーを使用している場合、ドライバーは Amazon Redshift の自己署名証明書から ACM 証明書への移行を自動的に管理します。ただし、JDBC URL に SSL オプションを指定する必要があります。

以下は、JDBC ドライバーの URL と aws_iam_role を指定して Amazon Redshiftに接続する方法の例です。

```
df.write \
  .format("io.github.spark_redshift_community.spark.redshift ") \
  .option("url", "jdbc:redshift:iam://<the-rest-of-the-connection-string>") \
  .option("dbtable", "<your-table-name>") \
  .option("tempdir", "s3a://<your-bucket>/<your-directory-path>") \
  .option("aws_iam_role", "<your-aws-role-arn>") \
  .mode("error") \
  .save()
```

Amazon S3 と Spark 間の認証

IAM ロールを使用して Spark と Amazon S3 間での認証を行う場合は、以下のいずれかの方法を使用します。

- AWS SDK for Java は、DefaultAWSCredentialsProviderChain クラスによって実装されたデフォルトの認証情報プロバイダーチェーンを使用して、AWS 認証情報を自動的に見つけようとします。詳細については、「デフォルトの認証情報プロバイダチェーンの使用」を参照してください。
- Hadoop 設定プロパティで AWS キーを指定することができます。例えば、tempdir 設定が s3n:// ファイルシステムを指している場合、Hadoop XML 設定ファイルで fs.s3n.awsAccessKeyId および fs.s3n.awsSecretAccessKey プロパティを設定するか、sc.hadoopConfiguration.set() を呼び出して Spark のグローバル Hadoop 設定を変更します。

例えば、s3n ファイルシステムを使用している場合は、以下を追加します。

```
sc.hadoopConfiguration.set("fs.s3n.awsAccessKeyId", "YOUR_KEY_ID")
sc.hadoopConfiguration.set("fs.s3n.awsSecretAccessKey", "YOUR_SECRET_ACCESS_KEY")
```

s3a ファイルシステムの場合は、以下を追加します。

```
sc.hadoopConfiguration.set("fs.s3a.access.key", "YOUR_KEY_ID")
sc.hadoopConfiguration.set("fs.s3a.secret.key", "YOUR_SECRET_ACCESS_KEY")
```

Python を使用している場合は、以下のオペレーションを使用します。

```
sc._jsc.hadoopConfiguration().set("fs.s3n.awsAccessKeyId", "YOUR_KEY_ID")
sc._jsc.hadoopConfiguration().set("fs.s3n.awsSecretAccessKey",
 "YOUR_SECRET_ACCESS_KEY")
```

- `tempdir` URL で認証キーをエンコードします。例えば、URI `s3n://ACCESSKEY:SECRETKEY@bucket/path/to/temp/dir` では キーペア (ACCESSKEY、SECRETKEY) をエンコードします。

Redshift と Amazon S3 間の認証

クエリで COPY コマンドと UNLOAD コマンドを使用している場合は、ユーザーに代わってクエリを実行するために Amazon S3 に Amazon Redshift へのアクセス権を付与する必要もあります。そのためには、まず Amazon Redshift が他の AWS のサービスにアクセスすることを許可し、次に IAM ロールを使用して COPY オペレーションと UNLOAD オペレーションを許可します。

ベストプラクティスとして、アクセス許可ポリシーを IAM ロールにアタッチし、それを必要に応じてユーザーやグループに割り当てることをお勧めします。詳細については、「Amazon Redshift での Identity and Access Management」を参照してください。

AWS Secrets Manager との統合

Redshift のユーザー名とパスワードの認証情報は、AWS Secrets Manager に保存されているシークレットから取得できます。Redshift 認証情報を自動的に提供するには、`secret.id` パラメータを使用します。Redshift 認証情報シークレットを作成する方法の詳細については、「AWS Secrets Manager データベースシークレットを作成する」を参照してください。

GroupID	ArtifactID	サポート対象リビジョン	説明
com.amazonaws.secretsmanager	aws-secretsmanager-jdbc	1.0.12	Java 用 AWS Secrets Manager SQL Connection Library

GroupID	ArtifactID	サポート対象リビジョン	説明
			を使用すると、Java デベロッパーは AWS Secrets Manager に保存されているシークレットを使用して、簡単に SQL データベースに接続できます。

> **Note**
>
> 謝辞: このドキュメントには、Apache 2.0 ライセンスの下でライセンス提供された Apache Software Foundation によって開発されたサンプルコードと説明が含まれています。

プッシュダウンによるパフォーマンスの向上

Spark コネクタは、述語とクエリのプッシュダウンを自動的に適用してパフォーマンスを最適化します。このサポートは、クエリでサポートされている関数を使用している場合、Spark コネクタがその関数を SQL クエリに変換し、Amazon Redshift でそのクエリを実行することを意味します。この最適化により、取得されるデータが少なくなるため、Apache Spark が処理するデータが少なくなり、パフォーマンスが向上します。デフォルトでは、プッシュダウンは自動的に有効になります。非アクティブにするには、autopushdown を false に設定します。

```
import sqlContext.implicits._val
 sample= sqlContext.read
    .format("io.github.spark_redshift_community.spark.redshift")
    .option("url",jdbcURL )
    .option("tempdir", tempS3Dir)
    .option("dbtable", "event")
    .option("autopushdown", "false")
    .load()
```

プッシュダウンでは以下の機能がサポートされています。このリストにない関数を使用している場合、Spark コネクタは Amazon Redshift ではなく Spark でその関数を実行するため、パフォーマン

スが最適化されません。Spark の関数の一覧については、「Built-in Functions」(組み込み関数) を参照してください。

- 集計関数
  - avg
  - count
  - max
  - min
  - sum
  - stddev_samp
  - stddev_pop
  - var_samp
  - var_pop
- ブール演算子
  - in
  - isnull
  - isnotnull
  - contains
  - endswith
  - startswith
- 論理演算子
  - and
  - or
  - not (または !)
- 数学関数
  - +
  - -
  - *
  - /
  - - (単項)
  - abs

- acos
- asin
- atan
- ceil
- cos
- exp
- floor
- greatest
- least
- log10
- pi
- pow
- round
- sin
- sqrt
- tan

- その他の関数
  - cast
  - coalesce
  - decimal
  - if
  - 情報
- リレーショナル演算子
  - !=
  - =
  - >
  - >=
  - <
  - <=

- 文字列関数

- ascii
- lpad
- rpad
- translate
- upper
- lower
- length
- trim
- ltrim
- rtrim
- like
- substring
- concat

- 日付および時刻関数
  - add_months
  - date
  - date_add
  - date_sub
  - date_trunc
  - timestamp
  - trunc
- 数学オペレーション
  - CheckOverflow
  - PromotePrecision
- リレーショナルオペレーション
  - Aliases (例、AS)
  - CaseWhen
  - Distinct
  - InSet

  - 結合とクロス結合

- Limits
- Unions、union all
- ScalarSubquery
- Sorts (ascending および descending)
- UnscaledValue

その他の設定オプション

このページでは、Amazon Redshift Spark コネクタのために指定できるオプションを説明しています。

文字列列の最大サイズ

Redshift は、テーブルの作成時に文字列列をテキスト列として作成し、VARCHAR (256) として格納します。より大きなサイズをサポートする列が必要な場合は、maxlength を使用して文字列列の最大長を指定できます。maxlength を指定する方法の例を次に示します。

```
columnLengthMap.foreach { case (colName, length) =>
  val metadata = new MetadataBuilder().putLong("maxlength", length).build()
  df = df.withColumn(colName, df(colName).as(colName, metadata))
}
```

列のタイプ

列タイプを設定するには、redshift_type フィールドを使用します。

```
columnTypeMap.foreach { case (colName, colType) =>
  val metadata = new MetadataBuilder().putString("redshift_type", colType).build()
  df = df.withColumn(colName, df(colName).as(colName, metadata))
}
```

列の圧縮エンコード

列に特定の圧縮エンコードを使用するには、エンコードフィールドを使用します。サポートされている圧縮エンコードの詳細なリストについては、「圧縮エンコード」を参照してください。

列の説明

説明を設定するには、description フィールドを使用します。

Redshift と Amazon S3 間の認証

デフォルトでは、結果は Parquetet 形式で Amazon S3 にアンロードされます。結果をパイプ区切りのテキストファイルとしてアンロードするには、次のオプションを指定します。

```
.option("unload_s3_format", "TEXT")
```

プッシュダウンステートメント

[Parameter] (パラメータ)	必須	デフォルト	説明
spark.datasource.redshift.community.autopushdown.lazyMode	なし	正	コネクタがプッシュダウンステートメント Redshift を遅延実行するかどうかを指定します。 true の場合、Spark コネクタはクエリを実行する前に関連するモデルと情報をすべて取得するため、一般的にパフォーマンスが向上します。 false の場合、Spark コネクタはメインの Spark ドライバースレッドでプッシュダウンステートメントを直ちに実行し、式間でシリアル化されます。

コネクタパラメータ

パラメータマップまたは Spark SQL の `OPTIONS` は次の設定をサポートしています。

パラメータ	必須	デフォルト	説明
dbtable	はい (query が指定されていない限り)	該当なし	Redshift で作成したり読み込んだりするテーブル。データを Redshift に戻して保存する場合、このパラメータは必須です。
query	はい (dbtable が指定されていない限り)	該当なし	Redshift で読み込むクエリ。
user	いいえ	該当なし	Redshift の username。password パラメータと一緒に使用する必要があります。ユーザーとパスワードが URL のパラメータではない場合にのみ有効です。両方使用すると、エラーが発生します。
password	いいえ	該当なし	Redshift の password。user パラメータと一緒に使用する必要があります。ユーザーとパスワードが URL のパラメータではない場合にのみ有効です。両方使用すると、エラーが発生します。

パラメータ	必須	デフォルト	説明
url	いいえ	該当なし	JDBC URL。書式は jdbc:subprotocol://host:port/database?user=username&password=password です。 subprotocol は、ロードした JDBC ドライバーに応じて、postgresql または Redshift にすることができます。1 つの Redshift 互換ドライバーがクラスパスに含まれていて、この URL と一致する必要があることに注意してください。 ホストとポートは Redshift マスターノードを指す必要があるため、ドライバーアプリケーションからのアクセスを許可するようにセキュリティグループや VPC を設定する必要があります。 database は Redshift のデータベース名です。

パラメータ	必須	デフォルト	説明
			user および password はデータベースにアクセスするための認証情報で、この JDBC の URL に埋め込む必要があります。また、データベースユーザーにはテーブルにアクセスするための許可が必要です。
aws_iam_role	IAM ロールを使用して Redshift の COPY/UNLOAD オペレーションを許可する場合のみ	該当なし	Redshift クラスターにアタッチされた IAM ロールの完全に指定された ARN。

パラメータ	必須	デフォルト	説明
forward_spark_s3_credentials	いいえ	誤	Spark が Amazon S3 への接続に使用する認証情報をこのライブラリが自動的に検出するかどうか、またそれらの認証情報を JDBC ドライバー経由で Redshift に転送するかどうかを示します。これらの認証情報は JDBC クエリの一部として送信されます。そのため、このオプションを使用する場合は、JDBC 接続による SSL 暗号化を有効にすることをお勧めします。
temporary_aws_access_key_id	いいえ	該当なし	AWS アクセスキー S3 バケットへの書き込み権限が必要です。
temporary_aws_secret_access_key	いいえ	該当なし	AWSアクセスキーに対応する シークレットアクセスキーです。
temporary_aws_session_token	いいえ	該当なし	AWS提供されたアクセスキーに対応するセッショントークン。

パラメータ	必須	デフォルト	説明
tempdir	いいえ	該当なし	Amazon S3 の書き込み可能な場所。読み取り時にデータをアンロードし、書き込み時に Avro データを Redshift にロードするために使用されます。通常の ETL パイプラインの一部として Spark 用の Redshift データソースを使用している場合は、バケットにライフサイクルポリシーを設定して、それをこのデータの一時的な場所として使用すると便利です。
jdbcdriver	いいえ	JDBC URL のサブプロトコルによって決定されます	使用する JDBC ドライバーのクラス名。このクラスはクラスパス上にある必要があります。ほとんどの場合、適切なドライバークラス名は JDBC URL のサブプロトコルによって自動的に決定されるため、このオプションを指定する必要はありません。

パラメータ	必須	デフォルト	説明
diststyle	いいえ	Even	テーブルを作成するときに使用する Redshift 分散スタイル。有効なオプションは、EVEN、KEY、または ALL です。KEY を使用する場合は、distkey オプションを使用して分散キーも設定する必要があります。
distkey	いいえ (DISTSTYLE_KEY を使用する場合を除き)	該当なし	テーブル作成時に分散キーとして使用するテーブルの列の名前。
sortkeyspec	いいえ	該当なし	完全な Redshift ソートキーの定義。
include_column_list	いいえ	誤	このライブラリがスキーマから列を自動的に抽出し、列のマッピングオプションに従って COPY コマンドに追加するかどうかを示します。

パラメータ	必須	デフォルト	説明
description	いいえ	該当なし	テーブルの説明。説明は SQL COMMENT コマンドで設定され、ほとんどのクエリツールに表示されます。個々の列の説明を設定するには、description メタデータを参照してください。

パラメータ	必須	デフォルト	説明
preactions	いいえ	該当なし	COPY コマンドをロードする前に実行する SQL コマンドのセミコロン区切りリスト。新しいデータをロードする前に DELETE コマンドなどを実行すると便利な場合があります。コマンドに %s が含まれている場合、テーブル名は実行前にフォーマットされます (ステージングテーブルを使用している場合)。このコマンドが失敗した場合、例外として処理されます。ステージングテーブルを使用している場合、preactions が失敗した場合は変更が元に戻され、バックアップテーブルが復元されます。

パラメータ	必須	デフォルト	説明
extracopyoptions	いいえ	該当なし	データをロードするときに Redshift COPY コマンドに追加する追加オプションのリスト (TRUNCATECOLUMNS や MAXERROR n など)。使用可能なパラメータの詳細なリストについては、「任意指定のパラメータ」を参照してください。 これらのオプションは COPY コマンドの最後に追加されるため、コマンドの最後で意味のあるオプションのみ使用できることに注意してください。これでほとんどのユースケースが力バーされるはずです。

パラメータ	必須	デフォルト	説明
sse_kms_key	いいえ	該当なし	AWS デフォルトの暗号化ではなく、Redshift UNLOAD オペレーションに S3 でサーバー側の暗号化に使用する AWS KMS キー ID。Redshift IAM ロールには、KMS キーを使用して書き込みを行うために KMS キーへのアクセス権が必要です。Spark IAM ロールには、読み取りオペレーションのためにキーへのアクセス権が必要です。Spark の IAM ロールが適切なアクセス権を持っている限り、暗号化されたデータの読み取りに変更は必要ありません (AWS がこれを処理します)。

パラメータ	必須	デフォルト	説明
tempformat	いいえ	AVRO	Redshift に書き込むときに Amazon S3 に一時ファイルを保存する形式です。有効な値は AVRO、CSV、CSV GZIP (圧縮 CSV)、および PARQUET です。
csvnullstring (実験的)	いいえ	Null	CSV tempformat を使用するときに null に書き込む文字列値。これは、実際のデータには表示されない値である必要があります。
autopushdown	なし	正	SQL オペレーションの Spark 論理プランをキャプチャして分析することにより、述語とクエリのプッシュダウンを適用するかどうかを示します。オペレーションは SQL クエリに変換され、Redshift で実行されてパフォーマンスが向上します。

パラメータ	必須	デフォルト	説明
autopushdown.s3_result_cache	いいえ	誤	クエリ SQL をキャッシュしてデータの Amazon S3 パスマッピングをメモリにアンロードします。これにより、同じクエリを同じ Spark セッションで再度実行する必要がなくなります。autopushdown がオンになっている場合にのみサポートされます。キャッシュされた結果には古い情報が含まれている可能性があるため、読み取りオペレーションと書き込みオペレーションを混在させる場合は、このパラメータを使用しないことをお勧めします。
unload_s3_format	いいえ	Parquet	クエリ結果をアンロードする形式。有効なオプションは Parquet と Text で、クエリ結果をパイプで区切られたテキスト形式でアンロードするように指定します。

パラメータ	必須	デフォルト	説明
extraunloadoptions	いいえ	該当なし	Redshift の UNLOAD コマンドに追加する追加オプション。一部のオプションはコネクタ内で設定された他のオプションと競合する場合があるため、すべてのオプションの動作が保証されているわけではありません。
copydelay	いいえ	30000	Redshift COPY オペレーションの再試行間の遅延 (ミリ秒単位)。
copyretrycount	いいえ	2	Redshift COPY オペレーションを再試行する回数。

パラメータ	必須	デフォルト	説明
tempdir_region	いいえ	該当なし	`tempdir` が所在する AWS リージョン。このオプションを設定すると、`tempdir` とのやり取りにおけるコネクタのパフォーマンスが向上します。また、コネクタの読み取りおよび書き込みオペレーション中に COPY および UNLOAD オペレーションの一部としてこの値が自動的に指定されます。 この設定は以下の状況にあるときに推奨されます。 1) コネクタが AWS の外部で実行されているときに、自動リージョン検出が失敗して、コネクタのパフォーマンスに悪影響を及ぼす場合。 2) `tempdir` が、Redshift クラスターとは異なるリージョンにある場合、この設定を使用すると、`extracopy options` およ

パラメータ	必須	デフォルト	説明
			び extraunloadoptions パラメータを使用してリージョンを手動で指定する必要が軽減されます。PARQUET を tempformat として使用する場合、このパラメータを使用しても、tempdir を Redshift クラスターとは異なるリージョンに配置することはできません。 3) コネクタが tempdir とは異なるリージョンで実行されている場合、tempdir のコネクタのアクセスパフォーマンスが向上する場合。

パラメータ	必須	デフォルト	説明
secret.id	いいえ	該当なし	AWS Secrets Manager に保存されているシークレットの名前または ARN。このパラメータを使用して Redshift 認証情報を自動的に提供できますが、これが可能なのは、ユーザー、パスワード、および DbUser 認証情報が JDBC URL に渡されない、または他のオプションとして渡されない場合のみです。

パラメータ	必須	デフォルト	説明
secret.region	いいえ	該当なし	`secret.id` の値を検索するプライマリ AWS リージョン (米国東部 (バージニア北部) など)。 このリージョンを指定しない場合、コネクタはデフォルトの認証情報プロバイダーチェーンを使用して、`secret.id` のリージョン解決を試みます。場合によっては (コネクタをコネクタ外部で使用している場合など)、コネクタがリージョンを検出できないことがあります。以下の状況では、この設定を使用することをお勧めします。 1) コネクタが AWS 外部で動作している場合、自動リージョン検出は失敗し、Redshift による認証はできなくなります コネクタが `secret.id` とは異なるリージョンで実行されている場合

パラメータ	必須	デフォルト	説明
			(これにより、シークレットに対するコネクタのアクセスパフォーマンスが向上します)。
secret.vpcEndpointUrl	いいえ	該当なし	デフォルトの認証情報プロバイダーチェーンを上書きするときの、AWS Secrets Manager の PrivateLink DNS エンドポイント URL。
secret.vpcEndpointRegion	いいえ	該当なし	デフォルトの認証情報プロバイダーチェーンを上書きするときの、AWS Secrets Manager の PrivateLink DNS エンドポイント リージョン。

パラメータ	必須	デフォルト	説明
jdbc.*	いいえ	該当なし	基盤となる JDBC ドライバーに渡す追加のパラメータ。ここで、ワイルドカードは JDBC パラメータの名前です (jdbc.ssl など)。jdbc プレフィックスは JDBC ドライバーに渡される前に削除されることに注意してください。Redshift JDBC ドライバーで使用できるすべてのオプションを確認するには、「JDBC ドライバーバージョン 2.x の設定オプション」を参照してください。

パラメータ	必須	デフォルト	説明
ラベル	いいえ	""	コネクタでクエリを実行するときにクエリグループセットに含める識別子。100 文字以下で、すべての文字が有効な unicodeIdentifierParts である必要があります。ID が 100 文字を超える場合、超過分は削除されます。コネクタを使用してクエリを実行すると、クエリグループは次のような JSON 形式の文字列として設定されます `{"spark-redshift-connector":{"svc":"","ver":"5.1.0-amzn-1-spark_3.3","op":"Read","lbl":""}}` 。このオプションは lbl キーの値に置き換わります。

> **Note**
>
> 謝辞: このドキュメントには、Apache 2.0 ライセンスの下でライセンス提供された Apache Software Foundation によって開発されたサンプルコードと説明が含まれています。

サポートされているデータ型

Amazon Redshift の以下のデータ型は Spark コネクタでサポートされています。Amazon Redshift でサポートされているデータ型の完全なリストについては、「データ型」を参照してください。データ型が下の表にない場合、そのデータ型は Spark コネクタではサポートされていません。

データ型	エイリアス
SMALLINT	INT2
INTEGER	INT、INT4
BIGINT	INT8
DECIMAL	NUMERIC
REAL	FLOAT4
DOUBLE PRECISION	FLOAT8、FLOAT
BOOLEAN	BOOL
CHAR	CHARACTER、NCHAR、BPCHAR
VARCHAR	CHARACTER VARYING、NVARCHAR、TEXT
DATE	
TIMESTAMP	Timestamp without time zone
TIMESTAMPTZ	Timestamp with time zone
SUPER	

データ型	エイリアス
TIME	Time without time zone
TIMETZ	Time with time zone
VARBYTE	VARBINARY、BINARY VARYING

複雑なデータ型

Spark コネクタを使用して、Redshift の SUPER データ型列との間で、ArrayType、MapType、StructType のような Spark の複雑なデータ型を読み書きできます。読み取りオペレーションにスキーマを指定すると、列のデータは Spark 内の対応する複合型 (ネストされた型も含む) に変換されます。さらに、autopushdown を有効にすると、ネストされた属性、マップ値、配列インデックスの投影が Redshift にプッシュダウンされるため、データの一部だけにアクセスするときに、ネストされたデータ構造全体をアンロードする必要がなくなります。

コネクタから DataFrames を書き込む場合、MapType 型の列 (StringType を使用)、StructType、または ArrayType は、Redshift SUPER データ型列に書き込まれます。これらのネストされたデータ構造を記述する場合、tempformat パラメータは CSV、CSV GZIP、または PARQUET である必要があります。AVRO を使用すると、例外が発生します。また、StringType 以外のキータイプを持つ MapType データ構造を記述すると、例外が発生します。

StructType

次の例は、構造体を含む SUPER データ型のテーブルを作成する方法を示しています。

```
create table contains_super (a super);
```

その後、次の例のようなスキーマを使用して、テーブルの SUPER 列 a から StringType フィールド hello をクエリできます。

```
import org.apache.spark.sql.types._

val sc = // existing SparkContext
val sqlContext = new SQLContext(sc)

val schema = StructType(StructField("a", StructType(StructField("hello",
 StringType) ::Nil)) :: Nil)
```

```
val helloDF = sqlContext.read
.format("io.github.spark_redshift_community.spark.redshift")
.option("url", jdbcURL )
.option("tempdir", tempS3Dir)
.option("dbtable", "contains_super")
.schema(schema)
.load().selectExpr("a.hello")
```

次の例は、構造体を a 列に書き込む方法を示しています。

```
import org.apache.spark.sql.types._
import org.apache.spark.sql._

val sc = // existing SparkContext
val sqlContext = new SQLContext(sc)

val schema = StructType(StructField("a", StructType(StructField("hello",
 StringType) ::Nil)) :: Nil)
val data = sc.parallelize(Seq(Row(Row("world"))))
val mydf = sqlContext.createDataFrame(data, schema)

mydf.write.format("io.github.spark_redshift_community.spark.redshift").
option("url", jdbcUrl).
option("dbtable", tableName).
option("tempdir", tempS3Dir).
option("tempformat", "CSV").
mode(SaveMode.Append).save
```

MapType

MapType を使用してデータを表す場合は、スキーマ内の MapType データ構造を使用して、マップ内のキーに対応する値を取得します。MapType データ構造のすべてのキーは String 型でなければならず、すべての値は int のように同じ型でなければなりません。

次の例は、列 a のキー hello の値を取得する方法を示しています。

```
import org.apache.spark.sql.types._

val sc = // existing SparkContext
val sqlContext = new SQLContext(sc)
```

```
val schema = StructType(StructField("a", MapType(StringType, IntegerType))::Nil)

val helloDF = sqlContext.read
    .format("io.github.spark_redshift_community.spark.redshift")
    .option("url", jdbcURL )
    .option("tempdir", tempS3Dir)
    .option("dbtable", "contains_super")
    .schema(schema)
    .load().selectExpr("a['hello']")
```

ArrayType

列に構造体ではなく配列が含まれている場合は、コネクタを使用して配列の最初の要素をクエリできます。

```
import org.apache.spark.sql.types._

val sc = // existing SparkContext
val sqlContext = new SQLContext(sc)

val schema = StructType(StructField("a", ArrayType(IntegerType)):: Nil)

val helloDF = sqlContext.read
    .format("io.github.spark_redshift_community.spark.redshift")
    .option("url", jdbcURL )
    .option("tempdir", tempS3Dir)
    .option("dbtable", "contains_super")
    .schema(schema)
    .load().selectExpr("a[0]")
```

制限事項

Spark コネクタで複雑なデータ型を使用する場合は、次の制限があります。

- ネストされた構造体フィールド名とマップキーはすべて小文字でなければなりません。大文字を含む複雑なフィールド名をクエリする場合は、回避策として、スキーマを省略し、`from_json` spark 関数を使用して、返された文字列をローカルに変換するよう試みることができます。
- 読み取りまたは書き込みオペレーションで使用されるマップフィールドには、`StringType` キーのみが必要です。
- `CSV`、`CSV GZIP`、および `PARQUET` のみが、複合型を Redshift に書き込むためにサポートされる tempformat 値です。`AVRO` を使用しようとすると、例外が発生します。

Amazon Redshift での ODBC ドライバーのバージョン 2.x による接続の設定

ODBC 接続を使用すると、多くのサードパーティー SQL クライアントツールおよびアプリケーションから Amazon Redshift クラスターに接続できます。クライアントツールで JDBC がサポートされる場合、JDBC に用意された設定が簡単なため、ODBC ではなくその接続タイプを使用することを選択できます。クライアントツールで JDBC がサポートされていない場合は、このセクションの手順に従って、クライアントコンピュータまたは Amazon EC2 インスタンスで ODBC 接続を設定できます。

Amazon Redshift には、Linux、Windows、Mac オペレーティングシステム用の 64 ビット ODBC ドライバーが用意されています。32 ビット ODBC ドライバーは廃止されています。今後、緊急のセキュリティパッチを除き、32 ビット ODBC ドライバーのアップデートはリリースされません。

ODBC ドライバーの変更に関する最新情報については、「変更ログ」を参照してください。

トピック

- ODBC URL の取得
- Microsoft Windows での Amazon Redshift ODBC ドライバーの使用
- Linux での Amazon Redshift ODBC ドライバーの使用
- Apple macOS での Amazon Redshift ODBC ドライバーの使用
- 認証方法
- データ型変換
- ODBC ドライバーオプション
- 以前のバージョンの ODBC ドライバー
- ODBC ドライバーバージョン 2.x と 1.x の違い

ODBC URL の取得

Amazon Redshift では、Amazon Redshift コンソールに、クラスターの ODBC URL が表示されます。この URL には、クライアントコンピュータとデータベースの間の接続をセットアップするために必要な情報が含まれています。

ODBC の URL 形式を次に示します。

```
Driver={driver}; Server=endpoint_host; Database=database_name; UID=user_name;
 PWD=password; Port=port_number
```

前述の形式のフィールドには、次の値があります。

フィールド	値
Driver	使用する 64 ビット ODBC ドライバーの名前: [Amazon Redshift ODBC Driver (x64)] (Amazon Redshift ODBC ドライバー (x64))
Server	Amazon Redshift クラスターのエンドポイント ホスト。
Database	クラスター用に作成したデータベース。
UID	データベースに接続するアクセス許可を持つデータベースユーザーアカウントのユーザー名。この値はクラスターレベルのアクセス許可ではなくデータベースレベルのアクセス許可ですが、クラスターの起動時に設定した Redshift 管理者ユーザーアカウントを使用できます。
PWD	データベースに接続するデータベースユーザーアカウントのパスワード。
Port	クラスターを起動したときに指定したポート番号。ファイアウォールがある場合、このポートが開いていて使用できることを確認します。

ODBC URL の例を次に示します。

```
Driver={Amazon Redshift ODBC Driver (x64)}; Server=examplecluster.abc123xyz789.us-
west-2.redshift.amazonaws.com; Database=dev; UID=adminuser;
 PWD=insert_your_admin_user_password_here; Port=5439
```

ODBC URL の場所については、「クラスター接続文字列を検索する」を参照してください。

Microsoft Windows での Amazon Redshift ODBC ドライバーの使用

Amazon Redshift データウェアハウスにアクセスするクライアントコンピュータに Amazon Redshift ODBC ドライバーをインストールする必要があります。ドライバーをインストールするコンピュータには、それぞれ以下の最小要件があります。

- マシンの管理者権限。
- マシンは、以下のシステム要件を満たしている。
  - 次のいずれかの OS。
    - Windows 10 または 8.1。
    - Windows Server 2025、2022 , 2019、2016、または 2012。
  - 100 MB の使用可能なディスク容量。
  - 64 ビット Windows 用の Visual Studio 2015 の Visual C++ 再頒布可能パッケージがインストールされている。インストールパッケージは、Microsoft ウェブサイトの「Visual Studio 2022 の Visual C++ 再頒布可能パッケージのダウンロード」からダウンロードできます。

Amazon Redshift ODBC ドライバーをダウンロードしてインストールする

Windows オペレーティングシステム用の Amazon Redshift ODBC ドライバーをダウンロードしてインストールするには、次の手順に従います。Amazon Redshift での使用が認定されているサードパーティーアプリケーションと固有のドライバーを必要とするアプリケーションを実行している場合にのみ、これらとは異なるドライバーを使用できます。

ODBC ドライバーをダウンロードしてインストールするには

1. 次のドライバーをダウンロードします: 64 ビット ODBC ドライバーバージョン 2.2.4.0

   このドライバーの名前は [Amazon Redshift ODBC Driver (x64)] (Amazon Redshift ODBC ドライバー (x64)) です。
2. 「Amazon Redshift ODBC ドライバーバージョン 2.x ライセンス」を確認します。
3. .msi ファイルをダブルクリックし、ウィザードの手順に従ってドライバーをインストールします。

ODBC 接続用のシステム DSN エントリの作成

ODBC ドライバーをダウンロードしてインストールした後、クライアントマシンか Amazon EC2 インスタンスにデータソース名 (DSN) エントリを追加します。SQL クライアントツールは、このデータソースを使用して Amazon Redshift データベースに接続できます。

ユーザー DSN の代わりにシステム DSN を作成することをお勧めします。一部のアプリケーションでは、データのロードに使用しているデータベースユーザーアカウントとは異なるデータベースユーザーアカウントで作成されたユーザー DSN を検出できないことがあります。

> **Note**
>
> AWS Identity and Access Management (IAM) 認証情報または ID プロバイダー (IdP) 認証情報を使用した認証には、追加の手順が必要です。詳細については、「IAM 認証情報を使用するように JDBC または ODBC 接続を設定する」を参照してください。

ODBC 接続用のシステム DSN エントリを作成するには

1. [Start] (スタート) メニューで、「ODBC Data Sources」と入力します。[ODBC Data Sources] (ODBC データソース) を選択します。

   Amazon Redshift への接続に使用するクライアントアプリケーションと同じビット数を持つ ODBC Data Source Administrator を選択してください。
2. [ODBC Data Source Administrator] (ODBC データソース管理者) で、[Driver] (ドライバー) タブを選択し、次のドライバーフォルダ [Amazon Redshift ODBC Driver (x64)] (Amazon Redshift ODBC ドライバー (x64)) を探します。
3. コンピューターのすべてのユーザーに対してドライバーを設定する場合は [システム DSN] タブを選択し、データベースユーザーアカウント別にドライバーを設定する場合は [ユーザー DSN] タブを選択します。
4. [追加] を選択します。[新しいデータソースを作成] ウィンドウが開きます。
5. [Amazon Redshift ODBC driver (x64)] (Amazon Redshift ODBC ドライバー (x64)) を選択し、[Finish] (完了) を選択します。[Amazon Redshift ODBC Driver DSN Setup (ODBC ドライバー DSN セットアップ)] ウィンドウが開きます。
6. [Connection Settings] (接続の設定) セクションで以下の情報を入力します。
   - データソース名

     データソースの名前を入力します。例えば、「Amazon Redshift 入門ガイド」に従っているのであれば、この DSN に関連付けるクラスターを覚えやすくするために `exampleclusterdsn` と入力します。
   - サーバー

     Amazon Redshift クラスターのエンドポイントホストを指定します。この情報は、クラスターの詳細ページの Amazon Redshift コンソールで見つけることができます。詳細については、「Amazon Redshift での接続の設定」を参照してください。
   - ポート

データベースが使用するポート番号を入力します。クラスターの作成、変更、または移行時に選択したポートに応じて、選択したポートへのアクセスを許可します。

- データベース

  Amazon Redshift データベースの名前を入力します。データベース名を指定せずにクラスターを起動した場合は、devを入力します。それ以外の場合は、起動プロセスで選択した名前を使用します。「Amazon Redshift 入門ガイド」に従っているのであれば、dev と入力します。

7. [Authentication] (認証) セクションで、標準認証または IAM 認証を設定する設定オプションを指定します。
8. [SSL Options] (SSL オプション) を選択し、次の値を指定します。
   - 認証モード

     Secure Sockets Layer (SSL) を処理するモードを選択します。テスト環境では、preferを使用できます。ただし、本番稼働環境の場合、および安全なデータ交換が必要な場合は、verify-ca または verify-full を使用します。

   - 最小 TLS

     オプションとして、ドライバーがデータストアで接続の暗号化に使用できる TLS/SSL の最小バージョンを選択します。例えば、TLS 1.2 を指定すると、接続の暗号化に TLS 1.1 は使用できません。デフォルトのバージョンは、TLS 1.2 です。
9. [Proxy] (プロキシ) タブで、任意のプロキシ接続設定を指定します。
10. [Cursor] (カーソル) タブで、クエリ結果を SQL クライアントツールまたはアプリケーションに返す方法に関するオプションを指定します。
11. [詳細オプション] で、logLevel、logPath、compression、および他のオプションの値を指定します。
12. [テスト] を選択します。クライアントコンピュータが Amazon Redshift データベースに接続できる場合は、[Connection successful] (接続成功) というメッセージが表示されます。クライアントコンピュータがデータベースに接続できない場合、ログファイルを生成して AWS のサポートに連絡し、潜在的な問題をトラブルシューティングすることができます。ログの生成については、「(リンク)」を参照してください。
13. [OK] を選択してください。

Linux での Amazon Redshift ODBC ドライバーの使用

Amazon Redshift データウェアハウスにアクセスするクライアントコンピュータに Amazon Redshift ODBC ドライバーをインストールする必要があります。ドライバーをインストールするコンピュータには、それぞれ以下の最小要件があります。

- マシン上のルートアクセス。
- 以下のいずれかのディストリビューション。
  - Amazon Linux 2
  - Amazon Linux 2023
  - Red Hat® Enterprise Linux® (RHEL) 8 以降
  - CentOS 8 以降
  - Debian 11 以降
- 150 MB の使用可能なディスク容量。
- unixODBC 2.2.14 以降。
- glibc 2.26 以降。

Amazon Redshift ODBC ドライバーをダウンロードしてインストールする

Linux 用の Amazon Redshift ODBC ドライバーバージョン 2.x をダウンロードしてインストールするには

1. 次のドライバーをダウンロードします。
   - x86 64 ビット RPM ドライバーバージョン 2.2.4.0
     - Linux アーティファクト署名キー: キー
     - Linux アーティファクト署名付き Hash: Hash
   - ARM 64 ビット RPM ドライバーバージョン 2.2.4.0
     - Linux アーティファクト署名キー: キー
     - Linux アーティファクト署名付き Hash: Hash
   - x86 64 ビット DEB ドライバーバージョン 2.2.4.0
     - Linux アーティファクト署名キー: キー
     - Linux アーティファクト署名付き Hash: Hash
   - ARM 64 ビット DEB ドライバーバージョン 2.2.4.0
     - Linux アーティファクト署名キー: キー

- Linux アーティファクト署名付き Hash: Hash

> **Note**
>
> 32 ビット ODBC ドライバーの提供は終了しました。今後、緊急のセキュリティパッチを除きアップデートはリリースされません。

2. インストールする前に、AWS がドライバーパッケージに署名していることを確認します。パッケージ、その署名キー (証明書)、および署名付きハッシュ (デタッチされた署名) をダウンロードしたら、次のコマンドを実行します。

```
openssl x509 -in SignatureKeyFile -pubkey -noout > redshift-odbc-pubkey.pem
openssl dgst -sha256 -verify redshift-odbc-pubkey.pem \
    -signature SignedHashFile \
    PackageFile
```

パッケージが正規のものである場合、openssl は Verified OK を出力します。出力がそれ以外の場合は、パッケージをインストールしないでください。ファイルを再ダウンロードするか、AWS サポートにお問い合わせください。

例えば、x86 64 ビット RPM を検証するには:

```
openssl x509 -in AmazonRedshiftODBC-64-bit-2.2.4.0.x86_64-certificate.pem -pubkey -
noout > redshift-odbc-pubkey.pem
openssl dgst -sha256 -verify redshift-odbc-pubkey.pem \
    -signature AmazonRedshiftODBC-64-bit-2.2.4.0.x86_64-signature.bin \
    AmazonRedshiftODBC-64-bit-2.2.4.0.x86_64.rpm
```

3. パッケージをダウンロードした場所に移動し、以下のいずれかのコマンドを実行します。使用している Linux ディストリビューションに対応するコマンドを使用します。

RHEL および CentOS オペレーティングシステム (RPM) では、次のコマンドを実行します。

```
sudo yum --nogpgcheck localinstall RPMFileName
```

RPMFileName を RPM パッケージファイル名に置き換えます。例えば、次のコマンドは 64 ビットドライバーのインストールを示しています。

```
sudo yum --nogpgcheck localinstall AmazonRedshiftODBC-64-bit-2.x.xx.xxxx.x86_64.rpm
```

Debian および Ubuntu オペレーティングシステム (DEB) では、ダウンロードした .deb ファイルがあるディレクトリに移動し、次のコマンドを実行します。

```
sudo apt install ./DEBFileName
```

DEBFileName を DEB パッケージのファイル名に置き換えます。例えば、次のようになります。

```
sudo apt install ./AmazonRedshiftODBC-64-bit-2.x.xx.xxxx.amd64.deb
```

ODBC ドライバーマネージャーを使用して ODBC ドライバーを設定する

Linux では、ODBC ドライバーマネージャーを使用して ODBC 接続を設定します。ODBC ドライバーマネージャーは設定ファイルを使用して ODBC データソースおよびドライバーを定義および設定します。使用する ODBC ドライバーマネージャーは、使用するオペレーティングシステムによって異なります。

unixODBC ドライバーマネージャーを使用して ODBC ドライバーを設定する

Amazon Redshift ODBC ドライバーを設定するには、以下のファイルが必要です。

- amazon.redshiftodbc.ini
- odbc.ini
- odbcinst.ini

デフォルトの場所にインストールした場合、amazon.redshiftodbc.ini 設定ファイルは /opt/amazon/redshiftodbcx64 にあります。

また、/opt/amazon/redshiftodbcx64 の下に、odbc.ini と odbcinst.ini のサンプルファイルがあります。これらのファイルは、Amazon Redshift ODBC ドライバーとデータソース名 (DSN) の設定例として使用できます。

設定ファイルに Amazon Redshift ODBC ドライバーのインストールディレクトリを使用することは推奨されません。インストール済みのディレクトリのサンプルファイルは例示のみを目的としていま

す。Amazon Redshift ODBC ドライバーを後で再インストールするか、新しいバージョンにアップグレードすると、インストールディレクトリが上書きされます。インストールディレクトリ内のファイルに加えた変更はすべて失われます。

これを回避するには、インストールディレクトリ以外のディレクトリに amazon.redshiftodbc.ini ファイルをコピーします。このファイルをユーザーのホームディレクトリにコピーする場合は、隠しファイルにするためにファイル名の先頭にピリオド (.) を追加します。

odbc.ini および odbcinst.ini ファイルについては、ユーザーのホームディレクトリにある設定ファイルを使用するか、別のディレクトリに新しいバージョンを作成します。デフォルトでは、Linux オペレーティングシステムには、ユーザーのホームディレクトリ (/home/$USER または ~/.) に odbc.ini ファイルと odbcinst.ini ファイルが必要です。これらの既定のファイルは隠しファイルで、各ファイル名の前にドット (.) が表示されます。これらのファイルは、-aフラグを使用してディレクトリの内容をリストした場合にのみ表示されます。

odbc.ini ファイルと odbcinst.ini ファイルに対してどちらのオプションを選択しても、ファイルを変更してドライバーと DSN 設定情報を追加します。新しいファイルを作成する場合は、これらの設定ファイルの場所を指定する環境変数を設定する必要もあります。

デフォルトでは、ODBC ドライバーマネージャーは、隠しバージョンの odbc.ini および odbcinst.ini 設定ファイル (.odbc.ini と .odbcinst.ini という名前) を使用して設定し、ホームディレクトリに配置します。また、ドライバーのインストールディレクトリの amazon.redshiftodbc.ini ファイルを使用するように設定されています。これらの設定ファイルを他の場所に保存する場合は、ドライバーマネージャーがファイルを見つけることができるように、次に説明する環境変数を設定します。

unixODBC を使用している場合は、次の手順に従います。

- odbc.ini のファイルのフルパスとファイル名に ODBCINI を設定します。
- odbcinst.ini のファイルを含むディレクトリのフルパスに ODBCSYSINI を設定します。
- amazon.redshiftodbc.ini のファイルのフルパスとファイル名に AMAZONREDSHIFTODBCINI を設定します。

以下は、上記の値を設定する例です。

```
export ODBCINI=/usr/local/odbc/odbc.ini
```

```
export ODBCSYSINI=/usr/local/odbc
export AMAZONREDSHIFTODBCINI=/etc/amazon.redshiftodbc.ini
```

Linux でデータソース名 (DSN) を使用して接続を設定する

データソース名 (DSN) を使用してデータストアに接続する場合は、データソース名 (DSN) を定義するように odbc.ini ファイルを設定します。odbc.ini ファイルのプロパティを設定して、データストアの接続情報を指定する DSN を作成します。

Linux オペレーティングシステムでは次の形式を使用します。

```
[ODBC Data Sources]
driver_name=dsn_name

[dsn_name]
Driver=path/driver_file
Host=cluster_endpoint
Port=port_number
Database=database_name
locale=locale
```

次に、Linux オペレーティングシステムでの、64 ビット ODBC ドライバーを使用する odbc.ini の設定の例を示します。

```
[ODBC Data Sources]
Amazon_Redshift_x64=Amazon Redshift ODBC Driver (x64)

[Amazon_Redshift_x64]
Driver=/opt/amazon/redshiftodbcx64/librsodbc64.so
Host=examplecluster.abc123xyz789.us-west-2.redshift.amazonaws.com
Port=5932Database=dev
locale=en-US
```

Linux で DSN を使用しない接続を設定する

DSN を使用しない接続を介してデータストアに接続するには、odbcinst.ini ファイルでドライバーを定義します。次に、アプリケーションに DSN を使用しない接続文字列を指定します。

Linux オペレーティングシステムでは次の形式を使用します。

```
[ODBC Drivers]
driver_name=Installed
...

[driver_name]
Description=driver_description
Driver=path/driver_file

...
```

次に、Linux オペレーティングシステムでの、64 ビット ODBC ドライバーを使用する `odbcinst.ini` の設定の例を示します。

```
[ODBC Drivers]
Amazon Redshift ODBC Driver (x64)=Installed

[Amazon Redshift ODBC Driver (x64)]
Description=Amazon Redshift ODBC Driver (64-bit)
Driver=/opt/amazon/redshiftodbcx64/librsodbc64.so
```

Apple macOS での Amazon Redshift ODBC ドライバーの使用

Amazon Redshift データウェアハウスにアクセスするクライアントコンピュータに Amazon Redshift ODBC ドライバーをインストールする必要があります。ドライバーをインストールするコンピュータには、それぞれ以下の最小要件があります。

- マシン上のルートアクセス。
- Apple macOS システム要件
  - Apple macOS バージョン 11.7 以降 (Apple macOS Big Sur、Monterey、Ventura 以降など) の 64 ビットバージョンが必要です。Redshift ODBC ドライバーは 64 ビットクライアントアプリケーションのみをサポートします。
  - 150 MB の使用可能なディスク容量。
  - ドライバーは、iODBC 3.52.9+ または unixODBC 2.3.7+ で構築されたアプリケーションをサポートします。

Amazon Redshift ODBC ドライバーをダウンロードしてインストールする

Apple macOS で Amazon Redshift ODBC ドライバーをダウンロードしてインストールするには、次の手順に従います。Amazon Redshift での使用が認定されているサードパーティーアプリケーションと固有のドライバーを必要とするアプリケーションを実行している場合にのみ、これらとは異なるドライバーを使用できます。

ODBC ドライバーをダウンロードしてインストールするには

1. 次のドライバーをダウンロードします: 64 ビット ODBC ドライバーバージョン 2.2.4.0

   このドライバーは、x86_64 アーキテクチャと arm64 アーキテクチャの両方でサポートされています。このドライバーの名前は [Amazon Redshift ODBC Driver (x64)] (Amazon Redshift ODBC ドライバー (x64)) です。
2. 「Amazon Redshift ODBC ドライバーバージョン 2.x ライセンス」を確認します。
3. .pkg ファイルをダブルクリックし、ウィザードの手順に従ってドライバーをインストールします。または、以下のコマンドを実行します。

   ```
   sudo installer -pkg PKGFileName -target /
   ```

   PKGFileName を pkg パッケージファイル名に置き換えます。例えば、次のコマンドは 64 ビットドライバーのインストールを示しています。

   ```
   sudo installer -pkg ./AmazonRedshiftODBC-64-bit.X.X.XX.X.universal.pkg -target /
   ```

ODBC ドライバーマネージャーを使用して ODBC ドライバーを設定する

Mac では、ODBC ドライバーマネージャーを使用して ODBC 接続を設定します。ODBC ドライバーマネージャーは設定ファイルを使用して ODBC データソースおよびドライバーを定義および設定します。使用する ODBC ドライバーマネージャーは、使用するオペレーティングシステムによって異なります。

iODBC または unixODBC ドライバーマネージャーを使用して ODBC ドライバーを設定する

Amazon Redshift ODBC ドライバーを設定するには、以下のファイルが必要です。

- `amazon.redshiftodbc.ini`
- `odbc.ini`

- odbcinst.ini

デフォルトの場所にインストールした場合、amazon.redshiftodbc.ini 設定ファイルは /opt/amazon/redshiftodbcx64 にあります。

また、/opt/amazon/redshiftodbcx64 の下に、odbc.ini と odbcinst.ini のサンプルファイルがあります。これらのファイルは、Amazon Redshift ODBC ドライバーとデータソース名 (DSN) の設定例として使用できます。インストール済みのディレクトリのサンプルファイルは例示のみを目的としています。

設定ファイルに Amazon Redshift ODBC ドライバーのインストールディレクトリを使用することは推奨されません。Amazon Redshift ODBC ドライバーを後で再インストールするか、新しいバージョンにアップグレードすると、インストールディレクトリが上書きされます。インストールディレクトリ内のファイルに加えた変更はすべて失われます。

これを回避するには、インストールディレクトリ以外のディレクトリに odbc.ini、odbcinst.ini、amazon.redshiftodbc.ini ファイルをコピーします。これのファイルをユーザーのホームディレクトリにコピーする場合は、隠しファイルにするためにファイル名の先頭にピリオド (.) を追加します。

ファイルを変更して DSN 設定情報を追加します。新しいファイルを作成する場合は、これらの設定ファイルの場所を指定する環境変数を設定する必要もあります。

以下に、環境変数を設定する方法の例を示します。

```
export ODBCINI=/Library/ODBC/odbc.ini
export ODBCSYSINI=/Library/ODBC
export ODBCINSTINI=${ODBCSYSINI}/odbcinst.ini
```

コマンドラインアプリケーションの場合: エクスポートコマンドをシェルスタートアップファイル (例: ~/.bash_profile または ~/.zshrc) に追加します。

サポートされているバージョンのドライバーマネージャーについては、こちらを参照してください。

Apple macOS でデータソース名 (DSN) を使用して接続を設定する

データソース名 (DSN) を使用してデータストアに接続する場合は、データソース名 (DSN) を定義するように odbc.ini ファイルを設定します。odbc.ini ファイルのプロパティを設定して、Redshift データウェアハウスの接続情報を指定する DSN を作成します。

Apple macOS では、次の形式を使用します。

```
[ODBC Data Sources]
driver_name=dsn_name

[dsn_name]
Driver=path/driver_file
Host=cluster_endpoint
Port=port_number
Database=database_name
locale=locale
```

次に、Apple macOS で 64 ビット ODBC ドライバーを使用する odbc.ini の設定の例を示します。

```
[ODBC Data Sources]
Amazon_Redshift_x64=Amazon Redshift ODBC Driver (x64)

[Amazon_Redshift_x64]
Driver=/opt/amazon/redshiftodbcx64/librsodbc64.dylib
Host=examplecluster.abc123xyz789.us-west-2.redshift.amazonaws.com
Port=5932
Database=dev
locale=en-US
```

Apple macOS で DSN を使用しない接続を設定する

DSN を使用しない接続を介して Redshift データウェアハウスに接続するには、odbcinst.ini ファイルでドライバーを定義します。次に、アプリケーションに DSN を使用しない接続文字列を指定します。

Apple macOS では、次の形式を使用します。

```
[ODBC Drivers]
driver_name=Installed
...

[driver_name]
Description=driver_description
Driver=path/driver_file

...
```

次に、Apple macOS で 64 ビット ODBC ドライバーを使用する odbcinst.ini の設定の例を示します。

```
[ODBC Drivers]
Amazon Redshift ODBC Driver (x64)=Installed

[Amazon Redshift ODBC Driver (x64)]
Description=Amazon Redshift ODBC Driver (64-bit)
Driver=/opt/amazon/redshiftodbcx64/librsodbc64.dylib
```

認証方法

不正なアクセスからデータを保護するために、Amazon Redshift データストアでは、すべての接続をユーザー認証情報を使用して認証する必要があります。

次の表は、Amazon Redshift ODBC ドライバーバージョン 2.x への接続に使用できる各認証方法の必須および選択可能な接続オプションを示しています。

認証方法	必須	オプションです。
標準	• ホスト • ポート • データベース • UID • パスワード	
IAM プロフィール	• ホスト • ポート • データベース • IAM • プロファイル	• ClusterID • リージョン • AutoCreate • EndpointURL • StsEndpointURL • InstanceProfile **Note** [ClusterID] と [Region] (リージョン) が別々に設定されていない場合、[Host] (ホスト) で

認証方法	必須	オプションです。
		設定する必要があります。
IAM 認証情報	• ホスト • ポート • データベース • IAM • AccessKeyID • SecretAccessKey	• ClusterID • リージョン • AutoCreate • EndpointURL • StsEndpointURL • SessionToken • UID **Note** [ClusterID] と [Region] (リージョン) が別々に設定されていない場合、[Host] (ホスト) で設定する必要があります。

認証方法	必須	オプションです。
AD FS	• ホスト • ポート • データベース • IAM • plugin_name • UID • パスワード • IdP_Host • IdP_Port	• ClusterID • リージョン • AutoCreate • EndpointUrl • StsEndpointUrl • Preferred_Role • loginToRp • SSL_Insecure **Note** [ClusterID] と [Region] (リージョン) が別々に設定されていない場合、[Host] (ホスト) で設定する必要があります。

認証方法	必須	オプションです。
Azure AD	• ホスト • ポート • データベース • IAM • plugin_name • UID • パスワード • IdP_Tenant • Client_ID • Client_Secret	• ClusterID • リージョン • AutoCreate • EndpointUrl • StsEndpointUrl • Preferred_Role • dbgroups_filter **Note** [ClusterID] と [Region] (リージョン) が別々に設定されていない場合、[Host] (ホスト) で設定する必要があります。
JWT	• ホスト • ポート • データベース • IAM • plugin_name • web_identity_token	• provider_name

認証方法	必須	オプションです。
Okta	• ホスト • ポート • データベース • IAM • plugin_name • UID • パスワード • IdP_Host • App_Name • App_ID	• ClusterID • リージョン • AutoCreate • EndpointUrl • StsEndpointUrl • Preferred_Role **Note** [ClusterID] と [Region] (リージョン) が別々に設定されていない場合、[Host] (ホスト) で設定する必要があります。

認証方法	必須	オプションです。
Ping Federate	• ホスト • ポート • データベース • IAM • plugin_name • UID • パスワード • IdP_Host • IdP_Port	• ClusterID • リージョン • AutoCreate • EndpointUrl • StsEndpointUrl • Preferred_Role • SSL_Insecure • partner_spid **Note** [ClusterID] と [Region] (リージョン) が別々に設定されていない場合、[Host] (ホスト) で設定する必要があります。

認証方法	必須	オプションです。
Browser Azure AD	• ホスト • ポート • データベース • IAM • plugin_name • IdP_Tenant • Client_ID • UID	• ClusterID • リージョン • AutoCreate • EndpointUrl • StsEndpointUrl • Preferred_Role • dbgroups_filter • IdP_Response_Timeout • listen_port **Note** [ClusterID] と [Region] (リージョン) が別々に設定されていない場合、[Host] (ホスト) で設定する必要があります。

認証方法	必須	オプションです。
Browser SAML	• ホスト • ポート • データベース • IAM • plugin_name • login_url • UID	• ClusterID • リージョン • AutoCreate • EndpointUrl • StsEndpointUrl • Preferred_Role • dbgroups_filter • IdP_Response_Timeout • listen_port **Note** [ClusterID] と [Region] (リージョン) が別々に設定されていない場合、[Host] (ホスト) で設定する必要があります。
Auth プロファイル	• ホスト • ポート • データベース • AccessKeyID • SecretAccessKey	

認証方法	必須	オプションです。
Browser Azure AD OAUTH2	• ホスト • ポート • データベース • IAM • plugin_name • IdP_Tenant • Client_ID • UID	• ClusterID • リージョン • EndpointUrl • IdP_Response_Timeout • listen_port • scope • provider_name **Note** [ClusterID] と [リージョン] が別々に設定されていない場合、[ホスト] で設定する必要があります。
AWS IAM アイデンティティーセンター	• ホスト • データベース • plugin_name • idc_region • issuer_url	• idc_client_display_name • idp_response_timeout • listen_port

外部認証情報サービスの使用

また、AD FS、Azure AD、および Okta の組み込みサポートに加えて、Windows バージョンの Amazon Redshift ODBC ドライバーは、他の認証情報サービスのサポートを提供します。ドライバーは、任意の SAML ベースの認証情報プロバイダープラグインを使用して接続を認証できます。

Windows で外部認証情報サービスを設定するには

1. 必要に応じて、認証情報プロバイダープラグインとその他の認証パラメータを指定する IAM プロファイルを作成します。プロファイルは ASCII でエンコードされている必要があり、次のキーと

値のペアが含まれている必要があります。PluginPath はプラグインアプリケーションのフルパスです。

```
plugin_name = PluginPath
```

例:

```
plugin_name = C:\Users\kjson\myapp\CredServiceApp.exe
```

プロファイルの作成方法については、「Amazon Redshift クラスター管理ガイド」の「設定プロファイルを使用する」を参照してください。

2. このプロファイルを使用するようにドライバーを設定します。ドライバーは、プロファイルで指定された認証設定を検出して使用します。

データ型変換

Amazon Redshift ODBC ドライバーのバージョン 2.x では、Amazon Redshift および SQL データ型の変換を行う多くの一般的なデータ形式がサポートされています。

サポートされているデータ型のマッピングを以下の表に示しています。

Amazon Redshift のタイプ	SQL タイプ
BIGINT	SQL_BIGINT
BOOLEAN	SQL_BIT
CHAR	SQL_CHAR
DATE	SQL_TYPE_DATE
DECIMAL	SQL_NUMERIC
DOUBLE PRECISION	SQL_DOUBLE
GEOGRAPHY	SQL_LONGVARBINARY
GEOMETRY	SQL_LONGVARBINARY

Amazon Redshift のタイプ	SQL タイプ
INTEGER	SQL_INTEGER
REAL	SQL_REAL
SMALLINT	SQL_SMALLINT
SUPER	SQL_LONGVARCHAR
TEXT	SQL_LONGVARCHAR
TIME	SQL_TYPE_TIME
TIMETZ	SQL_TYPE_TIME
TIMESTAMP	SQL_TYPE_TIMESTAMP
TIMESTAMPTZ	SQL_TYPE_TIMESTAMP
VARBYTE	SQL_LONGVARBINARY
VARCHAR	SQL_VARCHAR

ODBC ドライバーオプション

ドライバー設定オプションを使用して、Amazon Redshift ODBC ドライバーの動作を制御できます。ドライバーオプションでは、大文字と小文字を区別しません。

Microsoft Windows では、通常、データソース名 (DSN) を設定するときにドライバーオプションを設定します。プログラムを使用して接続するとき、または HKEY_LOCAL_MACHINE\SOFTWARE\ODBC\ODBC.INI*your_DSN* でレジストリキーを追加または変更することによって、接続文字列にドライバーオプションを設定することもできます。

Linux では、odbc.ini および amazon.redshiftodbc.ini ファイルでドライバー設定オプションを設定します。amazon.redshiftodbc.ini ファイルに設定されている設定オプションは、すべての接続に適用されます。これに対して、odbc.iniファイルで設定されている設定オプションは接続に固有です。odbc.ini で設定された設定オプションは、amazon.redshiftodbc.ini で設定された設定オプションよりも優先されます。

以下に、Amazon Redshift ODBC バージョン 2.x ドライバーで指定できるオプションの説明を示します。

トピック

- AccessKeyID
- app_id
- ApplicationName
- app_name
- AuthProfile
- AuthType
- AutoCreate
- BoolsAsChar
- CaFile
- client_id
- client_ secret
- ClusterId
- compression
- データベース
- DatabaseMetadataCurrentDbOnly
- dbgroups_filter
- ドライバー
- DSN
- EnableTableTypes
- EndpointUrl
- 取得
- ForceLowercase
- group_federation
- https_proxy_host
- https_proxy_password
- https_proxy_port

- https_proxy_username
- IAM
- idc_client_display_name
- idc_region
- idp_host
- idp_port
- idp_response_timeout
- idp_tenant
- idp_partition
- idp_use_https_proxy
- InstanceProfile
- issuer_url
- KeepAlive
- KeepAliveCount
- KeepAliveInterval
- KeepAliveIdle
- listen_port
- login_url
- loginToRp
- LogLevel
- LogPath
- LoginTimeout
- MaxLongVarcharSize
- MaxVarcharSize
- Min_TLS
- partner_spid
- Password | PWD
- plugin_name

- Port | PortNumber
- Prefer_PQ
- preferred_role
- プロファイル
- provider_name
- ProxyHost
- ProxyPort
- ProxyPwd
- ProxyUid
- ReadOnly
- リージョン
- SecretAccessKey
- SessionToken
- Server | HostName | Host
- ssl_insecure
- SSLMode
- StreamingCursorRows
- StsConnectionTimeout
- StsEndpointUrl
- トークン
- token_type
- UID | User | LogonID
- UseDeclareFetch
- UseUnicode
- web_identity_token

AccessKeyID

- デフォルト値 - なし
- データ型 – 文字列

ユーザーまたはロールの IAM アクセスキー。このパラメータを設定する場合は、SecretAccessKey も指定する必要があります。

このパラメータはオプションです。

app_id

- デフォルト値 – なし
- データ型 – 文字列

Okta が提供する Amazon Redshift アプリケーションに関連付けられた一意の ID。

このパラメータはオプションです。

ApplicationName

- デフォルト値 - なし
- データ型 - 文字列

監査目的のために Amazon Redshift に渡すクライアントアプリケーションの名前。指定したアプリケーション名は、SYS_CONNECTION_LOG テーブルの「application_name」列に表示されます。これにより、問題をデバッグするときに接続ソースを追跡およびトラブルシューティングできます。

このパラメータはオプションです。

app_name

- デフォルト値 – なし
- データ型 – 文字列

Amazon Redshift への接続を認証するために使用する Okta アプリケーションの名前。

このパラメータはオプションです。

AuthProfile

- デフォルト値 - なし
- データ型 - 文字列

接続設定を管理するために使用される認証プロファイル。このパラメータを設定する場合は、AccessKeyID と SecretAccessKey も設定する必要があります。

このパラメータはオプションです。

AuthType

- デフォルト値 - 標準
- データ型 - 文字列

このオプションは、[Amazon Redshift ODBC Driver DSN Setup] (Amazon Redshift ODBC ドライバー DSN セットアップ) ダイアログボックスを使用して DSN を設定するときにドライバーが使用する認証モードを指定します。

- 標準: Amazon Redshift ユーザー名とパスワードを使用した標準認証。
- AWS プロファイル: プロファイルを使用した IAM 認証。
- AWS IAM 認証情報: IAM 認証情報を使用した IAM 認証。
- ID プロバイダー: AD FS: Active Directory フェデレーションサービス (AD FS) を使用した IAM 認証。
- ID プロバイダー: 認可プラグイン: AWS IAM アイデンティティセンターのトークン、または AWS IAM アイデンティティセンターにリンクされているウェブ ID プロバイダーからの OpenID Connect (OIDC) JSON ベースの ID トークン (JWT) を受け入れる認可プラグイン。
- ID プロバイダー: Azure AD: Azure AD ポータルを使用した IAM 認証。
- ID プロバイダー: JWT: JSON ウェブトークン (JWT) を使用した IAM 認証。
- ID プロバイダー: Okta: Okta を使用した IAM 認証。
- ID プロバイダー: PingFederate: PingFederate を使用した IAM 認証。

このオプションは、Windows ドライバーの [Amazon Redshift ODBC Driver DSN Setup] (Amazon Redshift ODBC ドライバー DSN セットアップ) ダイアログボックスを使用して DSN を設定する場合にのみ使用できます。接続文字列または Windows 以外のマシンを使用して接続を設定すると、ドライバーは標準、AWS プロファイル、または指定した認証情報に基づく AWS のIAM 認証情報の認証を使用するかどうかを自動的に決定します。ID プロバイダーを使用するには、plugin_name プロパティを設定する必要があります。

このパラメータは必須です。

AutoCreate

- デフォルト値: 0
- データ型 – ブール型

指定されたユーザーが存在しない場合に、ドライバーが新しいユーザーを作成するかどうかを指定するブール値。

- 1 | TRUE: [UID] で指定されたユーザーが存在しない場合、ドライバーは新しいユーザーを作成します。
- 0 | FALSE: ドライバーは新しいユーザーを作成しません。指定したユーザーが存在しない場合、認証は失敗します。

このパラメータはオプションです。

BoolsAsChar

- デフォルト値: 0
- データ型 – ブール型

ドライバーがビットタイプではなく文字タイプとして BOOLEAN 列を報告するかどうかを指定します。

- 1 | TRUE: ドライバーは、列サイズが 1 の SQL_VARCHAR として BOOLEAN 列を報告します。ドライバーは、ブール値を文字列「0」および「1」として返します。
- 0 | FALSE: ドライバーは BOOLEAN 列を SQL_BIT として報告します。これがデフォルトの動作です。

このオプションは、アプリケーションが文字データとして返される BOOLEAN 列に依存している場合に使用します。

このパラメータはオプションです。このオプションは、ドライバーバージョン 2.2.1 以降で使用できます。

CaFile

- デフォルト値 - なし

- データ型 - 文字列

IAM 認証のいくつかの形式で使用される CA 証明書ファイルへのファイルパス。

このパラメータは Linux でのみ使用できます。

このパラメータはオプションです。

client_id

- デフォルト値 – なし
- データ型 - 文字列

Azure AD の Amazon Redshift アプリケーションに関連付けられたクライアント ID。

このパラメータは、Azure AD サービスを介して認証する場合に必要です。

client_ secret

- デフォルト値 - なし
- データ型 - 文字列

Azure AD の Amazon Redshift アプリケーションに関連付けられたシークレットキー。

このパラメータは、Azure AD サービスを介して認証する場合に必要です。

ClusterId

- デフォルト値 - なし
- データ型 - 文字列

接続する Amazon Redshift クラスターの名前。IAM 認証に使用されます。クラスター ID は Server パラメータでは指定されません。

このパラメータはオプションです。

compression

- デフォルト値 - オフ
- データ型 - 文字列

Amazon Redshift サーバーとクライアントまたはドライバーとの間のワイヤプロトコル通信で使用する圧縮方法。

次の値を指定できます。

- lz4: Amazon Redshift とのワイヤプロトコル通信で使用する圧縮方法を `lz4` に設定します。
- zstd: Amazon Redshift とのワイヤプロトコル通信で使用する圧縮方法を `zstd` に設定します。
- off: Amazon Redshift とのワイヤプロトコル通信では圧縮を使用しません。

このパラメータはオプションです。

データベース

- デフォルト値 - なし
- データ型 - 文字列

アクセスする Amazon Redshift データベースの名前。

このパラメータは必須です。

DatabaseMetadataCurrentDbOnly

- デフォルト値 - 1
- データ型 – ブール型

ドライバーが複数のデータベースとクラスターからメタデータを返すかどうかを指定するブール値。

- 1 | TRUE: ドライバーは、現在のデータベースからのみメタデータを返します。
- 0 | FALSE。ドライバーは、複数の Amazon Redshift データベースとクラスターにわたってメタデータを返します。

このパラメータはオプションです。

dbgroups_filter

- デフォルト値 - なし
- データ型 - 文字列

Azure、Browser Azure、および Browser SAML 認証型を使用しているときに Amazon Redshift への SAML レスポンスから受信する DbGroups を除外するために指定できる正規表現。

このパラメータはオプションです。

ドライバー

- デフォルト値 - Amazon Redshift ODBC ドライバー (x64)
- データ型 - 文字列

ドライバーの名前。[Amazon Redshift ODBC Driver (X64)] (Amazon Redshift ODBC ドライバー (x64)) はサポートされる唯一の値です。

DSN を設定しない場合、このパラメータは必須です。

DSN

- デフォルト値 - なし
- データ型 - 文字列

ドライバーデータソースの名前。アプリケーションは、SQLDriverConnect API で DSN を指定します。

[Driver] (ドライバー) を設定しない場合、このパラメータは必須です。

EnableTableTypes

- デフォルト値 - 1
- データ型 – ブール型

アプリケーションが SQLTables カタログ関数を呼び出すときに、ドライバーがデータソースから詳細なテーブルタイプ情報を認識するかどうかを指定します。デフォルトでは、ドライバーは詳細なテーブルタイプを認識します。

- 1 | TRUE: ドライバーは次のテーブルタイプを認識します。TABLE、VIEW、SYSTEM TABLE、SYSTEM VIEW、EXTERNAL TABLE、LOCAL TEMPORARY。
- 0 | FALSE: ドライバーは、詳細なテーブルタイプ情報を一般的な TABLE および VIEW テーブルタイプに正規化します。

このパラメータはオプションです。ドライバーバージョン 2.2.1 以降で有効です。

EndpointUrl

- デフォルト値 - なし
- データ型 - 文字列

IAM 認証のために Amazon Redshift Coral サービスと通信するために使用される優先エンドポイント。

このパラメータはオプションです。

取得

- デフォルト値 – 100
- データ型 – 整数

Declare/Fetch モードを有効にすると、このオプションはドライバーがラウンドトリップごとに取得する行数を制御し、ワークロードのメモリ使用量とネットワークオーバーヘッドのバランスを調整できます。値を大きくすると、一度により多くの行がフェッチされるため、フェッチあたりのメモリが増加する代わりにサーバーとのラウンドトリップ回数が減ります。値を小さくするとメモリ使用量は減りますが、ドライバーがサーバーに戻る頻度が増加します。UseDeclareFetch が有効になっているときにドライバーが一度に返す行数を設定します。詳細については、「UseDeclareFetch」を参照してください。

このパラメータはオプションです。

ForceLowercase

- デフォルト値: 0
- データ型 – ブール型

シングルサインオン認証を使用するときに ID プロバイダーから Amazon Redshift に送信されたすべてのデータベースグループ (DbGroups) をドライバーが小文字にするかどうかを指定するブール値。

- 1 | TRUE: ドライバーは、ID プロバイダーから送信されるすべての DbGroups を小文字にします。
- 0 | FALSE: ドライバーは DbGroups を変更しません。

このパラメータはオプションです。

group_federation

- デフォルト値: 0
- データ型 – ブール型

プロビジョニングされたクラスターでの一時的なクラスター認証情報を取得するために `getClusterCredentialsWithIAM` API を使用するかどうかを指定するブール値。このオプションにより、IAM ユーザーは、プロビジョニングされたクラスターでの Redshift データベースロールと統合できます。このオプションは Redshift Serverless の名前空間には適用されないことに注意してください。

- 1 | TRUE: ドライバーは `getClusterCredentialsWithIAM` API を使用して、プロビジョニングされたクラスターでの一時的なクラスター認証情報を取得します。
- 0 | FALSE: ドライバーは、プロビジョニングされたクラスターでの一時的なクラスター認証情報を取得するためにデフォルトの `getClusterCredentials` API を使用します。

このパラメータはオプションです。

https_proxy_host

- デフォルト値 - なし
- データ型 - 文字列

IAM 認証プロセスを渡すプロキシサーバーのホスト名または IP アドレス。

このパラメータはオプションです。

https_proxy_password

- デフォルト値 - なし
- データ型 - 文字列

プロキシサーバーにアクセスするために使用するパスワード。IAM 認証に使用されます。

このパラメータはオプションです。

https_proxy_port

- デフォルト値 - なし
- データ型 – 整数

プロキシサーバーがクライアント接続をリッスンするために使用するポートの番号。IAM 認証に使用されます。

このパラメータはオプションです。

https_proxy_username

- デフォルト値 - なし
- データ型 - 文字列

プロキシサーバーへのアクセスに使用するユーザー名。IAM 認証に使用されます。

このパラメータはオプションです。

IAM

- デフォルト値: 0
- データ型 – ブール型

ドライバーが接続を認証するために IAM 認証方法を使用するかどうかを指定するブール値。

- 1 | TRUE: ドライバーは、IAM 認証方法の 1 つを使用します (アクセスキーと秘密キーのペア、プロファイル、または認証情報サービスを使用)。
- 0 | FALSE。ドライバーは標準認証を使用します (データベースのユーザー名とパスワードを使用)。

このパラメータはオプションです。

idc_client_display_name

- デフォルト値 - Amazon Redshift ODBC ドライバー
- データ型 - 文字列

BrowserIdcAuthPlugin を使用しているクライアントに使用される表示名。

このパラメータはオプションです。

idc_region

- デフォルト値 - なし
- データ型 - 文字列

AWS IAM アイデンティティセンターインスタンスが配置されている AWS リージョン。

このパラメータは、plugin_name 設定オプションで `BrowserIdcAuthPlugin` を使用して認証する場合にのみ必要です。

idp_host

- デフォルト値 – なし
- データ型 – 文字列

Amazon Redshift への認証に使用する IdP (ID プロバイダー) のホスト。

このパラメータはオプションです。

idp_port

- デフォルト値 - なし
- データ型 – 整数

Amazon Redshift への認証に使用する IdP (ID プロバイダー) のポート。クラスターの作成、変更、または移行時に選択したポートに応じて、選択したポートへのアクセスを許可します。

このパラメータはオプションです。

idp_response_timeout

- デフォルト値 – 120
- データ型 – 整数

ブラウザのプラグインを介して SAML または Azure AD サービスを使用する場合に、ドライバーが ID プロバイダーからの SAML 応答を待機する秒数です。

このパラメータはオプションです。

idp_tenant

- デフォルト値 – なし
- データ型 - 文字列

Amazon Redshift アプリケーションに関連付けられた Azure AD のテナント ID。

このパラメータは、Azure AD サービスを介して認証する場合に必要です。

idp_partition

- デフォルト値 - なし
- データ型 - 文字列

ID プロバイダー (IdP) が設定されているクラウドパーティションを指定します。これにより、ドライバーが接続する IdP 認証エンドポイントが決まります。

このパラメータを空白のままにすると、ドライバーはデフォルトで商用パーティションを使用します。可能な値は以下のとおりです。

- us-gov: IdP が Azure Government で設定されている場合は、この値を使用します。例えば、Azure AD Government はエンドポイント `login.microsoftonline.us` を使用します。
- cn: IdP が中国のクラウドパーティションで設定されている場合は、この値を使用します。例えば、Azure AD China はエンドポイント `login.chinacloudapi.cn` を使用します。

このパラメータはオプションです。

idp_use_https_proxy

- デフォルト値: 0
- データ型 – ブール型

ドライバーがプロキシサーバー経由で ID プロバイダー (IdP) の認証プロセスを渡すかどうかを指定するブール値。

- 1 | TRUE: ドライバーは、プロキシサーバーを介して IdP 認証プロセスを渡します。
- 0 | FALSE。ドライバーはプロキシサーバー経由で IdP 認証プロセスを渡しません。

このパラメータはオプションです。

InstanceProfile

- デフォルト値: 0
- データ型 – ブール型

認証にプロファイルを使用するように設定されている場合に、ドライバーが Amazon EC2 インスタンスプロファイルを使用するかどうかを指定するブール値。

- 1 | TRUE: ドライバーは Amazon EC2 インスタンスプロファイルを使用します。
- 0 | FALSE。代わりに、ドライバーは、[Profile Name] (プロファイル名) オプション ([Profile] (プロファイル)) で指定されたプロファイルの連鎖ロールを使用します。

このパラメータはオプションです。

issuer_url

- デフォルト値 - なし
- データ型 - 文字列

AWS IAM アイデンティティセンターサーバーのインスタンスエンドポイントを指します。

このパラメータは、plugin_name 設定オプションで `BrowserIdcAuthPlugin` を使用して認証する場合にのみ必要です。

KeepAlive

- デフォルト値 - 1
- データ型 – ブール型

接続のタイムアウトを防ぐために TCP キープアライブを使用するようにドライバーを指定するブール値。

- 1 | TRUE: ドライバーは、TCP キープアライブを使用して、接続がタイムアウトするのを防ぎます。
- 0 | FALSE。ドライバーは TCP キープアライブを使用しません。

このパラメータはオプションです。

KeepAliveCount

- デフォルト値: 0
- データ型 – 整数

接続が切断されていると見なされる前に失うことが許容される TCP キープアライブパケットの数。このパラメータが 0 に設定されていると、ドライバーはこの設定のためにシステムデフォルトを使用します。

このパラメータはオプションです。

KeepAliveInterval

- デフォルト値: 0
- データ型 – 整数

TCP キープアライブを再送信する間隔の秒数。このパラメータが 0 に設定されていると、ドライバーはこの設定のためにシステムデフォルトを使用します。

このパラメータはオプションです。

KeepAliveIdle

- デフォルト値: 0
- データ型 – 整数

ドライバーが TCP キープアライブパケットを送信する前にアイドル状態である秒数。このパラメータが 0 に設定されていると、ドライバーはこの設定のためにシステムデフォルトを使用します。

このパラメータはオプションです。

listen_port

- デフォルト値 - 7890
- データ型 – 整数

ブラウザのプラグインを介して SAML、Azure AD または AWS IAM アイデンティティセンターのサービスを使用する場合に、ID プロバイダーから SAML レスポンスを受信するためにドライバーが使用するポートです。

このパラメータはオプションです。

login_url

- デフォルト値 – なし
- データ型 - 文字列

汎用 Browser SAML プラグインを使用する場合に、ID プロバイダーのウェブサイト上のリソース URL です。

このパラメータは、ブラウザプラグインを介して SAML または Azure AD サービスで認証する場合に必要です。

loginToRp

- デフォルト値 - urn:amazon:webservices
- データ型 - 文字列

AD FS 認証の種類に使用する証明書利用者の信頼。

この設定はオプションです。

LogLevel

- デフォルト値: 0
- データ型 – 整数

> **Note**
>
> LogLevel および LogPath はグローバル設定です。ドライバーはそれらを次の優先順位で読み取ります。
>
> 1. 接続文字列 (LogLevel=6;LogPath=/path;) – 最大優先度で、接続時に他のすべての設定をオーバーライドします。
> 2. Windows の場合: [DSN セットアップ] ダイアログ ([詳細設定] タブ)。
>
>   Linux および macOS の場合: odbc.ini ファイルの [ODBC] セクション。
> 3. amazon.redshiftodbc.ini ファイルの **[DRIVER]** セクション – 上記のソースから口グ記録が有効になっていない場合のフォールバックとして使用されます。
>
> 個別の DSN セクション (例: [MyDSN]) で LogLevel または LogPath を設定しても効果はありません。これらのオプションは、DSN ごとのセクションからは読み取られません。

このプロパティを使用して、ドライバーの口グ記録を有効または無効にし、口グファイルに含まれる詳細の量を指定します。口グ記録によってパフォーマンスが低下し、ディスク容量が大量に消費される可能性があるため、問題をキャプチャするのに十分な時間だけ口グ記録を有効にすることをお勧めします。

プロパティは以下のいずれかの値に設定します。

- 0: OFF。すべての口グ記録を無効にします。
- 1: FATAL。致命的なエラー用に予約されています。
- 2: ERROR。接続失敗や認証エラーなどのエラーイベントを口グに記録します。
- 3: WARN。エラーに加えて警告や予期しない状態を口グに記録します。
- 4: INFO。接続ステータスやサーバーパラメータなどの一般的な進行状況情報を口グに記録します。
- 5: DEBUG。ODBC API 関数呼び出しを含むすべてのドライバーアクティビティを、引数や戻り値と共に口グに記録します。
- 6: TRACE。追加の低レベルプロトコルや内部状態の詳細など、すべてのドライバーアクティビティを口グに記録します。

> **Important**
>
> ログレベル 5 (DEBUG) 以上では、すべてのドライバーアクティビティが取得されます。これには、接続認証情報やクエリデータなどの機密情報が含まれる場合もあります。これらのログレベルは、問題を取得するために一時的にのみ有効にし、サポートやサードパーティーとログを共有する前に、すべての機密情報が秘匿化されていることを確認してください。

ログ記録を有効にすると、ドライバーは LogPath プロパティで指定された場所に次のログファイルを生成します。

- 接続のハンドシェイク中に発生したドライバーのアクティビティをログに記録する `redshift_odbc.log.1` ファイル。
- データベースに接続した後のすべてのドライバーのアクティビティ用 `redshift_odbc.log` ファイル。

このパラメータはオプションです。

LogPath

- デフォルト値 - OS 固有の TEMP ディレクトリ
- データ型 - 文字列

> **Note**
>
> `LogLevel` および `LogPath` はグローバル設定です。ドライバーはそれらを次の優先順位で読み取ります。
>
> 1. 接続文字列 (`LogLevel=6;LogPath=/path;`) – 最大優先度で、接続時に他のすべての設定をオーバーライドします。
> 2. Windows の場合: [DSN セットアップ] ダイアログ ([詳細設定] タブ)。
>
>   Linux および macOS の場合: `odbc.ini` ファイルの `[ODBC]` セクション。
> 3. `amazon.redshiftodbc.ini` ファイルの **`[DRIVER]`** セクション – 上記のソースから口グ記録が有効になっていない場合のフォールバックとして使用されます。

個別の DSN セクション (例: [MyDSN]) で LogLevel または LogPath を設定しても効果はありません。これらのオプションは、DSN ごとのセクションからは読み取られません。

LogLevel が 0 より大きい場合にドライバーがログファイルを保存するフォルダへのフルパス。

このパラメータはオプションです。

LoginTimeout

- デフォルト値: 0
- データ型 – 整数

タイムアウトする前に、ドライバーがサーバーへの接続が確立されるまで待機する秒数。値 0 はタイムアウトがないことを意味します (ドライバーは無期限に待機します)。

このオプションは、接続文字列 (LoginTimeout または短い エイリアス LT を使用) や DSN 設定で設定できます。あるいは、SQLSetConnectAttr と SQL_ATTR_LOGIN_TIMEOUT を使用してプログラムで設定することもできます。Windows では、このオプションは [DSN セットアップ] ダイアログ ([詳細設定] タブ) でも使用できます。

このパラメータはオプションです。

MaxLongVarcharSize

- デフォルト値 – 65535
- データ型 – 整数

[MaxVarcharSize] によって大きなテキスト (SQL_LONGVARCHAR) として再分類された列について、ドライバーが報告する列サイズ。この値は、メタデータ応答において、その列に実際に宣言されたサイズの代わりに使用されます。

MSDASQL などの一部のミドルウェアレイヤーでは、報告されるサイズが 8,001 ~ 32,768 バイトの大容量テキスト列が拒否されるため、デフォルトの 65,535 が推奨されます。65,535 を報告することで、ドライバーは再分類された列をエラーなしで受け入れます。ほとんどのユーザーは、この値を変更する必要はありません。

[MaxVarcharSize] が 0 (再分類が無効) に設定されている場合、このパラメータは効果がありません。

このパラメータはオプションです。ドライバーバージョン 2.1.16 以降で有効です。

MaxVarcharSize

- デフォルト値 – 255
- データ型 – 整数

ドライバーが通常のテキスト列 (SQL_VARCHAR) として報告する列サイズの最大値です。ドライバーは、この値を超える CHAR 列および VARCHAR 列を、大容量テキスト (UseUnicode が有効な場合は SQL_LONGVARCHAR または SQL_WLONGVARCHAR) として再分類します。

この再分類により、特定のサイズを超える通常のテキスト列を処理できないアプリケーションまたはミドルウェアレイヤーの エラーを防ぐことができます。例えば、MSDASQL (OLE DB) を介して Amazon Redshift にアクセスする SQL Server リンクサーバーは、約 8,000 バイトを超える SQL_VARCHAR 列を処理できないため、オーバーサイズの列を大容量テキストとして再分類することで、変換の失敗を回避できます。

- 255 (デフォルト): 255 文字を超える列は SQL_LONGVARCHAR として報告されます。
- 0: 再分類を無効にします。すべての列は、サイズに関係なく SQL_VARCHAR として報告されます。
- カスタム値: 任意の正の整数。指定された値より大きい列は SQL_LONGVARCHAR として報告されます。

列が再分類されると、ドライバーは [MaxLongVarcharSize] で指定された値を使用してそのサイズを報告します。

このパラメータはオプションです。ドライバーバージョン 2.1.16 以降で有効です。

Min_TLS

- デフォルト値 - 1.2
- データ型 - 文字列

ドライバーがデータストアが接続の暗号化に使用することを許可する TLS/SSL の最小バージョン。例えば、TLS 1.2 を指定すると、接続の暗号化に TLS 1.1 は使用できません。

Min_TLS には、次の値を使用できます。

- 1.0: 接続は少なくとも TLS 1.0 を使用する必要があります。
- 1.1: 接続は少なくとも TLS 1.1 を使用する必要があります。
- 1.2: 接続は少なくとも TLS 1.2 を使用する必要があります。

このパラメータはオプションです。

partner_spid

- デフォルト値 – なし
- データ型 – 文字列

PingFederate サービスを使用して接続を認証するときに使用するパートナー SPID (サービスプロバイダー ID) の値。

このパラメータはオプションです。

Password | PWD

- デフォルト値 - なし
- データ型 - 文字列

[ユーザー] フィールドに入力したデータベースユーザー名に対応するパスワード ([UID] | [ユーザー] | [LogonID])。

このパラメータはオプションです。

plugin_name

- デフォルト値 – なし
- データ型 - 文字列

認証に使用する認証情報プロバイダープラグイン名。

サポートされる値は次のとおりです。

- ADFS: 認証に Active Directory フェデレーションサービスを使用する。

- AzureAD: 認証に Microsoft Azure Active Directory (AD) サービスを使用する。
- BrowserAzureAD: 認証に Microsoft Azure Active Directory (AD) サービス用のブラウザプラグインを使用する。
- BrowserIdcAuthPlugin : AWS IAM アイデンティティセンターを使用する認可プラグイン。
- BrowserSAML: 認証に Okta や Ping などの SAML サービス用のブラウザプラグインを使用する。
- IdpTokenAuthPlugin: AWS IAM アイデンティティセンターのトークン、または AWS IAM アイデンティティセンターにリンクされているウェブ ID プロバイダーからの OpenID Connect (OIDC) JSON ベースの ID トークン (JWT) を受け入れる認可プラグイン。
- JWT: 認証に JSON ウェブトークン (JWT) を使用する。
- Ping: 認証に PingFederate サービスを使用する。
- Okta: 認証に Okta サービスを使用する。

このパラメータはオプションです。

Port | PortNumber

- デフォルト値 - 5439
- データ型 – 整数

Amazon Redshift サーバーがクライアント接続をリッスンするために使用する TCP ポートの番号。

このパラメータはオプションです。

Prefer_PQ

- デフォルト値 - 1
- データ型 – ブール型

ドライバーが TLS 1.3 ClientHello でポスト量子ハイブリッドキー交換グループをアドバタイズするかどうかを指定します。ポスト量子グループが提供され、サーバーもそれらをサポートする場合、サーバーはその中から 1 つを選択します。それ以外の場合、ハンドシェイクは従来のキー交換に透過的にフォールバックします。

- 1 | TRUE: ドライバーはポスト量子ハイブリッド TLS キー交換グループをアドバタイズします。

- 0 | FALSE: ドライバーはポスト量子グループをアドバタイズしません。ドライバーが提供するポスト量子キー交換グループがネットワークミドルボックスに認識されず、TLS ハンドシェイクがブロックされる場合は、この設定を使用します。

このパラメータはオプションです。ドライバーバージョン 2.2.1 以降で有効です。

preferred_role

- デフォルト値 – なし
- データ型 - 文字列

Amazon Redshift への接続中に引き受けるロール。IAM 認証に使用されます。

このパラメータはオプションです。

プロファイル

- デフォルト値 - なし
- データ型 - 文字列

Amazon Redshift への認証に使用されるユーザーの AWS プロファイルの名前。

- Use Instance Profile パラメータ (InstanceProfile プロパティ) が 1 | TRUE に設定されている場合、その設定が優先され、ドライバーは代わりに Amazon EC2 インスタンスプロファイルを使用します。
- プロファイルを含む認証情報ファイルのデフォルトの場所は `~/.aws/Credentials` です。`AWS_SHARED_CREDENTIALS_FILE` 環境変数を使用すると、別の認証情報ファイルを指定できます。

このパラメータはオプションです。

provider_name

- デフォルト値 - なし
- データ型 - 文字列

CREATE IDENTITY PROVIDER クエリを使用してユーザーが作成した認証プロバイダー。ネイティブの Amazon Redshift 認証で使用されます。

このパラメータはオプションです。

ProxyHost

- デフォルト値 - なし
- データ型 - 文字列

接続に使用するプロキシサーバーのホスト名または IP アドレス。

このパラメータはオプションです。

ProxyPort

- デフォルト値 - なし
- データ型 – 整数

プロキシサーバーがクライアント接続をリッスンするために使用するポートの番号。

このパラメータはオプションです。

ProxyPwd

- デフォルト値 - なし
- データ型 - 文字列

プロキシサーバーにアクセスするために使用するパスワード。

このパラメータはオプションです。

ProxyUid

- デフォルト値 - なし
- データ型 - 文字列

プロキシサーバーへのアクセスに使用するユーザー名。

このパラメータはオプションです。

ReadOnly

- デフォルト値: 0
- データ型 – ブール型

ドライバーが読み込み専用モードにするかどうかを指定するブール値。

- 1 | TRUE: 接続は読み取り専用モードであり、データストアに書き込むことができません。
- 0 | FALSE: 接続は読み取り専用モードではなく、データストアに書き込むことができます。

このパラメータはオプションです。

リージョン

- デフォルト値 - なし
- データ型 - 文字列

クラスターのある AWS リージョン。

このパラメータはオプションです。

SecretAccessKey

- デフォルト値 - なし
- データ型 - 文字列

ユーザーまたはロールの IAM シークレットキー。このパラメータを設定する場合は、AccessKeyID も設定する必要があります。

このパラメータはオプションです。

SessionToken

- デフォルト値 - なし
- データ型 - 文字列

認証に使用している IAM ロールに関連付けられた一時的な IAM セッショントークン。

このパラメータはオプションです。

Server | HostName | Host

- デフォルト値 - なし
- データ型 - 文字列

接続先のエンドポイントサーバー。

このパラメータは必須です。

ssl_insecure

- デフォルト値: 0
- データ型 – ブール型

ドライバーが IdP サーバー証明書の信頼性をチェックするかどうかを指定するブール値。

- 1 | TRUE: ドライバーは、IdP サーバー証明書の信頼性をチェックしません。
- 0 | FALSE: ドライバーは、IdP サーバー証明書の信頼性をチェックします。

このパラメータはオプションです。

SSLMode

- デフォルト値 – `verify-ca`
- データ型 - 文字列

Amazon Redshift に接続するときに使用する SSL 証明書認証モード。以下の値を指定できます。

- `verify-full`: SSL、信頼できる認証機関、および証明書に一致するサーバー名のみを使用して接続します。
- `verify-ca`: SSL と信頼できる認証機関のみを使用して接続します。
- `require`: SSL のみを使用して接続します。
- `prefer`: 可能な場合は SSL を使用して接続します。それ以外の場合は、SSL を使用せずに接続します。

- allow: デフォルトでは、SSL を使用せずに接続します。サーバーが SSL 接続を必要とする場合は、SSL を使用します。
- disable: SSL を使用せずに接続します。

> **Note**
>
> 可能な限り、SSL 接続に verify-full を使用することをお勧めします。

このパラメータはオプションです。

StreamingCursorRows

- デフォルト値: 0
- データ型 – 整数

ドライバーが前進のみのカーソルに対して一度にメモリに保持する行数を制限し、大きな結果セットが一度にすべてメモリにロードされる必要がないようにします。

プロパティは以下のいずれかの値に設定します。

- 0: ドライバーは、結果セット全体をメモリにロードしてから、行をアプリケーションに返します (デフォルト)。結果セットが非常に大きいと、大量のメモリを消費する可能性があります。
- 正の整数: 前進のみのカーソルの場合、ドライバーは一度にこの数の行のみをメモリに保持します。これらの行をアプリケーションに返して解放し、アプリケーションが現在のバッチを消費すると次のバッチを取得します。

このオプションは、常にインメモリ動作を使用するスクロール可能なカーソルには影響しません。Windows では、Amazon Redshift ODBC ドライバー DSN セットアップダイアログによって、このフィールドに 100 の値が事前に入力されていることに注意してください。

ストリーミングカーソルモードには、前進のみのカーソルが必要です。クライアント側カーソル (CSC) オプションが有効になっている場合は、このオプションが優先され、ストリーミングカーソルモードは無効になります。

> **Note**
>
> StreamingCursorRows と UseDeclareFetch はどちらも、結果セット全体をメモリにロードするのではなく、結果をバッチします。UseDeclareFetch はサーバー側のカーソルを使用し、通常、スループットが大幅に向上するため、フェッチパフォーマンスが重要事項である場合はこちらを優先してください。StreamingCursorRows では、トランザクションラッピングやサーバー側のカーソルは必要ありません。このプロパティがピーク速度よりも重要な場合は、こちらを選択してください。これは、メモリフットプリントを抑えた前進のみのストリームです。UseDeclareFetch を有効にすると、そちらが優先され、StreamingCursorRows は強制的に 0 になります。詳細については、「UseDeclareFetch」を参照してください。

このパラメータはオプションです。

StsConnectionTimeout

- デフォルト値: 0
- データ型 – 整数

IAM 接続の最大待機時間 (秒)。0 に設定されている、または指定されていない場合、ドライバーは AWS STS の呼び出しに、それぞれ 60 秒間待機します。

このパラメータはオプションです。

StsEndpointUrl

- デフォルト値 - なし
- データ型 - 文字列

このオプションは、AWS Security Token Service (AWS STS) との通信に使用される優先エンドポイントを指定します。

このパラメータはオプションです。

トークン

- デフォルト値 - なし

- データ型 - 文字列

AWS IAM アイデンティティセンターが提供するアクセストークン、または AWS IAM アイデンティティセンターにリンクされているウェブ ID プロバイダーが提供する OpenID Connect (OIDC) JSON ウェブトークン (JWT)。アプリケーションは、このトークンを生成するために、AWS IAM アイデンティティセンターまたは AWS IAM アイデンティティセンターにリンクされている ID プロバイダーにより、アプリケーションのユーザーを認証する必要があります。

このパラメータは `IdpTokenAuthPlugin` で機能します。

token_type

- デフォルト値 - なし
- データ型 - 文字列

`IdpTokenAuthPlugin` で使用しているトークンのタイプ。

次の値を指定できます。

ACCESS_TOKEN

AWS IAM アイデンティティセンターが提供するアクセストークンを使用する場合は、これを入力します。

EXT_JWT

AWS IAM アイデンティティセンターと統合されているウェブベースの ID プロバイダーが提供する OpenID Connect (OIDC) JSON ウェブトークン (JWT) を使用する場合は、これを入力します。

このパラメータは `IdpTokenAuthPlugin` で機能します。

UID | User | LogonID

- デフォルト値 - なし
- データ型 - 文字列

Amazon Redshift サーバーへのアクセスに使用するユーザー名。

このパラメータは、データベース認証を使用する場合に必要です。

UseDeclareFetch

- デフォルト値: 0
- データ型 – ブール型

クエリがメモリに保持するには大きすぎる結果セットを返す場合、このオプションを有効にすると、ドライバーは結果全体を一度にバッファするのではなく、固定サイズのバッチで処理します。クエリが返す行数に関係なくクライアントのメモリフットプリントが一定の範囲に抑えられ、予測可能になります。これは、ドライバーが Declare/Fetch モードを使用するかどうかを指定するブール値で、一度に設定された行数を返します。バッチごとに返される行数を設定するには、Fetch オプションを使用します。詳細については、「取得」を参照してください。

- 1 | TRUE: ドライバーは Declare/Fetch モードを使用し、一度に Fetch オプションで指定された行数を返します。
- 0 | FALSE: ドライバーはクエリ結果全体を一度に返します。

このパラメータはオプションです。

> **Note**
>
> UseDeclareFetch を有効にすると、ドライバーは DSN または接続文字列で設定された値に関係なく、ストリーミングカーソルモードを無効にし、StreamingCursorRows を 0 にオーバーライドします。詳細については、「StreamingCursorRows」を参照してください。

UseUnicode

- デフォルト値: 0
- データ型 – ブール型

ドライバーが Redshift データを Unicode または通常の SQL タイプとして返すかどうかを指定するブール値。

- 1 | TRUE: ドライバーは、文字データ型に対してワイド SQL 型を返します。
  - SQL_CHAR の代わりに SQL_WCHAR が返されます。

  - SQL_VARCHAR の代わりに SQL_WVARCHAR が返されます。
  - SQL_LONGVARCHAR の代わりに SQL_WLONGVARCHAR が返されます。
- 0 | FALSE: ドライバーは、文字データ型に対して通常の SQL 型を返します。
  - SQL_WCHAR の代わりに SQL_CHAR が返されます。
  - SQL_WVARCHAR の代わりに SQL_VARCHAR が返されます。
  - SQL_WLONGVARCHAR の代わりに SQL_LONGVARCHAR が返されます。

このパラメータはオプションです。ドライバーバージョン 2.1.15 以降で有効です。

web_identity_token

- デフォルト値 - なし
- データ型 - 文字列

ID プロバイダーによって提供される OAUTH トークン。JWT プラグインで使用されます。

BasicJwtCredentialsProvider に plugin_name パラメータを設定する場合、このパラメータは必須です。

以前のバージョンの ODBC ドライバー

Amazon Redshift JDBC ドライバーバージョン 2.x より前のバージョンをダウンロードするのは、そのバージョンのドライバーがツールに必要な場合に限ります。

Microsoft Windows 用の以前の ODBC ドライバーバージョンを使用する

以下は、Microsoft Windows 用の以前のバージョンの Amazon Redshift ODBC ドライバーバージョン 2.x です。

- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.2.3.0/AmazonRedshiftODBC64-2.2.3.0.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.2.2.0/AmazonRedshiftODBC64-2.2.2.0.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.2.1.0/AmazonRedshiftODBC64-2.2.1.0.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.2.0.0/AmazonRedshiftODBC64-2.2.0.0.msi

- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.17.0/AmazonRedshiftODBC64-2.1.17.0.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.16.0/AmazonRedshiftODBC64-2.1.16.0.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.15.0/AmazonRedshiftODBC64-2.1.15.0.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.14.0/AmazonRedshiftODBC64-2.1.14.0.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.13.0/AmazonRedshiftODBC64-2.1.13.0.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.12.0/AmazonRedshiftODBC64-2.1.12.0.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.11.0/AmazonRedshiftODBC64-2.1.11.0.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.10.0/AmazonRedshiftODBC64-2.1.10.0.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.9.0/AmazonRedshiftODBC64-2.1.9.0.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.9.0/AmazonRedshiftODBC64-2.1.9.0.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.8.0/AmazonRedshiftODBC64-2.1.8.0.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.7.0/AmazonRedshiftODBC64-2.1.7.0.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.6.0/AmazonRedshiftODBC64-2.1.6.0.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.4.0/AmazonRedshiftODBC64-2.1.4.0.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.3.0/AmazonRedshiftODBC64-2.1.3.0.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.2.0/AmazonRedshiftODBC64-2.1.2.0.msi

- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.1.0/AmazonRedshiftODBC64-2.1.1.0.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.0.0/AmazonRedshiftODBC64-2.1.0.0.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.1.0/AmazonRedshiftODBC64-2.0.1.0.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.0.11/AmazonRedshiftODBC64-2.0.0.11.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.0.9/AmazonRedshiftODBC64-2.0.0.9.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.0.8/AmazonRedshiftODBC64-2.0.0.8.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.0.7/AmazonRedshiftODBC64-2.0.0.7.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.0.6/AmazonRedshiftODBC64-2.0.0.6.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.0.5/AmazonRedshiftODBC64-2.0.0.5.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.0.3/AmazonRedshiftODBC64-2.0.0.3.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.0.1/AmazonRedshiftODBC64-2.0.0.1.msi

旧バージョンの ODBC ドライバー (Linux 用)

以下は、Linux 用の以前のバージョンの Amazon Redshift ODBC ドライバーバージョン 2.x です。

- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.2.3.0/AmazonRedshiftODBC-64-bit-2.2.3.0.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.2.3.0/AmazonRedshiftODBC-64-bit-2.2.3.0.aarch64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.2.3.0/AmazonRedshiftODBC-64-bit-2.2.3.0.amd64.deb

- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.2.3.0/AmazonRedshiftODBC-64-bit-2.2.3.0.arm64.deb
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.2.2.0/AmazonRedshiftODBC-64-bit-2.2.2.0.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.2.2.0/AmazonRedshiftODBC-64-bit-2.2.2.0.aarch64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.2.2.0/AmazonRedshiftODBC-64-bit-2.2.2.0.amd64.deb
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.2.2.0/AmazonRedshiftODBC-64-bit-2.2.2.0.arm64.deb
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.2.1.0/AmazonRedshiftODBC-64-bit-2.2.1.0.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.2.1.0/AmazonRedshiftODBC-64-bit-2.2.1.0.aarch64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.2.1.0/AmazonRedshiftODBC-64-bit-2.2.1.0.amd64.deb
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.2.1.0/AmazonRedshiftODBC-64-bit-2.2.1.0.arm64.deb
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.2.0.0/AmazonRedshiftODBC-64-bit-2.2.0.0.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.2.0.0/AmazonRedshiftODBC-64-bit-2.2.0.0.aarch64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.2.0.0/AmazonRedshiftODBC-64-bit-2.2.0.0.amd64.deb
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.2.0.0/AmazonRedshiftODBC-64-bit-2.2.0.0.arm64.deb
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.17.0/AmazonRedshiftODBC-64-bit-2.1.17.0.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.17.0/AmazonRedshiftODBC-64-bit-2.1.17.0.aarch64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.16.0/AmazonRedshiftODBC-64-bit-2.1.16.0.x86_64.rpm

- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.16.0/AmazonRedshiftODBC-64-bit-2.1.16.0.aarch64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.15.0/AmazonRedshiftODBC-64-bit-2.1.15.0.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.15.0/AmazonRedshiftODBC-64-bit-2.1.15.0.aarch64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.14.0/AmazonRedshiftODBC-64-bit-2.1.14.0.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.14.0/AmazonRedshiftODBC-64-bit-2.1.14.0.aarch64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.13.0/AmazonRedshiftODBC-64-bit-2.1.13.0.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.13.0/AmazonRedshiftODBC-64-bit-2.1.13.0.aarch64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.12.0/AmazonRedshiftODBC-64-bit-2.1.12.0.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.12.0/AmazonRedshiftODBC-64-bit-2.1.12.0.aarch64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.11.0/AmazonRedshiftODBC-64-bit-2.1.11.0.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.11.0/AmazonRedshiftODBC-64-bit-2.1.11.0.aarch64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.10.0/AmazonRedshiftODBC-64-bit-2.1.10.0.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.9.0/AmazonRedshiftODBC-64-bit-2.1.9.0.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.8.0/AmazonRedshiftODBC-64-bit-2.1.8.0.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.7.0/AmazonRedshiftODBC-64-bit-2.1.7.0.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.6.0/AmazonRedshiftODBC-64-bit-2.1.6.0.x86_64.rpm

- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.4.0/AmazonRedshiftODBC-64-bit-2.1.4.0.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.3.0/AmazonRedshiftODBC-64-bit-2.1.3.0.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.2.0/AmazonRedshiftODBC-64-bit-2.1.2.0.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.1.0/AmazonRedshiftODBC-64-bit-2.1.1.0.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.0.0/AmazonRedshiftODBC-64-bit-2.1.0.0.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.1.0/AmazonRedshiftODBC-64-bit-2.0.1.0.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.0.11/AmazonRedshiftODBC-64-bit-2.0.0.11.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.0.9/AmazonRedshiftODBC-64-bit-2.0.0.9.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.0.8/AmazonRedshiftODBC-64-bit-2.0.0.8.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.0.7/AmazonRedshiftODBC-64-bit-2.0.0.7.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.0.6/AmazonRedshiftODBC-64-bit-2.0.0.6.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.0.5/AmazonRedshiftODBC-64-bit-2.0.0.5.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.0.3/AmazonRedshiftODBC-64-bit-2.0.0.3.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.0.1/AmazonRedshiftODBC-64-bit-2.0.0.1.x86_64.rpm

旧バージョンの ODBC ドライバー (Apple macOS 用)

以下は、Apple macOS 用の以前のバージョンの Amazon Redshift ODBC ドライバーバージョン 2.x です。

- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.2.3.0/AmazonRedshiftODBC-64-bit.2.2.3.0.universal.pkg
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.2.2.0/AmazonRedshiftODBC-64-bit.2.2.2.0.universal.pkg
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.2.1.0/AmazonRedshiftODBC-64-bit.2.2.1.0.universal.pkg
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.2.0.0/AmazonRedshiftODBC-64-bit.2.2.0.0.universal.pkg
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.17.0/AmazonRedshiftODBC-64-bit.2.1.17.0.universal.pkg
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.16.0/AmazonRedshiftODBC-64-bit.2.1.16.0.universal.pkg
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.15.0/AmazonRedshiftODBC-64-bit.2.1.15.0.universal.pkg
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.14.0/AmazonRedshiftODBC-64-bit.2.1.14.0.universal.pkg
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.13.0/AmazonRedshiftODBC-64-bit.2.1.13.0.universal.pkg
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.1.12.0/AmazonRedshiftODBC-64-bit.2.1.12.0.universal.pkg

ODBC ドライバーバージョン 2.x と 1.x の違い

このセクションでは、ODBC 1.x ドライバーと ODBC 2.x ドライバーの違いについて説明し、2.x ドライバーへの移行方法に関するガイダンスを示します。アプリケーションに影響を与える可能性のある変更とその対処方法について説明します。

対処が必要となる主な違い

次のアクションにより、最も一般的な移行の問題が解決されます。ほとんどのアプリケーションでは、これらのサブセットのみが必要です。

DSN と接続文字列

- 削除または名前が変更された可能性のあるドライバーオプションについては、DSN を確認してください。一部の ODBC 1.x ドライバーオプションは、ODBC 2.x ドライバーではサポートされてい

ません。その他のオプションは、名前が変更されました。詳細については、「接続文字列と DSN オプション」を参照してください。

- アプリケーションがワイド文字タイプコード (**SQL_WVARCHAR**、**SQL_WCHAR**) に依存している場合は、**UseUnicode=true** を設定します。ODBC 2.x ドライバーでは、UseUnicode はデフォルトで false になり、ナロー文字タイプが報告されます。
- ログ記録設定を **[ODBC]** セクションに移動します。Linux および macOS では、個別の DSN セクションではなく、odbc.ini のグローバル [ODBC] セクションで LogLevel と LogPath を設定する必要があります。

クエリとスキーマ

- **SQLTables** タイプフィルターに **EXTERNAL TABLE** を含めます。ODBC 2.x では、Amazon Redshift Spectrum テーブルとデータ共有テーブルの両方が EXTERNAL TABLE として報告されます。タイプで SQLTables をフィルタリングする場合は、EXTERNAL TABLE を追加してこれらのオブジェクトを表示し続けます。または、次の項目で説明するように、EnableTableTypes オプションを 0 に設定して、詳細なテーブルタイプ情報を一般的な TABLE テーブルタイプと VIEW テーブルタイプに正規化します。
- 詳細なテーブルタイプはデフォルトで有効になっています。ODBC 2.x では、EnableTableTypes オプションがデフォルトで有効になります。1.x ドライバーでは、このオプションはデフォルトで無効になっていました。2.x のデフォルトでは、SQLTables は SYSTEM TABLE、SYSTEM VIEW、EXTERNAL TABLE、LOCAL TEMPORARY などの詳細なタイプを報告します。すべてのテーブルを汎用タイプ TABLE として報告し、すべてのビューを VIEW として報告するには、EnableTableTypes を 0 に設定します。
- 間隔データ型をサポートしていないアプリケーションに対しては、**INTERVAL** を **VARCHAR** にキャストします。一部のクライアントは ODBC 間隔タイプをサポートしていません。これらのクライアントの場合、次のようにクエリで列をキャストします: SELECT col::VARCHAR FROM ...

アプリケーションコード

- クエリのタイムアウト設定を検証します。ODBC 2.x は、ODBC 仕様に従って正しく SQL_ATTR_QUERY_TIMEOUT を強制します。ODBC 1.x では、エラーを表示することなく、この設定は無視されました。以前には成功した長時間実行クエリが、タイムアウトエラーで失敗する場合があります。必要に応じてタイムアウト値を確認して調整してください。
- data-at-execution パラメータのデータの長さを指定します。 2.x は SQL_NEED_LONG_DATA_LEN を Y として報告します (1.x は N として報告しました)。SQL_DATA_AT_EXEC パラメータをバイン

ドするアプリケーションでは、StrLen_or_IndPtr で合計データ長を事前に指定する必要があります。

接続文字列と DSN オプション

次の表は、名前が変更されたか、ODBC 2.x ドライバーで直接同等のオプションがある ODBC 1.x ドライバーのオプションを示しています。

1.x オプション	2.x の同等オプション	注意事項
MaxLongVarChar	MaxLongVarcharSize	デフォルトは 8190 から 65535 へ変更されました。
ConnectionTimeout	LoginTimeout	接続タイムアウトは同じで、名前が変更されました。デフォルトは 0 (タイムアウトなし) です。
VpcEndpointUrl	vpc_endpoint_url	
SSLCertPath	TrustStore -または-CaFile	サーバーの検証に使用される CA 証明書へのパス。Windows では、これを [DSN セットアップ] ダイアログの [信頼ストア] フィールドで設定します。このダイアログには CaFile フィールドがないためです。両方が設定されている場合、TrustStore が優先されます。

現在の 2.x ドライバーでは、次の 1.x オプションはサポートされていません。2.x ドライバーはそれらを無視するため、接続には影響しません。DSN からのそれらの削除はオプションですが、混乱を避けるために削除することをお勧めします。

- SingleRowMode – クライアントメモリを制限するには、代わりに StreamingCursorRows または UseDeclareFetch を使用します。
- UseSystemTrustStore – サポートされていません。Windows では、1.x ドライバーは Windows システム証明書ストアに照らしてサーバー証明書を検証することができました。2.x ドライバーは CA 証明書ファイルに照らして検証を行います。デフォルトでは、バンドルされた Amazon

Redshift ルート証明書を使用しますが、ユーザーが TrustStore または CaFile で指定したファイルを使用することもできます。

- TextAsLongVarchar、CheckCertRevocation、EnableAwsSdkLogs、UseLogPrefix、Locale、– 現在のリリースでは同等のものはありません。Amazon Redshift チームは、将来のリリースでこれらの同等オプションまたは代替オプションを検討中です。

サポートされる 2.x オプションの詳細なリストについては、「ODBC ドライバーオプション」を参照してください。

データ型の変更

次の表は、ODBC 仕様に準拠するために ODBC 2.x で変更された Amazon Redshift データ型マッピングを示しています。ほとんどのアプリケーションは、型コードではなくインデックスまたは名前で列をバインドするため、影響を受けません。

Amazon Redshift データタイプ	ODBC 1.x	ODBC 2.x
DOUBLE PRECISION	SQL_FLOAT (6)	SQL_DOUBLE (8)
INTERVAL YEAR TO MONTH	SQL_VARCHAR (12)、テキスト	SQL_INTERVAL_YEAR_TO_MONTH (107)、間隔構造
INTERVAL DAY TO SECOND	SQL_VARCHAR (12)、テキスト	SQL_INTERVAL_DAY_TO_SECOND (110)、間隔構造
VARCHAR / CHAR / TEXT	ワイドタイプ (SQL_WVARCHAR 、SQL_WCHAR 、SQL_WLONGVARCHAR)、デフォルトは UseUnicode=true	ナロータイプ (SQL_VARCHAR 、SQL_CHAR、SQL_LONGVARCHAR)。UseUnicode=true を設定してワイドタイプを復元します。
GEOMETRY / GEOGRAPHY (SQL_C_BINARY として)	16 進エンコードされた ASCII 文字列	Raw バイナリバイト

SQLColumns によって返される COLUMN_SIZE の値は、GEOMETRY、GEOGRAPHY、および SUPER データ型でも変更され、ODBC 仕様に従ってサイズ未設定の列を示すために NULL を返すようになりました。COLUMN_SIZE に基づいてバッファを割り当てるアプリケーションは、デフォルトのバッファサイズを使用して NULL を処理する必要があります。

移行後のトラブルシューティング

次の表は、移行後に発生する可能性がある、よくある問題とその解決方法を示しています。

症状	原因	対応方法
外部テーブルがスキーマブラウザに表示されない	Spectrum テーブルとデータ共有テーブルは EXTERNAL TABLE として報告されます。	SQLTables タイプフィルターに EXTERNAL TABLE を含めるか、EnableTableTypes を 0 に設定して、詳細なテーブルタイプを一般的な TABLE タイプと VIEW タイプに正規化します。
間隔列で pyodbc エラーが発生する	pyodbc は ODBC 間隔タイプをサポートしていません。	クエリで間隔を VARCHAR にキャストします。
文字データが予期しない文字として表示される (文字化け)	UseUnicode のデフォルトは false に変更されました。ワイド文字 (UTF-16) データを予期しているアプリケーションは、ナロー文字のバイトをワイドペアとして誤って解釈し、文字化けの出力を生成する場合があります。データ自体は変更されていません。	DSN で UseUnicode=true を設定するか、アプリケーションを更新して列を SQL_C_WCHAR ではなく SQL_C_CHAR としてバインドします。
長時間実行されるクエリがタイムアウトエラーを返す	SQL_ATTR_QUERY_TIMEOUT が適用されるようになりました。ODBC 1.x では、エラーを表示することなくこの設定は無視されました。	DSN から QueryTimeout を増減するか、アプリケーションで SQL_ATTR_QUERY_TIMEOUT を 0 に設定します。

症状	原因	対応方法
SQL_ATTR_CURRENT_CATALOG を設定すると HY011 が返される	この属性はオープン接続では設定できません。オープン接続でのデータベースの切り替えはサポートされていません。	接続する前に属性を設定するか、ターゲットデータベースで接続を閉じて再度開きます。
デュアルスタック環境で接続がハングする	バージョン 2.1.16 以前での制限です。	2.1.17 以降にアップグレード。
エラーメッセージ解析で予期しない結果が返される	メッセージ形式が変更されました。SQLSTATE コードに変更はありません。	メッセージテキストの代わりに SQLSTATE コードを解析します。
Data-at-execution パラメータのバインディング動作が異なる	SQL_NEED_LONG_DATA_LEN は FALSE から TRUE に変更されました。SQL_DATA_AT_EXEC を使用するアプリケーションは、合計データ長を事前に指定する必要があります。	SQL_DATA_AT_EXEC パラメータをバインドするときに、StrLen_or_IndPtr で長さの値を設定します。
ドライバーログが生成されない	LogLevel および LogPath は、個別の DSN セクションではなく、odbc.ini の [ODBC] グローバルセクションにある必要があります。	ログ記録設定を DSN から [ODBC] セクションに移動します。

古い ODBC 2.x バージョンからのアップグレード

古い ODBC 2.x バージョンを既に実行している場合は、最新のリリースにアップグレードしてください。すべてのバージョンでの機能強化とバグ修正の詳細なリストについては、GitHub の「Amazon Redshift ODBC ドライバー変更ログ」を参照してください。

詳細情報

- Amazon Redshift での ODBC ドライバーのバージョン 2.x による接続の設定

- ODBC ドライバーオプション
- Amazon Redshift ODBC ドライバー変更ログ

ODBC ドライバーバージョン 1.x の接続の設定

ODBC 接続を使用すると、多くのサードパーティー SQL クライアントツールおよびアプリケーションから Amazon Redshift クラスターに接続できます。これを行うには、クライアントコンピュータまたは Amazon EC2 インスタンスで接続をセットアップします。クライアントツールで JDBC がサポートされる場合、JDBC に用意された設定が簡単なため、ODBC ではなくその接続タイプを使用することを選択できます。クライアントツールで JDBC がサポートされていない場合は、このセクションの手順に従って ODBC 接続を設定します。

Amazon Redshift には、Linux、Windows、macOS オペレーティングシステム用の 64 ビット ODBC ドライバーが用意されています。32 ビット ODBC ドライバの提供は終了しました。今後、緊急のセキュリティパッチを除きアップデートはリリースされません。

ODBC ドライバーの機能と前提条件に関する最新情報については、Amazon Redshift ODBC ドライバーのリリースノートを参照してください。

Amazon Redshift ODBC ドライバーのインストールおよび設定の詳細については、Amazon Redshift ODBC connector installation and configuration guide を参照してください。

トピック

- ODBC URL の取得
- Microsoft Windows での Amazon Redshift ODBC ドライバーの使用
- Linux での Amazon Redshift ODBC ドライバーの使用
- macOS での Amazon Redshift ODBC ドライバーの使用
- ODBC ドライバーオプション
- 以前のバージョンの ODBC ドライバー

ODBC URL の取得

Amazon Redshift では、Amazon Redshift コンソールに、クラスターの ODBC URL が表示されます。この URL には、クライアントコンピュータとデータベースの間の接続をセットアップするために必要な情報が含まれています。

ODBC の URL 形式を次に示します。`Driver={driver};Server=endpoint;Database=database_name;UID=user_name;PWD=pass`

前述の形式のフィールドには、次の値があります。

フィールド	値
`Driver`	使用する 64 ビット ODBC ドライバーの名前: [Amazon Redshift (x64)]。 32 ビット ODBC ドライバーの名前: [Amazon Redshift (x86)]。
`Server`	Amazon Redshift クラスターのエンドポイント。
`Database`	クラスター用に作成したデータベース。
`UID`	データベースに接続するアクセス許可を持つユーザーアカウントのユーザー名。この値は Amazon Redshift に対するアクセス許可ではなくデータベースのアクセス許可ですが、クラスターの起動時にセットアップした管理者ユーザーアカウントを使用可能です。
`PWD`	データベースに接続するユーザーアカウントのパスワード。
`Port`	クラスターを起動したときに指定したポート番号。ファイアウォールがある場合、このポートが開いていて使用できることを確認します。

前述のテーブルのフィールドには、次の特殊文字を含めることができます。

```
[]{}(),;?*=!@
```

これらの特殊文字を使用する場合は、値を中かっこで囲む必要があります。例えば、接続文字列内のパスワード値 `Your;password123` は `PWD={Your;password123};` として表されます。

`Field=value` ペアはセミコロンで区切られるため、`}` と `;` の間に任意の数のスペースがある組み合わせは、`Field={value};` ペアの終わりと見なされます。フィールド値のシーケンス `};` を避けることをお勧めします。例えば、パスワードの値を `PWD={This is a passwor} ;d};` に設定するとパスワードは `This is a passwor} ;` となり、URL はエラーになります。

ODBC URL の例を次に示します。

```
Driver={Amazon Redshift (x64)};
```

```
                    Server=examplecluster.abc123xyz789.us-
west-2.redshift.amazonaws.com;
                    Database=dev;
                    UID=adminuser;
                    PWD=insert_your_admin_user_password_here;
                    Port=5439
```

ODBC 接続の取得方法については、「クラスター接続文字列を検索する」を参照してください。

Microsoft Windows での Amazon Redshift ODBC ドライバーの使用

Amazon Redshift データウェアハウスにアクセスするクライアントコンピュータに Amazon Redshift ODBC ドライバーをインストールします。ドライバーをインストールするコンピューターは、それぞれ以下の最小システム要件を満たしている必要があります。最小システム要件については、Amazon Redshift ODBC コネクターのインストールおよび設定ガイドを参照してください。

トピック

- Amazon Redshift ODBC ドライバーをダウンロードしてインストールする
- ODBC 接続用のシステム DSN エントリの作成

Amazon Redshift ODBC ドライバーをダウンロードしてインストールする

Windows オペレーティングシステム用の Amazon Redshift ODBC ドライバーをダウンロードするには、次の手順に従います。Amazon Redshift での使用が認定されている、固有のドライバーを必要とするサードパーティーアプリケーションを実行している場合にのみ、これらとは異なるドライバーを使用できます。

ODBC ドライバーをインストールするには

1. 使用する SQL クライアントツールまたはアプリケーションのシステムアーキテクチャに応じて、次のいずれかをダウンロードします。

   - 64 ビット ODBC ドライバーバージョン 1.6.3

     このドライバーの名前は、Amazon Redshift (x64) です。

   - 32 ビット ODBC ドライバーバージョン 1.4.52

     このドライバーの名前は、Amazon Redshift (x86) です。32 ビット ODBC ドライバの提供は終了しました。今後、緊急のセキュリティパッチを除きアップデートはリリースされません。

> **Note**
>
> SQL クライアントツールまたはアプリケーションのシステムアーキテクチャに対応する MSI パッケージをダウンロードします。例えば、SQL クライアントツールが 64 ビットである場合は、64 ビットドライバーをインストールします。

次に、Amazon Redshift ODBC および JDBC ドライバーのライセンス契約をダウンロードして確認します。

2. .msi ファイルをダブルクリックし、ウィザードの手順に従ってドライバーをインストールします。

ODBC 接続用のシステム DSN エントリの作成

ODBC ドライバーをダウンロードしてインストールした後、クライアントマシンか Amazon EC2 インスタンスにデータソース名 (DSN) エントリを追加します。SQL クライアントツールは、このデータソースを使用して Amazon Redshift データベースに接続します。

ユーザー DSN の代わりにシステム DSN を作成することをお勧めします。アプリケーションによっては、別のユーザーアカウントを使用してデータをロードします。これらのアプリケーションは、別のユーザーアカウントで作成されたユーザー DSN を検出できない場合があります。

> **Note**
>
> AWS Identity and Access Management (IAM) 認証情報または ID プロバイダー (IdP) 認証情報を使用した認証には、追加の手順が必要です。詳細については、「ステップ 5: IAM 認証情報を使用するように JDBC または ODBC 接続を設定する」を参照してください。

システム DSN エントリを作成する方法については、Amazon Redshift ODBC connector installation and configuration guide を参照してください。

Windows で ODBC 接続用のシステム DNS エントリを作成するには

1. [スタート] メニューで、[ODBC Data Sources (ODBC データソース)] を開きます。

Amazon Redshift への接続に使用するクライアントアプリケーションと同じビット数を持つ ODBC Data Source Administrator を選択してください。

2. [ODBC Data Source Administrator] (ODBC データソース管理者) で、[Driver] (ドライバー) タブを開き、ドライバーフォルダを探します。

   - [Amazon Redshift ODBC Driver (64-bit)] (Amazon Redshift ODBC ドライバー (64 ビット))
   - [Amazon Redshift ODBC Driver (32-bit)] (Amazon Redshift ODBC ドライバー (32 ビット))

3. コンピュータのすべてのユーザーに対してドライバーを設定する場合は [System DSN (システム DSN)] タブを選択し、使用するユーザーアカウントのみに対してドライバーを設定する場合は [User DSN (ユーザー DSN)] タブを選択します。
4. [Add] を選択します。[Create New Data Source (新しいデータソースの作成)] ウィンドウが開きます。
5. Amazon Redshift ODBC ドライバーを選択し、[Finish (完了)] を選択します。[Amazon Redshift ODBC Driver DSN Setup (ODBC ドライバー DSN セットアップ)] ウィンドウが開きます。
6. [接続設定] で以下の情報を入力します。

   データソース名

   データソースの名前を入力します。クラスターへの接続を作成するときに、後でデータソースを識別するためにどのような名前でも使用できます。例えば、「Amazon Redshift 入門ガイド」に従っているのであれば、この DSN に関連付けるクラスターを覚えやすくするために `exampleclusterdsn` と入力します。

   サーバー

   Amazon Redshift クラスターのエンドポイントを指定します。この情報は、クラスターの詳細ページの Amazon Redshift コンソールで見つけることができます。詳細については、「Amazon Redshift での接続の設定」を参照してください。

   ポート

   データベースが使用するポート番号を入力します。クラスターが起動または変更されたときに使用するように設定されたポートを使用します。

   データベース

Amazon Redshift データベースの名前を入力します。データベース名を指定せずにクラスターを起動した場合は、*dev*を入力します。それ以外の場合は、起動プロセスで選択した名前を使用します。「Amazon Redshift 入門ガイド」に従っているのであれば、*dev* と入力します。

7. [Authentication (認証)] で、標準認証または IAM 認証を設定する設定オプションを指定します。認証オプションの詳細については、Amazon Redshift ODBC Connector Installation and Configuration Guide の「Configuring Authentication on Windows」を参照してください。
8. [SSL Settings (SSL 設定)] で次の値を指定します。

   SSL 認証

   Secure Sockets Layer (SSL) を処理するモードを選択します。テスト環境では、`prefer`を使用できます。ただし、本番稼働環境の場合、および安全なデータ交換が必要な場合は、`verify-ca`または `verify-full` を使用します。Windows での SSL の使用の詳細については、Amazon Redshift ODBC Connector Installation and Configuration Guide の「Configuring SSL Verification on Windows」を参照してください。
9. [Additional Options (追加オプション)] で、クエリ結果を SQL クライアントツールまたはアプリケーションに返す方法に関するオプションを指定します。詳細については、Amazon Redshift ODBC Connector Installation and Configuration Guide の「Configuring Additional Options on Windows」を参照してください。
10. [Logging Options (ログ記録オプション)] で以下の値を指定します。詳細については、Amazon Redshift ODBC Connector Installation and Configuration Guide の「Configuring Logging Options on Windows」を参照してください。

    次に、[OK] を選択します。
11. [Data Type Options (データ型オプション)] で、データ型の値を指定します。詳細については、Amazon Redshift ODBC Connector Installation and Configuration Guide の「Configuring Data Type Options on Windows」を参照してください。

    次に、[OK] を選択します。
12. [Test] を選択します。クライアントコンピュータが Amazon Redshift データベースに接続できる場合は、[Connection successful (接続成功)] というメッセージが表示されます。

    クライアントコンピューターがデータベースに接続できない場合、考えられる問題をトラブルシューティングすることができます。詳細については、「Amazon Redshift での接続の問題のトラブルシューティング」を参照してください。

13. Windows で TCP キープアライブを設定して、接続がタイムアウトしないようにします。Windows で TCP キープアライブを設定する方法については、Amazon Redshift ODBC Connector Installation and Configuration Guide を参照してください。
14. トラブルシューティングに役立つように、ログ記録を設定します。Windows でログ記録を設定する方法については、Amazon Redshift ODBC Connector Installation and Configuration Guide を参照してください。

Linux での Amazon Redshift ODBC ドライバーの使用

Amazon Redshift データウェアハウスにアクセスするクライアントコンピュータに Amazon Redshift ODBC ドライバーをインストールします。ドライバーをインストールするコンピューターは、それぞれ以下の最小システム要件を満たしている必要があります。最小システム要件については、Amazon Redshift ODBC コネクターのインストールおよび設定ガイドを参照してください。

トピック

- Amazon Redshift ODBC ドライバーをダウンロードしてインストールする
- ODBC ドライバーマネージャーを使用してドライバーを設定する

Amazon Redshift ODBC ドライバーをダウンロードしてインストールする

このセクションのステップを使用して、サポートされている Linux ディストリビューションに Amazon Redshift ODBC ドライバーをダウンロードしてインストールします。インストールプロセスでは、以下のディレクトリにドライバーファイルがインストールされます。

- `/opt/amazon/redshiftodbc/lib/64` (64 ビットドライバーの場合)
- `/opt/amazon/redshiftodbc/ErrorMessages`
- `/opt/amazon/redshiftodbc/Setup`
- `/opt/amazon/redshiftodbc/lib/32` (32 ビットドライバーの場合)

Amazon Redshift ODBC ドライバーをインストールするには

1. 使用する SQL クライアントツールまたはアプリケーションのシステムアーキテクチャに応じて、次のいずれかをダウンロードします。
   - 64 ビット RPM ドライバーバージョン 1.6.3
   - 64 ビット Debian ドライバーバージョン 1.6.3

- 32 ビットドライバーバージョン 1.4.52。

これらのドライバーの名前はすべて Amazon Redshift ODBC ドライバーです。32 ビット ODBC ドライバの提供は終了しました。今後、緊急のセキュリティパッチを除きアップデートはリリースされません。

> **Note**
>
> SQL クライアントツールまたはアプリケーションのシステムアーキテクチャに対応するパッケージをダウンロードします。例えば、クライアントツールが 64 ビットである場合は、64 ビットドライバーをインストールします。

次に、Amazon Redshift ODBC および JDBC ドライバーのライセンス契約をダウンロードして確認します。

2. パッケージをダウンロードした場所に移動し、以下のいずれかのコマンドを実行します。使用している Linux ディストリビューションに対応するコマンドを使用します。

   - RHEL および CentOS オペレーティングシステムでは、次のコマンドを実行します。

     ```
     yum -nogpgcheck localinstall RPMFileName
     ```

     RPMFileName を RPM パッケージファイル名に置き換えます。例えば、次のコマンドは 64 ビットドライバーのインストールを示しています。

     ```
     yum -nogpgcheck localinstall AmazonRedshiftODBC-64-bit-1.x.xx.xxxx-x.x86_64.rpm
     ```

   - SLES では、以下のコマンドを実行します。

     ```
     zypper install RPMFileName
     ```

     RPMFileName を RPM パッケージファイル名に置き換えます。例えば、次のコマンドは 64 ビットドライバーのインストールを示しています。

     ```
     zypper install AmazonRedshiftODBC-1.x.x.xxxx-x.x86_64.rpm
     ```

   - Debian では、以下のコマンドを実行します。

```
sudo apt install ./DEBFileName.deb
```

DEBFileName.deb を Debian パッケージのファイル名に置き換えます。例えば、次のコマンドは 64 ビットドライバーのインストールを示しています。

```
sudo apt install ./AmazonRedshiftODBC-1.x.x.xxxx-x.x86_64.deb
```

> **Important**
>
> ドライバーのインストールが完了したら、ドライバーをシステムで使用できるように設定します。ドライバー設定の詳細については、「ODBC ドライバーマネージャーを使用してドライバーを設定する」を参照してください。

ODBC ドライバーマネージャーを使用してドライバーを設定する

Linux オペレーティングシステムでは、ODBC ドライバーマネージャーを使用して ODBC 接続を設定します。ODBC ドライバーマネージャーは設定ファイルを使用して ODBC データソースおよびドライバーを定義および設定します。使用する ODBC ドライバーマネージャーは、使用するオペレーティングシステムによって異なります。Linux の場合は、unixODBC ドライバーマネージャーです。

Amazon Redshift ODBC ドライバーの設定がサポートされている ODBC ドライバーマネージャーの詳細については、Linux オペレーティングシステム向けの「Linux での Amazon Redshift ODBC ドライバーの使用」を参照してください。詳細については、「Amazon Redshift ODBC connector installation and configuration guide」の「Specifying ODBC Driver Managers on Non- Windows Machines」を参照してください。

Amazon Redshift ODBC ドライバーを設定するには、`amazon.redshiftodbc.ini`、`odbc.ini`、および `odbcinst.ini` の 3 つのファイルが必要です。

デフォルトの場所にインストールした場合、`amazon.redshiftodbc.ini`設定ファイルは以下のいずれかのディレクトリにあります。

- `/opt/amazon/redshiftodbc/lib/64` (Linux オペレーティングシステム上の 64 ビットドライバーの場合)
- `/opt/amazon/redshiftodbc/lib/32` (Linux オペレーティングシステム上の 32 ビットドライバーの場合)

さらに、Linux では、/opt/amazon/redshiftodbc/Setup の下に odbc.ini と odbcinst.ini のサンプルファイルがあります。これらのファイルは、Amazon Redshift ODBC ドライバーとデータソース名 (DSN) の設定例として使用できます。

設定ファイルに Amazon Redshift ODBC ドライバーのインストールディレクトリを使用することは推奨されません。Setup ディレクトリのサンプルファイルは例示のみを目的としています。Amazon Redshift ODBC ドライバーを後で再インストールするか、新しいバージョンにアップグレードすると、インストールディレクトリが上書きされます。その後、これらのファイルに加えた変更はすべて失われます。

これを回避するには、インストールディレクトリ以外のディレクトリに amazon.redshiftodbc.ini ファイルをコピーします。このファイルをユーザーのホームディレクトリにコピーする場合は、隠しファイルにするためにファイル名の先頭にピリオド (.) を追加します。

odbc.ini および odbcinst.ini ファイルについては、ユーザーのホームディレクトリにある設定ファイルを使用するか、別のディレクトリに新しいバージョンを作成します。デフォルトでは、Linux オペレーティングシステムには、ユーザーのホームディレクトリ (/home/$USER または ~/) に odbc.ini ファイルと odbcinst.ini ファイルが必要です。これらの既定のファイルは隠しファイルで、各ファイル名の前にドット (.) が表示されます。これらのファイルは、-aフラグを使用してディレクトリの内容をリストした場合にのみ表示されます。

odbc.ini ファイルと odbcinst.ini ファイルに対してどちらのオプションを選択しても、ファイルを変更してドライバーと DSN 設定情報を追加します。新しいファイルを作成する場合は、これらの設定ファイルの場所を指定する環境変数を設定する必要もあります。

デフォルトでは、ODBC ドライバーマネージャーは、隠しバージョンの odbc.ini および odbcinst.ini 設定ファイル (.odbc.ini と .odbcinst.iniという名前) をホームディレクトリに配置します。また、ドライバーのインストールディレクトリの amazon.redshiftodbc.ini サブフォルダにある /lib ファイルを使用するように設定されています。これらの設定ファイルを他の場所に保存する場合は、ドライバーマネージャーがファイルを見つけることができるように、次に説明する環境変数を設定します。詳細については、Amazon Redshift ODBC connector installation and configuration guide の「Specifying the Locations of the Driver Configuration Files」を参照してください。

Linux オペレーティングシステムでのデータソース名の作成

データソース名 (DSN) を使用してデータストアに接続する場合は、DSN を定義するように odbc.ini ファイルを設定します。odbc.ini ファイルのプロパティを設定して、データストアの接続情報を指定する DSN を作成します。

odbc.ini ファイルの設定方法については、「Amazon Redshift ODBC コネクタのインストールと設定ガイド」の「Windows 以外のマシンでデータソース名を作成する」を参照してください。

Linux オペレーティングシステムでは次の形式を使用します。

```
[ODBC Data Sources]
driver_name=dsn_name

[dsn_name]
Driver=path/driver_file

Host=cluster_endpoint
Port=port_number
Database=database_name
locale=locale
```

次に、Linux オペレーティングシステムでの、64 ビット ODBC ドライバーを使用する odbc.ini の設定の例を示します。

```
[ODBC Data Sources]
Amazon_Redshift_x64=Amazon Redshift (x64)

[Amazon Redshift (x64)]
Driver=/opt/amazon/redshiftodbc/lib/64/libamazonredshiftodbc64.so
Host=examplecluster.abc123xyz789.us-west-2.redshift.amazonaws.com
Port=5932
Database=dev
locale=en-US
```

次に、Linux オペレーティングシステムでの、32 ビット ODBC ドライバーを使用する odbc.ini の設定の例を示します。

```
[ODBC Data Sources]
Amazon_Redshift_x32=Amazon Redshift (x86)

[Amazon Redshift (x86)]
Driver=/opt/amazon/redshiftodbc/lib/32/libamazonredshiftodbc32.so
Host=examplecluster.abc123xyz789.us-west-2.redshift.amazonaws.com
Port=5932
Database=dev
locale=en-US
```

Linux オペレーティングシステムで DSN を使用しない接続を設定する

DSN を使用しない接続を介してデータストアに接続するには、odbcinst.ini ファイルでドライバーを定義します。次に、アプリケーションに DSN を使用しない接続文字列を指定します。

この場合の odbcinst.ini ファイルの設定方法については、「Amazon Redshift ODBC connector installation and configuration guide」の「Configuring a DSN-less Connection on a Non-Windows Machine」を参照してください。

Linux オペレーティングシステムでは次の形式を使用します。

```
[ODBC Drivers]
driver_name=Installed
...

[driver_name]
Description=driver_description
Driver=path/driver_file

...
```

次に、Linux オペレーティングシステムで、デフォルトのディレクトリにインストールされた 64 ビットドライバーでの odbcinst.ini 設定の例を示します。

```
[ODBC Drivers]
Amazon Redshift (x64)=Installed

[Amazon Redshift (x64)]
Description=Amazon Redshift ODBC Driver (64-bit)
Driver=/opt/amazon/redshiftodbc/lib/64/libamazonredshiftodbc64.so
```

次に、Linux オペレーティングシステムで、デフォルトのディレクトリにインストールされた 32 ビットドライバーでの odbcinst.ini 設定の例を示します。

```
[ODBC Drivers]
Amazon Redshift (x86)=Installed

[Amazon Redshift (x86)]
Description=Amazon Redshift ODBC Driver (32-bit)
Driver=/opt/amazon/redshiftodbc/lib/32/libamazonredshiftodbc32.so
```

環境変数の設定

正しい ODBC ドライバーマネージャーを使用して、正しいドライバーをロードします。これを行うには、ライブラリパス環境変数を設定します。詳細については、Amazon Redshift ODBC connector installation and configuration guide の「Specifying ODBC Driver Managers on Non-Windows Machines」を参照してください。

デフォルトでは、ODBC ドライバーマネージャーは、隠しバージョンの `odbc.ini` および `odbcinst.ini` 設定ファイル (`.odbc.ini` と `.odbcinst.ini`という名前) をホームディレクトリに配置します。また、ドライバーのインストールディレクトリの `amazon.redshiftodbc.ini` サブフォルダにある `/lib` ファイルを使用するように設定されています。これらの設定ファイルを他の場所に保存する場合は、ドライバーマネージャーがファイルを検索できるように環境変数を設定します。詳細については、Amazon Redshift ODBC Connector Installation and Configuration Guide の「Specifying the Locations of the Driver Configuration Files」を参照してください。

接続機能の設定

ODBC 設定には、次の接続機能を設定できます。

- 認証情報を提供し、Amazon Redshift データベースへの接続を認証するように ODBC ドライバーを設定します。
- SSL が有効な Amazon Redshift サーバーに接続している場合は、Secure Sockets Layer (SSL) で有効なソケットに接続するように ODBC ドライバーを設定します。
- プロキシサーバー経由で Amazon Redshift に接続するように ODBC ドライバーを設定します。
- クエリ処理モードを使用するように ODBC ドライバーを設定して、クエリが大量のメモリを消費しないようにします。
- IAM 認証プロセスをプロキシサーバー経由で渡すように ODBC ドライバーを設定します。
- 接続のタイムアウトを防ぐために TCP キープアライブを使用するように ODBC ドライバーを設定します。

これらの接続機能の詳細については、Amazon Redshift ODBC connector installation and configuration guide を参照してください。

macOS での Amazon Redshift ODBC ドライバーの使用

Amazon Redshift データウェアハウスにアクセスするクライアントコンピュータにドライバーをインストールします。ドライバーをインストールするコンピューターは、それぞれ以下の最小システム要

件を満たしている必要があります。最小システム要件については、Amazon Redshift ODBC コネクターのインストールおよび設定ガイドを参照してください。

トピック

- Amazon Redshift ODBC ドライバーをダウンロードしてインストールする
- ODBC ドライバーマネージャーを使用してドライバーを設定する

Amazon Redshift ODBC ドライバーをダウンロードしてインストールする

このセクションのステップを使用して、サポートされているバージョンの macOS に Amazon Redshift ODBC ドライバーをダウンロードしてインストールします。インストールプロセスでは、以下のディレクトリにドライバーファイルがインストールされます。

- `/opt/amazon/redshift/lib/universal`
- `/opt/amazon/redshift/ErrorMessages`
- `/opt/amazon/redshift/Setup`

macOS に Amazon Redshift ODBC ドライバーをインストールするには

1. macOS に Amazon Redshift ODBC ドライバーをインストールするには、macOS ドライバーバージョン 1.6.3 をダウンロードします。

   次に、Amazon Redshift ODBC および JDBC ドライバーのライセンス契約をダウンロードして確認します。
2. AmazonRedshiftODBC.pkg をダブルクリックしてインストーラを実行します。
3. インストーラの手順に従ってドライバーのインストールプロセスを完了します。インストールを実行するには、ライセンス契約の条件に同意する必要があります。

> **Important**
>
> ドライバーのインストールが完了したら、ドライバーをシステムで使用できるように設定します。ドライバー設定の詳細については、「ODBC ドライバーマネージャーを使用してドライバーを設定する」を参照してください。

ODBC ドライバーマネージャーを使用してドライバーを設定する

macOS オペレーティングシステムでは、ODBC ドライバーマネージャーを使用して ODBC 接続を設定します。ODBC ドライバーマネージャーは設定ファイルを使用して ODBC データソースおよびドライバーを定義および設定します。使用する ODBC ドライバーマネージャーは、使用するオペレーティングシステムによって異なります。macOS オペレーションシステムの場合、iODBC ドライバーマネージャーです。

Amazon Redshift ODBC ドライバーの設定がサポートされている ODBC ドライバーマネージャーの詳細については、macOS オペレーティングシステム向けの「macOS での Amazon Redshift ODBC ドライバーの使用」を参照してください。詳細については、「Amazon Redshift ODBC connector installation and configuration guide」の「Specifying ODBC Driver Managers on Non- Windows Machines」を参照してください。

Amazon Redshift ODBC ドライバーを設定するには、`amazon.redshiftodbc.ini`、`odbc.ini`、および `odbcinst.ini` の 3 つのファイルが必要です。

デフォルトの場所にインストールした場合、`amazon.redshiftodbc.ini` 設定ファイルは `/opt/amazon/redshift/lib` にあります。

さらに、macOS では、`/opt/amazon/redshift/Setup` の下に `odbc.ini` と `odbcinst.ini` のサンプルファイルがあります。これらのファイルは、Amazon Redshift ODBC ドライバーとデータソース名 (DSN) の設定例として使用できます。

設定ファイルに Amazon Redshift ODBC ドライバーのインストールディレクトリを使用することは推奨されません。`Setup` ディレクトリのサンプルファイルは例示のみを目的としています。Amazon Redshift ODBC ドライバーを後で再インストールするか、新しいバージョンにアップグレードすると、インストールディレクトリが上書きされます。その後、これらのファイルに加えた変更はすべて失われます。

これを回避するには、インストールディレクトリ以外のディレクトリに `amazon.redshiftodbc.ini` ファイルをコピーします。このファイルをユーザーのホームディレクトリにコピーする場合は、隠しファイルにするためにファイル名の先頭にピリオド (.) を追加します。

`odbc.ini` および `odbcinst.ini` ファイルについては、ユーザーのホームディレクトリにある設定ファイルを使用するか、別のディレクトリに新しいバージョンを作成します。デフォルトでは、macOS オペレーティングシステムには、ユーザーのホームディレクトリ (`/home/$USER` または `~/`) に `odbc.ini` ファイルと `odbcinst.ini` ファイルが必要です。これらの既定のファイルは

隠しファイルで、各ファイル名の前にドット (.) が表示されます。これらのファイルは、-aフラグを使用してディレクトリの内容をリストした場合にのみ表示されます。

odbc.ini ファイルと odbcinst.ini ファイルに対してどちらのオプションを選択しても、ファイルを変更してドライバーと DSN 設定情報を追加します。新しいファイルを作成する場合は、これらの設定ファイルの場所を指定する環境変数を設定する必要もあります。

デフォルトでは、ODBC ドライバーマネージャーは、隠しバージョンの odbc.ini および odbcinst.ini 設定ファイル (.odbc.ini と .odbcinst.iniという名前) をホームディレクトリに配置します。また、ドライバーのインストールディレクトリの amazon.redshiftodbc.ini サブフォルダにある /lib ファイルを使用するように設定されています。これらの設定ファイルを他の場所に保存する場合は、ドライバーマネージャーがファイルを見つけることができるように、次に説明する環境変数を設定します。詳細については、Amazon Redshift ODBC connector installation and configuration guide の「Specifying the Locations of the Driver Configuration Files」を参照してください。

macOS X オペレーティングシステムでのデータソース名の作成

データソース名 (DSN) を使用してデータストアに接続する場合は、DSN を定義するように odbc.ini ファイルを設定します。odbc.ini ファイルのプロパティを設定して、データストアの接続情報を指定する DSN を作成します。

odbc.ini ファイルの設定方法については、Amazon Redshift ODBC コネクタのインストールおよび設定ガイドの「Windows 以外のマシンでデータソース名を作成する」を参照してください。

macOS オペレーティングシステムでは次の形式を使用します。

```
[ODBC Data Sources]
driver_name=dsn_name

[dsn_name]
Driver=path/lib/amazonredshiftodbc.dylib

Host=cluster_endpoint
Port=port_number
Database=database_name
locale=locale
```

次の例は、macOS オペレーティングシステムでの odbc.ini の設定を示しています。

```
[ODBC Data Sources]
```

```
Amazon_Redshift_dylib=Amazon Redshift DSN for macOS

[Amazon Redshift DSN for macOS]
Driver=/opt/amazon/redshift/lib/amazonredshiftodbc.dylib
Host=examplecluster.abc123xyz789.us-west-2.redshift.amazonaws.com
Port=5932
Database=dev
locale=en-US
```

macOS オペレーティングシステムで DSN を使用しない接続を設定する

DSN を使用しない接続を介してデータストアに接続するには、odbcinst.ini ファイルでドライバーを定義します。次に、アプリケーションに DSN を使用しない接続文字列を指定します。

この場合の odbcinst.ini ファイルの設定方法については、「Amazon Redshift ODBC connector installation and configuration guide」の「Configuring a DSN-less Connection on a Non-Windows Machine」を参照してください。

macOS オペレーティングシステムでは次の形式を使用します。

```
[ODBC Drivers]
driver_name=Installed
...

[driver_name]
Description=driver_description
Driver=path/lib/amazonredshiftodbc.dylib

...
```

次の例は、macOS オペレーティングシステムでデフォルトのディレクトリにインストールされたドライバーの odbcinst.ini 設定を示しています。

```
[ODBC Drivers]
Amazon RedshiftODBC DSN=Installed

[Amazon RedshiftODBC DSN]
Description=Amazon Redshift ODBC Driver for macOS
Driver=/opt/amazon/redshift/lib/amazonredshiftodbc.dylib
```

環境変数の設定

正しい ODBC ドライバーマネージャーを使用して、正しいドライバーをロードします。これを行うには、ライブラリパス環境変数を設定します。詳細については、Amazon Redshift ODBC connector installation and configuration guide の「Specifying ODBC Driver Managers on Non-Windows Machines」を参照してください。

デフォルトでは、ODBC ドライバーマネージャーは、隠しバージョンの `odbc.ini` および `odbcinst.ini` 設定ファイル (`.odbc.ini` と `.odbcinst.ini`という名前) をホームディレクトリに配置します。また、ドライバーのインストールディレクトリの `amazon.redshiftodbc.ini` サブフォルダにある `/lib` ファイルを使用するように設定されています。これらの設定ファイルを他の場所に保存する場合は、ドライバーマネージャーがファイルを検索できるように環境変数を設定します。詳細については、Amazon Redshift ODBC Connector Installation and Configuration Guide の「Specifying the Locations of the Driver Configuration Files」を参照してください。

接続機能の設定

ODBC 設定には、次の接続機能を設定できます。

- 認証情報を提供し、Amazon Redshift データベースへの接続を認証するように ODBC ドライバーを設定します。
- SSL が有効な Amazon Redshift サーバーに接続している場合は、Secure Sockets Layer (SSL) で有効なソケットに接続するように ODBC ドライバーを設定します。
- プロキシサーバー経由で Amazon Redshift に接続するように ODBC ドライバーを設定します。
- クエリ処理モードを使用するように ODBC ドライバーを設定して、クエリが大量のメモリを消費しないようにします。
- IAM 認証プロセスをプロキシサーバー経由で渡すように ODBC ドライバーを設定します。
- 接続のタイムアウトを防ぐために TCP キープアライブを使用するように ODBC ドライバーを設定します。

これらの接続機能の詳細については、Amazon Redshift ODBC connector installation and configuration guide を参照してください。

ODBC ドライバーオプション

設定オプションを使用して、Amazon Redshift ODBC ドライバーの動作を制御できます。

Microsoft Windows では、通常、データソース名 (DSN) を設定するときにドライバーオプションを設定します。プログラムを使用して接続するとき、または `HKEY_LOCAL_MACHINE\SOFTWARE\ODBC`

\ODBC.INI*your_DSN* でレジストリキーを追加または変更することによって、接続文字列にドライバーオプションを設定することもできます。DSN の設定の詳細については、「Microsoft Windows での Amazon Redshift ODBC ドライバーの使用」を参照してください。

macOS では、odbc.ini および amazon.redshiftodbc.ini ファイルでドライバー設定オプションを設定します (「ODBC ドライバーマネージャーを使用してドライバーを設定する」を参照)。amazon.redshiftodbc.ini ファイルに設定されている設定オプションは、すべての接続に適用されます。これに対して、odbc.iniファイルで設定されている設定オプションは接続に固有です。odbc.ini で設定された設定オプションは、amazon.redshiftodbc.ini で設定された設定オプションよりも優先されます。

ODBC ドライバーの設定オプションを指定する方法については、Amazon Redshift ODBC connector installation and configuration guide を参照してください。

以前のバージョンの ODBC ドライバー

ツールに特定のバージョンのドライバーが必要な場合にのみ、Amazon Redshift ODBC ドライバーの以前のバージョンをダウンロードします。

以前のバージョンの ODBC ドライバー (Windows 用)

64 ビットドライバーは次のとおりです。

- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.6.3.1006/AmazonRedshiftODBC64-1.6.3.1006.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.6.1.1000/AmazonRedshiftODBC64-1.6.1.1000.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.5.20.1024/AmazonRedshiftODBC64-1.5.20.1024.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.5.16.1019/AmazonRedshiftODBC64-1.5.16.1019.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.5.9.1011/AmazonRedshiftODBC64-1.5.9.1011.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.5.7.1007/AmazonRedshiftODBC64-1.5.7.1007.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.65.1000/AmazonRedshiftODBC64-1.4.65.1000.msi

- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.62.1000/AmazonRedshiftODBC64-1.4.62.1000.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.59.1000/AmazonRedshiftODBC64-1.4.59.1000.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.56.1000/AmazonRedshiftODBC64-1.4.56.1000.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.53.1000/AmazonRedshiftODBC64-1.4.53.1000.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.52.1000/AmazonRedshiftODBC64-1.4.52.1000.msi

32 ビットドライバーは廃止され、以前のバージョンはサポートされていません。

以前のバージョンの ODBC ドライバー (Linux 用)

64 ビットドライバーのバージョンは次のとおりです。

- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.6.3.1006/AmazonRedshiftODBC-64-bit-1.6.3.1006-1.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.6.1.1000/AmazonRedshiftODBC-64-bit-1.6.1.1000-1.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.5.20.1024/AmazonRedshiftODBC-64-bit-1.5.20.1024-1.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.5.16.1019/AmazonRedshiftODBC-64-bit-1.5.16.1019-1.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.5.9.1011/AmazonRedshiftODBC-64-bit-1.5.9.1011-1.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.5.7.1007/AmazonRedshiftODBC-64-bit-1.5.7.1007-1.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.65.1000/AmazonRedshiftODBC-64-bit-1.4.65.1000-1.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.62.1000/AmazonRedshiftODBC-64-bit-1.4.62.1000-1.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.59.1000/AmazonRedshiftODBC-64-bit-1.4.59.1000-1.x86_64.rpm

- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.59.1000/AmazonRedshiftODBC-64-bit-1.4.59.1000-1.x86_64.deb
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.56.1000/AmazonRedshiftODBC-64-bit-1.4.56.1000-1.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.56.1000/AmazonRedshiftODBC-64-bit-1.4.56.1000-1.x86_64.deb
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.52.1000/AmazonRedshiftODBC-64-bit-1.4.52.1000-1.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.52.1000/AmazonRedshiftODBC-64-bit-1.4.52.1000-1.x86_64.deb

32 ビットドライバーは廃止され、以前のバージョンはサポートされていません。

以前のバージョンの ODBC ドライバー (macOS 用)

macOS 用の Amazon Redshift ODBC ドライバーのバージョンは次のとおりです。

- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.6.3.1006/AmazonRedshiftODBC-64-bit.1.6.3.1006.universal.pkg
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.6.1.1000/AmazonRedshiftODBC-64-bit.1.6.1.1000.universal.pkg
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.5.20.1024/AmazonRedshiftODBC-1.5.20.1024.arm64.dmg
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.5.20.1024/AmazonRedshiftODBC-1.5.20.1024.x86_64.dmg
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.5.16.1019/AmazonRedshiftODBC-1.5.16.1019.x86_64.dmg
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.5.9.1011/AmazonRedshiftODBC-1.5.9.1011.x86_64.dmg
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.5.7.1007/AmazonRedshiftODBC-1.5.7.1007.x86_64.dmg
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.65.1000/AmazonRedshiftODBC-1.4.65.1000.dmg
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.62.1000/AmazonRedshiftODBC-1.4.62.1000.dmg

- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.59.1000/AmazonRedshiftODBC-1.4.59.1000.dmg
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.56.1000/AmazonRedshiftODBC-1.4.56.1000.dmg
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.52.1000/AmazonRedshiftODBC-1.4.52.1000.dmg

メタデータ検出のベストプラクティス

Amazon Redshift には、メタデータを検出するための複数の方法があります。アプリケーションやツール (AI エージェントで構築されたものを含む) の場合は、ドライバーメタデータ API または SHOW コマンドを使用して、一貫性、信頼性、パフォーマンスの高いメタデータ検出を行うことをお勧めします。

トピック

- アプリケーションとツールに Amazon Redshift ドライバーメタデータ API を使用する
- SHOW コマンドを使用する
- システムテーブルのクエリ
- アプリケーション名接続プロパティを設定する

アプリケーションとツールに Amazon Redshift ドライバーメタデータ API を使用する

ビジネスインテリジェンスツールやクエリエディタなど、Amazon Redshift に接続するアプリケーションやツールの場合は、Amazon Redshift JDBC 2.x、ODBC 2.x、または Python ドライバーが提供するドライバーメタデータ API を使用して、データベース、スキーマ、テーブル、列、データ型などのデータウェアハウスオブジェクトに関するメタデータを検出することをお勧めします。あるいは、Amazon Redshift SHOW コマンドを使用することもできます。

ドライバーメタデータ API を使用すると、次の利点が得られます。

- 仕様に準拠する。JDBC および ODBC ドライバーは、標準のメタデータインターフェイス (JDBC では `DatabaseMetaData`、ODBC では `SQLTables` および `SQLColumns`) を実装しています。Python の DB-API (PEP 249) はメタデータ API 仕様を定義していないため、Amazon Redshift Python ドライバーは JDBC DatabaseMetaData 仕様に従い、`get_tables()`、`get_columns()`、`get_schemas()` などの同等のメソッドを提供します。これらの API は明確に定義された仕様に従っているため、統合コードは移植可能になりま

す。Amazon Redshift はその内部システムテーブルを進化させているため、アプリケーションを変更する必要はありません。

- パフォーマンスが最適化される。ドライバーメタデータ API は、メタデータを効率的に返すように最適化されています。AWS は今後もドライバーメタデータ API のパフォーマンスへの投資を継続します。
- 上位互換性が得られる。Amazon Redshift は、JDBC、ODBC、および Python コネクタの仕様に準拠しています。これらの標準 API に合わせてコーディングすると、アプリケーションは基盤となるシステムカタログ構造の変更から保護されます。

例: JDBC DatabaseMetaData.getTables() を使用したテーブルメタデータの取得

```
DatabaseMetaData dbmd = connection.getMetaData();

// getTables(catalog, schemaPattern, tableNamePattern, types)
//   catalog:          "test" – filters to the database named "test"
//   schemaPattern:    "test_pattern" – filters schemas matching this pattern (supports
 SQL wildcards % and _)
//   tableNamePattern: null – no filter, returns all table names
//   types:            {"TABLE", "EXTERNAL TABLE"} – only return regular tables and
 external tables
ResultSet rs = dbmd.getTables("test", "test_pattern", null, new String[] {"TABLE",
 "EXTERNAL TABLE"});
```

例: Python cursor.get_columns() を使用した列メタデータの取得

```
cursor: redshift_connector.Cursor = conn.cursor()

# get_columns(catalog, schema_pattern, table_name_pattern, column_name_pattern)
#   catalog:             'test' – filters to the database named "test"
#   schema_pattern:      'test_pattern' – filters schemas matching this pattern
 (supports SQL wildcards % and _)
#   table_name_pattern:  'testabc' – filters to the table named "testabc"
#   column_name_pattern: '%' – wildcard, returns all columns in the matching table
result: tuple = cursor.get_columns('test', 'test_pattern', 'testabc', '%')
```

例: ODBC SQLPrimaryKeys() を使用したプライマリキーメタデータの取得

```
// SQLPrimaryKeys(hstmt, catalog, catalog_len, schema, schema_len, table, table_len)
//   catalog: "test"         – filters to the database named "test"
```

```
//   schema:  "test_schema"  – filters to the schema named "test_schema"
//   table:   "test_table"   – retrieves primary key columns for this table
// Note: Unlike getTables/getColumns, SQLPrimaryKeys does NOT support wildcard
 patterns.
retcode = SQLPrimaryKeys(hstmt, (SQLCHAR *)"test", SQL_NTS, (SQLCHAR *)"test_schema",
 SQL_NTS, (SQLCHAR *)"test_table", SQL_NTS);

while (SQL_SUCCEEDED(retcode = SQLFetch(hstmt))) {
    for (i = 1; i <= columns; i++) {
        retcode = SQLGetData(hstmt, i, SQL_C_CHAR, buf, sizeof(buf), &indicator);
    }
}
```

例: ODBC SQLTables() を使用したデータベースとスキーマの一覧表示

ODBC API は、カタログまたはスキーマを一覧表示するための個別の関数を提供していません。代わりに、SQLTables( ) の特別な呼び出し規則を使用してこの情報を取得します。

すべてのデータベースを一覧表示するには (カタログ)

CatalogName を SQL_ALL_CATALOGS に設定して SQLTables( ) を呼び出します。SchemaName と TableName を空の文字列に設定します。結果セットは、TABLE_CAT 列のみの有効な値を返します。他のすべての列には NULL が含まれています。

```
// List all catalogs (databases) available on the data source.
retcode = SQLTables(hstmt,
    (SQLCHAR *)SQL_ALL_CATALOGS, SQL_NTS, // CatalogName = "%" (SQL_ALL_CATALOGS)
    (SQLCHAR *)"", 0,                     // SchemaName = "" (empty string)
    (SQLCHAR *)"", 0,                     // TableName  = "" (empty string)
    NULL, 0);                             // TableType  = NULL (not filtered)
```

すべてのスキーマを一覧表示するには

SchemaName を SQL_ALL_SCHEMAS に設定して SQLTables( ) を呼び出します。CatalogName と TableName を空の文字列に設定します。

```
// List all schemas available on the data source.
retcode = SQLTables(hstmt,
    (SQLCHAR *)"", 0,                     // CatalogName = "" (empty string)
    (SQLCHAR *)SQL_ALL_SCHEMAS, SQL_NTS,  // SchemaName = "%" (SQL_ALL_SCHEMAS)
    (SQLCHAR *)"", 0,                     // TableName  = "" (empty string)
```

```
    NULL, 0);                                   // TableType  = NULL (not filtered)
```

> **Note**
>
> ODBC 仕様では、スキーマの列挙に対して TABLE_SCHEM のみが有効として定義されています。また、Amazon Redshift はクロスデータベースメタデータ検出をサポートし、各スキーマのスコープは特定のデータベースであるため、TABLE_CAT も入力します。

SHOW コマンドを使用する

ドライバーメタデータ API がユースケースに対応していない場合は、Amazon Redshift の SHOW コマンドを使用してメタデータを取得できます。SHOW コマンドは、メタデータをすばやく取得できるように最適化されています。これらは、インタラクティブ SQL セッションでメタデータが必要な場合や、ドライバーメタデータ API を公開しないクライアントを介してアプリケーションが接続する場合に役立ちます。

一般的な検出操作では、次の SHOW コマンドがサポートされています。

- SHOW DATABASES
- SHOW SCHEMAS
- SHOW TABLES
- SHOW COLUMNS
- SHOW CONSTRAINTS
- SHOW GRANTS
- SHOW FUNCTIONS
- SHOW PROCEDURE
- SHOW PARAMETERS

例

```
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       |
         |                 |
```

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
```

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     |
```

詳細については、「Amazon Redshift データベース開発者ガイド」の「SHOW」を参照してください。

システムテーブルのクエリ

ドライバーメタデータ API も SHOW コマンドもニーズに合わない場合は、PostgreSQL カタログテーブル (pg_*)、情報スキーマ (information_schema)、SVV_TABLE_INFO などの Amazon Redshift システムビューに対して直接クエリを実行できます。

このアプローチは、Amazon Redshift Query Editor v2 や psql など、インタラクティブな SQL クエリから実行するアドホック検索、デバッグ、または管理タスクに役立ちます。ただし、システムカタログ構造は各 Amazon Redshift リリースで変更される可能性があります。本番稼働アプリケーション統合のメタデータインターフェイスとして、システムテーブルのクエリに直接依存することはお勧めしません。

システムテーブルに対してクエリを実行するときは、次のプラクティスに従います。

- データベース、スキーマ、およびテーブルを指定します。最善のパフォーマンスを得るには、クエリフィルターにデータベース名、スキーマ名、およびテーブル名を含めます。スコープを絞り込むと、Amazon Redshift がスキャンして返す必要があるメタデータの量が減ります。
- 等価フィルターまたは LIKE フィルターを使用します。WHERE 句で等価 (=) または LIKE 述語を使用して結果を絞り込みます。大規模なカタログビューに対して、フィルタリングされていないクエリを実行することは避けてください。オブジェクトが多いクラスターではクエリが低速になる可能性があります。
- フィルター述語で関数を使用しないでください。システムテーブルクエリをフィルタリングするときは、WHERE 句で LOWER( ) や文字列連結などの関数を使用しないでください。これらの操作を行うと、メタデータの効率的な検索が妨げられ、クエリのパフォーマンスが大幅に低下する可能性があります。

アンチパターンと推奨クエリの比較の例

```
-- Avoid:
SELECT * FROM svv_all_columns;

-- Recommended:
```

```
SELECT * FROM svv_all_columns WHERE database_name = 'dev' and schema_name = 'public'
 AND table_name = 'sales';
```

```
-- Avoid:
SELECT * FROM svv_all_columns WHERE LOWER(schema_name) = 'public';

-- Recommended:
SELECT * FROM svv_all_columns WHERE schema_name = 'public' AND table_name LIKE 'sales
%';
```

```
-- Avoid:
SELECT * FROM svv_all_columns WHERE database_name || '.' || schema_name || '.' ||
 table_name = 'dev.public.sales';

-- Recommended:
SELECT * FROM svv_all_columns WHERE database_name = 'dev' and schema_name = 'public'
 AND table_name = 'sales';
```

```
-- Avoid:
SELECT * FROM svv_all_columns WHERE CONCAT(database_name, '.', schema_name) =
 'dev.public';

-- Recommended:
SELECT * FROM svv_all_columns WHERE database_name = 'dev' and schema_name = 'public';
```

アプリケーション名接続プロパティを設定する

Amazon Redshift に接続するときは、オプションのアプリケーション名ドライバープロパティを設定します。このプロパティは、各接続を確立したアプリケーションまたはツールを識別し、Amazon Redshift 環境のモニタリング、監査、およびトラブルシューティングを容易にします。

指定したアプリケーション名が SYS_CONNECTION_LOG の application_name 列に表示されます。これを使用して、以下の操作を行うことができます。

- 接続ソースを追跡する。クラスターに接続するアプリケーション、サービス、またはツールを識別します。
- パフォーマンスのトラブルシューティングを行う。クエリを元のアプリケーションと関連付けて、パフォーマンスの問題を分離します。
- アクセスを監査する。データウェアハウス全体のアプリケーションレベルの使用パターンを理解します。

次の表は、各 Amazon Redshift ドライバーのアプリケーション名を設定する方法を示しています。

ドライバー	プロパティ	例
JDBC	`ApplicationName`	`jdbc:redshift:// endpoint:port/database ApplicationName=my-etl-pipeline`
ODBC	`ApplicationName`	DSN または接続文字列で `ApplicationName=my-bi-tool` を設定します。
Python	`application_name`	`redshift_connector.connect(..., application_name='my-data-catalog')`

各ドライバーのアプリケーション名プロパティの詳細については、以下を参照してください。

- JDBC: ApplicationName
- ODBC: ApplicationName
- Python: application_name

> **Note**
>
> *team*-*service*-*environment* (例: quicksight-analytics-dashboard-prod) などのアプリケーション名には、わかりやすく一貫した命名規則を使用します。このアプローチにより、SYS_CONNECTION_LOG データのフィルタリングと分析が容易になります。

接続のセキュリティオプションを設定する

Amazon Redshift では Secure Sockets Layer (SSL) 接続がサポートされているため、データとサーバー証明書を暗号化でき、クライアントが接続するサーバー証明書を検証できます。

SSL

SSL 接続をサポートするために、Amazon Redshift では、AWS Certificate Manager(ACM) によって発行された SSL 証明書を作成して各クラスターにインストールします。ACM 証明書は、ほとんどのオペレーションシステム、ウェブブラウザ、およびクライアントから公的に信頼されています。SQL クライアントまたはアプリケーションが、`sslmode`接続オプションを`require`、`verify-ca`、または`verify-full`に設定した SSL を使用して Amazon Redshift に接続する場合は、証明書バンドルをダウンロードする必要があるかもしれません。クライアントが証明書を必要とする場合、Amazon Redshift は以下のようにバンドル証明書を提供します。

> **Note**
>
> 可能な限り、SSL 接続に `verify-full` を使用することをお勧めします。

- https://s3.amazonaws.com/redshift-downloads/amazon-trust-ca-bundle.crtからバンドルをダウンロードします。
  - 予想される MD5 チェックサム番号は 418dea9b6d5d5de7a8f1ac42e164cdcf です。
  - sha256 のチェックサム番号は 36dba8e4b8041cd14b9d60158893963301bcb92e1c456847784de2acb5bd550 です。

  `https://s3.amazonaws.com/redshift-downloads/redshift-ca-bundle.crt`にあった以前の証明書バンドルは使用しないでください。
- 中国の AWS リージョンでは、https://s3.cn-north-1.amazonaws.com.cn/redshift-downloads-cn/amazon-trust-ca-bundle.crt からバンドルをダウンロードします。
  - 予想される MD5 チェックサム番号は 418dea9b6d5d5de7a8f1ac42e164cdcf です。
  - sha256 のチェックサム番号は 36dba8e4b8041cd14b9d60158893963301bcb92e1c456847784de2acb5bd550 です。

  `https://s3.cn-north-1.amazonaws.com.cn/redshift-downloads-cn/redshift-ca-bundle.crt`および`https://s3.cn-north-1.amazonaws.com.cn/redshift-downloads-cn/redshift-ssl-ca-cert.pem`にあった以前の証明書バンドルは使用しないでください。

> **Important**
>
> Amazon Redshift は、SSL 証明書の管理方法を変更しました。引き続き SSL でクラスターに接続するには、現在の信頼ルート CA 証明書を更新する必要があります。詳細については、「SSL 接続用 ACM 証明書への移行」を参照してください。

デフォルトでは、クラスターデータベースは SSL の使用の有無にかかわらず接続を受け入れます。クラスターが SSL 接続を要求するように設定するには、クラスターに関連付けられているパラメータグループで `require_SSL` パラメータを `true` に設定します。

Amazon Redshift は、連邦情報処理標準 (FIPS) 140-2 に準拠する SSL モードをサポートしています。FIPS 準拠 SSL モードはデフォルトで無効化されています。

> **Important**
>
> システムが FIPS に準拠する必要がある場合のみ、FIPS 準拠の SSL モードを有効にします。

FIPS 準拠の SSL モードを有効にするには、Amazon Redshift クラスターまたは Redshift Serverless ワークグループに関連付けたパラメータグループで、`use_fips_ssl` パラメータと `require_SSL` パラメータの両方を `true` に設定します。クラスターのパラメータグループの変更については、「Amazon Redshift パラメータグループを作成します。」を参照してください。ワークグループのパラメータグループの変更については、「Amazon Redshift Serverless への FIPS 準拠の SSL 接続の設定」を参照してください。

Amazon Redshift では、Elliptic Curve Diffie—Hellman Ephemeral (ECDHE) 鍵契約プロトコルがサポートされます。ECDHE を使用すると、クライアントとサーバーはそれぞれ、安全でないチャネルを介して共有シークレットを確立するために使用される楕円曲線パブリック/プライベートキーペアを持ちます。ECDHE を有効にするために、Amazon Redshift で設定を行う必要はありません。ECDHE を使用してクライアント/サーバー間の通信を暗号化する SQL クライアントツールから接続する場合、Amazon Redshift は提供された暗号リストを使用して適切な接続を行います。詳細については、ウィキペディアの Elliptic curve diffie—hellman と OpenSSL ウェブサイトの Ciphers を参照してください。

ODBC の SSL および信頼 CA 証明書

最新の Amazon Redshift ODBC ドライバー (バージョン 1.3.7.1000 以降) を使用して接続している場合は、このセクションをスキップします。最新のドライバーをダウンロードする方法は、「Amazon Redshift での ODBC ドライバーのバージョン 2.x による接続の設定」を参照してください。

引き続き SSL でクラスターに接続するには、現在の信頼ルート CA 証明書を更新する必要があります。詳細については、「SSL」を参照してください。

ダウンロードした証明書が、予想される MD5 チェックサム番号と一致していることを確認できます。これを行うには、Linux オペレーティングシステムでは Md5sum プログラムを使用するか、Windows および macOS オペレーティングシステムでは別のツールを使用できます。

ODBC DSN には、クライアント接続の暗号化処理方法とサーバー証明書の検証を決定する `sslmode` 設定が含まれます。Amazon Redshift では、クライアント接続から次の `sslmode` 値をサポートします。

- `disable`

  SSL は無効となり、接続は暗号化されません。

- `allow`

  SSL は、サーバーで必要な場合に使用されます。

- `prefer`

  SSL は、サーバーでサポートされている場合に使用されます。Amazon Redshift は SSL をサポートしているため、`sslmode` を `prefer` に設定すると SSL が使用されます。

- `require`

  SSL が必要です。

- `verify-ca`

  SSL を使用する必要があり、サーバー証明書を検証する必要があります。

- `verify-full`

  SSL を使用する必要があります。サーバー証明書を検証し、サーバーのホスト名が証明書上のホスト名属性と一致する必要があります。

クライアントとサーバー間の接続で SSL が使用され、サーバー証明書が検証されているかどうかを確認できます。これを実行するには、クライアント上の ODBC DSN の sslmode 設定と、サーバー上の Amazon Redshift クラスターの require_SSL 設定を確認します。次の表は、クライアントとサーバーのさまざまな設定の組み合わせの暗号化の結果を示しています。

sslmode (クライアント)	require_SSL (サーバー)	結果
disable	false	接続は暗号化されません。
disable	true	サーバーにより SSL が要求されていますが、クライアントの接続では SSL が無効になっているため、接続を行うことはできません。
allow	true	接続は暗号化されます。
allow	false	接続は暗号化されません。
prefer または require	true	接続は暗号化されます。
prefer または require	false	接続は暗号化されます。
verify-ca	true	接続は暗号化され、サーバー証明書が検証されます。
verify-ca	false	接続は暗号化され、サーバー証明書が検証されます。
verify-full	true	接続は暗号化され、サーバー証明書とホスト名が検証されます。
verify-full	false	接続は暗号化され、サーバー証明書とホスト名が検証されます。

Microsoft Windows で ODBC によりサーバー証明書を使用して接続する

SSL とサーバー証明書を使用してクラスターに接続する場合は、まず証明書をクライアントコンピュータまたは Amazon EC2 インスタンスにダウンロードします。次に、ODBC DSN を設定します。

1. Amazon Redshift の認定権限バンドルを、ドライバーのインストールディレクトリ内の`lib`フォルダでクライアントコンピュータにダウンロードし、ファイルを`root.crt`として保存します。ダウンロード情報については、「SSL」を参照してください。
2. [ODBC Data Source Administrator (ODBC データソース管理者)] を開き、ODBC 接続のシステム DSN エントリを追加または編集します。DNS エイリアスを使用する場合を除き、[SSL Mode (SSL モード)] で [`verify-full`] を選択します。DNS エイリアスを使用する場合は、[`verify-ca`] を選択します。次に、[Save] を選択します。

   ODBC DSN の設定方法の詳細については、「Amazon Redshift での ODBC ドライバーのバージョン 2.x による接続の設定」を参照してください。

Java の SSL およびサーバー証明書

SSL は、クライアントとクラスター間を移動するデータを暗号化することによりセキュリティを強化します。サーバー証明書の使用により、クラスターが Amazon Redshift クラスターであることが確認され、セキュリティがさらに強化されます。これを行うには、プロビジョニングしたすべてのクラスターに自動インストールされたサーバー証明書をチェックします。JDBC でサーバー証明書を使用する方法の詳細については、PostgreSQL ドキュメントの「Configuring the Client」を参照してください。

Java で信頼 CA 証明書を使用して接続する

> **Important**
>
> Amazon Redshift は、SSL 証明書の管理方法を変更しました。引き続き SSL でクラスターに接続するには、現在の信頼ルート CA 証明書を更新する必要があります。詳細については、「SSL」を参照してください。

信頼 CA 証明書を使用して接続するには

Amazon Redshift 認証機関バンドルの CA 証明書を Java トラストストアまたはプライベートトラストストアにインポートするには、`redshift-keytool.jar` ファイルを使用します。

1. Java コマンドラインの -Djavax.net.ssl.trustStore オプションを使用している場合は、可能であればコマンドラインから解除します。
2. redshift-keytool.jar をダウンロードします。
3. 次のいずれかを行ってください。

   - Amazon Redshift の認定権限バンドルを Java TrustStore にインポートするには、次のコマンドを実行します。

     ```
     java -jar redshift-keytool.jar -s
     ```

   - Amazon Redshift の認定権限バンドルをプライベート TrustStore にインポートするには、次のコマンドを実行します。

     ```
     java -jar redshift-keytool.jar -k <your_private_trust_store> -
     p <keystore_password>
     ```

SSL 接続用 ACM 証明書への移行

Amazon Redshift では、現在クラスターの SSL 証明書をAWS Certificate Manager (ACM)」発行の証明書に置き換えています。ACM は、現在のほとんどのシステムで信頼されている信頼のある公開認証機関 (CA) です。引き続き SSL でクラスターに接続するには、現在の信頼ルート CA 証明書を更新する必要があります。

この変更は、以下のすべてに該当する場合のみ適用されます。

- sslMode 接続オプションを require、verify-ca、または verify-full 設定オプションに指定して、SQL クライアントまたはアプリケーションから Amazon Redshift クラスターに SSL 接続されている。
- Amazon Redshift ODBC または JDBC ドライバーを使用していない、または ODBC バージョン 1.3.7.1000 または JDBC バージョン 1.2.8.1005 より前の Amazon Redshift ドライバーを使用している。

この変更が商用 Amazon Redshift リージョンに影響する場合は、現在の信頼ルート CA 証明書を 2017 年 10 月 23 日までに更新する必要があります。Amazon Redshift は、お客様のクラスターを 2017 年 10 月 23 日までに順次 ACM 証明書に切り替えていきます。この変更によるお客様のクラスターのパフォーマンスまたはアベイラビリティーへの影響はほとんどありません。

AWS GovCloud (US) (米国)リージョンのお客様が、この変更により影響を受ける場合には、サービスの中断を避けるために、現在の信頼ルート CA 証明書を 2020 年 4 月 1 日までに更新する必要があります。この日付以降、SSL 暗号化接続を使用して Amazon Redshift クラスターに接続するクライアントには、追加の信頼できる認定権限 (CA) が必要です。クライアントは、信頼できる認定権限を使用して、クラスターに接続するときに Amazon Redshift クラスターの ID を確認します。SQL クライアントとアプリケーションを更新して、新しい信頼された CA を含む更新された証明書バンドルを使用するには、アクションが必要です。

> **Important**
>
> 2021 年 1 月 5 日、中国リージョンにおいて、Amazon Redshift の クラスター用の SSL 証明書が、AWS Certificate Manager(ACM) 発行の証明書に置き換えられました。この変更が中国 (北京) リージョンまたは中国 (寧夏) リージョンに影響する場合は、サービスの中断を避けるために、現在の信頼ルート CA 証明書を 2021 年 1 月 5 日までに更新する必要があります。この日付以降、SSL 暗号化接続を使用して Amazon Redshift クラスターに接続するクライアントには、追加の信頼できる認定権限 (CA) が必要です。クライアントは、信頼できる認定権限を使用して、クラスターに接続するときに Amazon Redshift クラスターの ID を確認します。SQL クライアントとアプリケーションを更新して、新しい信頼された CA を含む更新された証明書バンドルを使用するには、アクションが必要です。

- 最新の Amazon Redshift ODBC ドライバーまたは JDBC ドライバーを使用する
- 旧 Amazon Redshift ODBC ドライバーまたは JDBC ドライバーを使用する
- その他の SSL 接続タイプを使用する

最新の Amazon Redshift ODBC ドライバーまたは JDBC ドライバーを使用する

最新の Amazon Redshift の ODBC ドライバーまたは JDBC ドライバーを使用することが推奨されています。ODBC バージョン 1.3.7.1000 および JDBC バージョン 1.2.8.1005 以前の Amazon Redshift ドライバーは、自動的に Amazon Redshift の自己署名証明書から ACM 証明書に移行されます。最新のドライバーをダウンロードする方法は、「Amazon Redshift での JDBC ドライバーバージョン 2.x による接続の構成」を参照してください。

最新の Amazon Redshift JDBC ドライバーを使用する場合は、JVM オプションの `-Djavax.net.ssl.trustStore` を使用しないことをお勧めします。`-Djavax.net.ssl.trustStore`を使用する必要がある場合は、Redshift の認定権限バンドルを指定のトラストストアにインポートします。ダウンロード情報については、「SSL」を参照してくださ

い。詳細については、「Amazon Redshift の認定権限バンドルを TrustStore にインポートする」を参照してください。

旧 Amazon Redshift ODBC ドライバーまたは JDBC ドライバーを使用する

- ODBC DSN が `SSLCertPath` を使用して設定されている場合は、指定されたパスの証明書ファイルに上書きします。
- `SSLCertPath` が設定されていない場合は、ドライバーの DLL の位置にある証明書ファイル (`root.crt`) に上書きします。

バージョン 1.2.8.1005 より前の Amazon Redshift JDBC ドライバーを使用する必要がある場合は、以下のいずれかを行います。

- JDBC 接続文字列で `sslCert` オプションを使用している場合は、`sslCert`オプションを解除します。続いて、Redshift の認証機関バンドルを Java トラストストアにインポートします。ダウンロード情報については、「SSL」を参照してください。詳細については、「Amazon Redshift の認定権限バンドルを TrustStore にインポートする」を参照してください。
- Java コマンドラインの `-Djavax.net.ssl.trustStore` オプションを使用している場合は、可能であればコマンドラインから解除します。続いて、Redshift の認証機関バンドルを Java トラストストアにインポートします。ダウンロード情報については、「SSL」を参照してください。詳細については、「Amazon Redshift の認定権限バンドルを TrustStore にインポートする」を参照してください。

Amazon Redshift の認定権限バンドルを TrustStore にインポートする

Amazon Redshift の認定権限バンドルの CA 証明書を Java TrustStore またはプライベート信頼ストアにインポートするには、`redshift-keytool.jar` を使用します。

Amazon Redshift の認定権限バンドルを TrustStore にインポートするには

1. redshift-keytool.jar をダウンロードします。
2. 次のいずれかを行ってください。

   - Amazon Redshift の認定権限バンドルを Java TrustStore にインポートするには、次のコマンドを実行します。

     ```
     java -jar redshift-keytool.jar -s
     ```

- Amazon Redshift の認定権限バンドルをプライベート TrustStore にインポートするには、次のコマンドを実行します。

```
java -jar redshift-keytool.jar -k <your_private_trust_store> -
p <keystore_password>
```

その他の SSL 接続タイプを使用する

次のいずれかを使用して接続している場合は、このセクションの手順に従います。

- オープンソースの ODBC ドライバー
- オープンソースの JDBC ドライバー
- Amazon Redshift RSQL コマンドラインインターフェイス
- libpq に基づく言語バインディング (例: psycopg2 (Python)、ruby-pg (Ruby))

その他の SSL 接続タイプで ACM 証明書を使用するには

1. Amazon Redshift の認定権限バンドルをダウンロードします。ダウンロード情報については、「SSL」を参照してください。
2. バンドルの証明書を `root.crt` ファイルに追加します。
   - Linux および macOS オペレーティングシステムでは、ファイルは `~/.postgresql/root.crt` です。
   - Microsoft Windows では、ファイルは `%APPDATA%\postgresql\root.crt` です。

クライアントツールおよびコードからの接続

Amazon Redshift には、クラスターやワークグループに接続する Amazon Redshift クエリエディタ v2 が用意されています。詳細については、「クエリエディタ v2 を使用してデータベースのクエリを実行する」を参照してください。

このセクションでは、接続用サードパーティー製ツールのいくつかのオプションについて説明します。さらに、クラスターにプログラムにより接続する方法について説明します。

トピック

- Amazon Redshift RSQL を使用した接続

- Amazon Redshift RSQL を使用してクラスターに接続する
- Amazon Redshift RSQL メタコマンド
- Amazon Redshift RSQL 変数
- Amazon Redshift RSQL エラーコード
- Amazon Redshift RSQL 環境変数

Amazon Redshift RSQL を使用した接続

Amazon Redshift RSQL は、Amazon Redshift クラスターおよびデータベースと対話するためのコマンドラインクライアントです。Amazon Redshift クラスターに接続したり、データベースオブジェクトを記述したり、データをクエリしたり、クエリ結果をさまざまな出力形式で表示したりすることができます。

Amazon Redshift RSQL は、PostgreSQL psql コマンドラインツールの機能に、Amazon Redshift に特有の機能を追加したものをサポートします。これには以下が含まれます。

- AD FS、PingIdentity、Okta、Azure ADm、またはその他の SAML/JWT ベースの ID プロバイダーを使用した、シングルサインオン認証を使用できます。多要素認証 (MFA) には、ブラウザベースの SAML アイデンティティプロバイダーを使用することもできます。
- テーブルソートキー、テーブルソートキー、レイトバインディングビュー (LBV)、マテリアライズドビューなど、Amazon Redshift オブジェクトのプロパティまたは属性を記述できます。また、AWS Glueカタログや Apache Hive Metastore の外部テーブル、Amazon RDS for PostgreSQL、Amazon Aurora PostgreSQL-Compatible Edition、RDS for MySQL (プレビュー)、Amazon Aurora MySQL-Compatible Edition(プレビュー) の外部データベース、Amazon Redshift のデータ共有を利用して共有されたテーブルのプロパティや属性を記述することができます。
- また、IF(\ELSEIF、\ELSE,\ENDIF)、\GOTOおよび\LABELのような拡張コントロールフローコマンドを使用することもできます。

入力パラメータとして渡されたスクリプトを実行する Amazon Redshift RSQL バッチモードでは、SQL と複雑なビジネスロジックの両方を含むスクリプトを実行できます。既存の自己管理型のオンプレミスデータウェアハウスがある場合は、Amazon Redshift RSQL を使用して、Teradata BTEQ スクリプトなどの既存の抽出、変換、ロード (ETL) および自動化スクリプトを置き換えることができます。RSQL を使用すると、手続き型言語でスクリプトを手動で再実装する必要がなくなります。

Amazon Redshift RSQL は Linux、Windows、macOS オペレーションシステムで利用できます。

Amazon Redshift RSQL の問題を報告するには、<redshift-rsql-support@amazon.com>に書き込んでください。

トピック

- Amazon Redshift RSQL の開始方法
- Amazon Redshift RSQL の変更ログ

Amazon Redshift RSQL の開始方法

Linux、macOS、または Microsoft Windows オペレーションシステムを搭載したコンピュータに Amazon Redshift RSQL をインストールします。

RSQL をダウンロードする

- Linux 64 ビット RPM: RSQL バージョン 1.1.2
  - Linux アーティファクト署名キー: キー
  - Linux アーティファクト署名付き Hash: Hash
- Mac OS 64 ビット PKG: RSQL バージョン 1.1.2
- Windows 64 ビット MSI: RSQL バージョン 1.1.2

Amazon Redshift RSQL の変更ログ で変更ログを確認し、以前のバージョンをダウンロードしてください。

Linux 用 RSQL をインストールする

Linux 用 RSQL をインストールするには、以下の手順に従います。

1. 次のコマンドでドライバーマネージャーをインストールします。

```
sudo yum install unixODBC
```

2. ODBC ドライバーをインストールします。Amazon Redshift ODBC ドライバーをダウンロードしてインストールする。
3. ini ファイルをホームディレクトリにコピーします。

```
cp /opt/amazon/redshiftodbcx64/odbc.ini ~/.odbc.ini
```

4. ファイルの場所を指すように環境変数を設定します。

```
export ODBCINI=~/.odbc.ini
export ODBCSYSINI=/opt/amazon/redshiftodbcx64/
export AMAZONREDSHIFTODBCINI=/opt/amazon/redshiftodbcx64/amazon.redshiftodbc.ini
```

5. これで、次のコマンドを実行して RSQL をインストールできます。

```
sudo rpm -i AmazonRedshiftRsql-<version>.rhel.x86_64.rpm
```

Mac 用 RSQL をインストールする

Mac OSX 用 RSQL をインストールするには、以下の手順に従います。

1. 次のコマンドでドライバーマネージャーをインストールします。

```
brew install unixodbc --build-from-source
```

2. ODBC ドライバーをインストールします。Amazon Redshift ODBC ドライバーをダウンロードしてインストールする。
3. ini ファイルをホームディレクトリにコピーします。

```
cp /opt/amazon/redshift/Setup/odbc.ini ~/.odbc.ini
```

4. ファイルの場所を指すように環境変数を設定します。

```
export ODBCINI=~/.odbc.ini
export ODBCSYSINI=/opt/amazon/redshift/Setup
export AMAZONREDSHIFTODBCINI=/opt/amazon/redshift/lib/amazon.redshiftodbc.ini
```

5. /usr/local/libにない場合は、libodbc.dylib の場所にDYLD_LIBRARY_PATHを設定します。

```
export DYLD_LIBRARY_PATH=$DYLD_LIBRARY_PATH:/usr/local/lib
```

6. pkg ファイルをダブルクリックしてインストーラを実行します。
7. インストーラの手順に従ってインストールを完了します。ライセンス契約の条項に同意する。

Windows 用 RSQL をインストールする

Windows 用 RSQL をインストールするには、以下の手順に従います。

1. ODBC ドライバーをインストールします。Amazon Redshift ODBC ドライバーをダウンロードしてインストールする。
2. RSQL ダウンロードファイルをダブルクリックしてインストーラを実行し、プロンプトに従ってインストールを完了します。

Amazon Redshift RSQL の変更ログ

1.1.2 (2025 年 12 月 11 日)

バグ修正

- \goto コマンドと \label コマンドで障害が発生するバグを修正しました。
- 変数が引用符で囲まれたときに RSQL が変数値を印刷できないバグを修正しました。
- UseDeclareFetch が有効になっていて、クエリ結果が ODBC DSN Fetch パラメータサイズを超えたときに RSQL がクラッシュするバグを修正しました。
- ページャーがオンになっていても、複数の結果ページが一度に返される問題を修正しました。
- トランザクションブロック内でクエリが失敗したときに RSQL がクラッシュするバグを修正しました。

1.1.1 (2025 年 11 月 20 日)

バグ修正

- -c フラグの使用時に RSQL がクエリを誤って解析する問題を修正しました。この修正はすべてのプラットフォームに適用されます。
- ユーザーが RSQL で \s コマンドを使用できなくなる Mac のバグを修正しました。

1.1.0 (2025 年 11 月 11 日)

バグ修正

- RSQL で予期しないクラッシュを引き起こしていたメモリリークの問題を解決しました。
- RSQL から OpenSSL 依存関係を削除しました。

- 同じ環境内の libpq/psql インストールとのリンクの競合を修正しました。
- Amazon Linux 2023、Windows、macOS のプラットフォーム互換性が向上しました。
- 現在の表示サイズを超えると出力が切り捨てられる問題を修正しました。

1.0.8 (2023-06-19)

バグ修正

- SHOW コマンドで出力が切り捨てられる問題を修正しました。
- 外部 Kinesis ストリームと Kafka トピックを記述するためのサポートを \de に追加しました。

1.0.7 (2023 年 3 月 22 日)

バグ修正

- RSQL がマテリアライズドビューを記述できない問題を修正しました。
- Amazon Redshift Serverless を使用しているときに stl_connection_log でアクセス権限拒否エラーが発生する問題を修正しました。
- RSQL が \GOTO ラベルを誤って処理する場合がある問題を修正しました。
- SSL メッセージがクワイエットモードで印刷される問題を修正しました。
- ストアドプロシージャを記述するときにランダムな文字が表示される問題を修正しました。
- 重複する ERROR/INFO メッセージが印刷される問題を修正しました。

新

- RSQL は、ODBC ドライバーから SSL 情報を直接取得するようになりました。

1.0.6 (2023 年 2 月 21 日)

バグ修正

- Redshift パッチ 1.0.46086 (P173) で \d がエラー (整数の入力構文が無効: "xid") をスローする問題を修正しました。

新

- サポートされているアーキテクチャを反映するようにインストールファイルの名前を変更しました。

1.0.5 (2022-06-27)

バグ修正

- SQL エラーメッセージを標準エラー (stderr) に送信します。
- ON_ERROR_STOP を使用する際の終了コードの問題を修正しました。スクリプトはエラーが発生した後に終了し、正しい終了コードを返すようになりました。
- Maxerror は大文字と小文字は区別しなくなりました。

新

- ODBC 2.x ドライバーのサポートを追加しました。

1.0.4 (2022 年 3 月 19 日)

- 環境変数 RSPASSWORD のサポートを追加しました。Amazon Redshift に接続するためのパスワードを設定しました。例えば、`export RSPASSWORD=TestPassw0rd`。

1.0.3 (2021 年 12 月 8 日)

バグ修正

- Windows OS でデータベースの切り替えに `\c` または `\logon` を使用しているときのダイアログポップアップを修正しました。
- SSL 情報のチェック時におけるクラッシュを修正しました。

Amazon Redshift RSQL の以前のバージョン

お使いのオペレーティングシステムに基づいて、必要な Amazon Redshift RSQL のバージョンをダウンロードするためのリンクのいずれかを選択してください。

Linux 64 ビット RPM

- RSQL バージョン 1.1.1
- RSQL バージョン 1.1.0
- RSQL バージョン 1.0.8
- RSQL バージョン 1.0.7
- RSQL バージョン 1.0.6
- RSQL バージョン 1.0.5
- RSQL バージョン 1.0.4
- RSQL バージョン 1.0.3
- RSQL バージョン 1.0.1

Mac OS 64 ビット DMG/PKG

- RSQL バージョン 1.1.1
- RSQL バージョン 1.1.0
- RSQL バージョン 1.0.8
- RSQL バージョン 1.0.7
- RSQL バージョン 1.0.6
- RSQL バージョン 1.0.5
- RSQL バージョン 1.0.4
- RSQL バージョン 1.0.3
- RSQL バージョン 1.0.1

Windows 64 ビット MSI

- RSQL バージョン 1.1.1
- RSQL バージョン 1.1.0
- RSQL バージョン 1.0.8
- RSQL バージョン 1.0.7
- RSQL バージョン 1.0.6
- RSQL バージョン 1.0.5
- RSQL バージョン 1.0.4
- RSQL バージョン 1.0.3

- RSQL バージョン 1.0.1

Amazon Redshift RSQL を使用してクラスターに接続する

Amazon Redshift では、クラスターに接続し、RSQL を使用してクラスターを操作できます。これは、データをクエリし、データベースオブジェクトを作成し、Amazon Redshift クラスターを管理するための安全な方法を提供するコマンドラインツールです。以下のセクションでは、データソース名 (DSN) の有無にかかわらず RSQL を使用してクラスターへの接続を確立する手順について説明します。

DSN なしで接続する

1. Amazon Redshift コンソールで、接続するクラスターを選択し、エンドポイント、データベース、ポートを書き留めます。
2. コマンドプロンプトで、コマンドラインパラメータを使用して、接続情報を指定します。

   ```
   rsql -h <endpoint> -U <username> -d <databasename> -p <port>
   ```

   ここでは、次の条件が適用されます。

   - *<endpoint>* は前のステップで書き留めた [Endpoint (エンドポイント)] です。
   - *<#######>* は、クラスターに接続する権限を持つユーザーの名前です。
   - *<databasename>* は前のステップで書き留めた [Database Name (データベース名)] です。
   - *<###>* は、前のステップで記録したポートです。*<###>* は、オプションのパラメータです。

   以下に例を示します。

   ```
   rsql -h testcluster.example.amazonaws.com -U user1 -d dev -p 5439
   ```
3. パスワードプロンプトで、*<#######>* ユーザーのパスワードを入力します。

   接続に成功した場合のレスポンスは次のようになります。

   ```
   % rsql -h testcluster.example.com -d dev -U user1 -p 5349
   Password for user user1:
   DSN-less Connected
   DBMS Name: Amazon Redshift
   Driver Name: Amazon Redshift ODBC Driver
   ```

```
Driver Version: 1.4.27.1000
Rsql Version: 1.0.1
Redshift Version: 1.0.29306
Type "help" for help.

(testcluster) user1@dev=#
```

接続するコマンドは、Linux、Mac OS、および Windows で同じパラメータを持ちます。

DSN を使用して接続する

RSQL を Amazon Redshift に接続するには、DSN を使用して接続プロパティの組織を簡素化できます。このトピックでは、ODBC ドライバーのインストール手順と DSN プロパティの説明が含まれています。

パスワードで DSN 接続を使用する

次に、パスワードを使用した DSN 接続の設定例を示します。Mac OSX のデフォルト<path to driver>は/opt/amazon/redshift/lib/libamazonredshiftodbc.dylib、Linux のデフォルトは/opt/amazon/redshiftodbc/lib/64/libamazonredshiftodbc64.soです。

```
[testuser]
Driver=/opt/amazon/redshiftodbc/lib/64/libamazonredshiftodbc64.so
SSLMode=verify-ca
Min_TLS=1.2
boolsaschar=0
Host=<server endpoint>
Port=<database port>
Database=<dbname>
UID=<username>
PWD=<password>
sslmode=prefer
```

接続に成功した場合の出力結果は次のようになります。

```
% rsql -D testuser
DSN Connected
DBMS Name: Amazon Redshift
Driver Name: Amazon Redshift ODBC Driver
Driver Version: 1.4.27.1000
Rsql Version: 1.0.1
Redshift Version: 1.0.29306
```

```
Type "help" for help.

(testcluster) user1@dev=#
```

シングルサインオン DSN の使用

シングルサインオン認証用に DSN を設定できます。次に、シングルサインオン SSO を使用する DSN 接続設定の例を示します。

```
[testokta]
Driver=<path to driver>
SSLMode=verify-ca
Min_TLS=1.2
boolsaschar=0
Host=<server endpoint>
clusterid=<cluster id>
region=<region name>
Database=<dbname>
locale=en-US
iam=1
plugin_name=<plugin name>
uid=<okta username>
pwd=<okta password>
idp_host=<idp endpoint>
app_id=<app id>
app_name=<app name>
preferred_role=<role arn>
```

成功した接続からの出力例。

```
% rsql -D testokta
DSN Connected
DBMS Name: Amazon Redshift
Driver Name: Amazon Redshift ODBC Driver
Driver Version: 1.4.27.1000
Rsql Version: 1.0.1
Redshift Version: 1.0.29306
Type "help" for help.

(testcluster) user1@dev=#
```

次の例は、Azure シングルサインオンを使用する DSN 接続設定の例を示しています。

```
[testazure]
Driver=<path to driver>
SSLMode=verify-ca
Min_TLS=1.2
boolsaschar=0
Host=<server endpoint>
Port=<cluster port>
clusterid=<cluster id>
region=<region name>
Database=<dbname>
locale=en-us
iam=1
plugin_name=<plugin name>
uid=<azure username>
pwd=<azure password>
idp_tenant=<Azure idp tenant uuid>
client_id=<Azure idp client uuid>
client_secret=<Azure idp client secret>
```

IAM プロファイルで DSN 接続を使用する

設定した IAM プロファイルを使用して Amazon Redshift に接続できます。IAM プロファイルは、GetClusterCredentialsを呼び出すための権限が必要です。次の例は、使用する DSN プロパティを示しています。ClusterIDおよびRegionのパラメータは、Hostがexamplecluster.abc123xyz789.us-west-2.redshift.amazonaws.comのように Amazon が提供するエンドポイントではない場合にのみ必要になります。

```
[testiam]
Driver=Default
Host=testcluster.example.com
Database=dev
DbUser=testuser
ClusterID=rsqltestcluster
Region=us-east-1
IAM=1
Profile=default
```

Profileキーの値は、AWSCLI 認証情報から選択した名前付きプロファイルです。この例は、defaultという名前のプロファイルの認証情報を示しています。

```
$ cat .aws/credentials
```

```
[default]
aws_access_key_id = ASIAIOSFODNN7EXAMPLE
aws_secret_access_key = wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
```

接続レスポンスは次のとおりです。

```
$ rsql -D testiam
DSN Connected
DBMS Name: Amazon Redshift
Driver Name: Amazon Redshift ODBC Driver
Driver Version: 1.4.27.1000
Rsql Version: 1.0.1
Redshift Version: 1.0.29306
Type "help" for help.

(testcluster) testuser@dev=>
```

インスタンスプロファイルでの DSN 接続の使用

Amazon EC2 インスタンスプロファイルを使用して Amazon Redshift に接続できます。インスタンスプロファイルには、GetClusterCredentialsを呼び出すための権限が必要です。使用する DSN プロパティについては、以下の例を参照してください。ClusterIDおよびRegionのパラメータは、Hostがexamplecluster.abc123xyz789.us-west-2.redshift.amazonaws.comのように Amazon が提供するエンドポイントではない場合にのみ必要になります。

```
[testinstanceprofile]
Driver=Default
Host=testcluster.example.com
Database=dev
DbUser=testuser
ClusterID=rsqltestcluster
Region=us-east-1
IAM=1
Instanceprofile=1
```

接続レスポンスは次のとおりです。

```
$ rsql -D testinstanceprofile
DSN Connected
DBMS Name: Amazon Redshift
Driver Name: Amazon Redshift ODBC Driver
```

```
Driver Version: 1.4.27.1000
Rsql Version: 1.0.1
Redshift Version: 1.0.29306
Type "help" for help.

(testcluster) testuser@dev=>
```

デフォルトの認証情報プロバイダーチェーンで DSN 接続を使用する

デフォルトの認証情報プロバイダーチェーンを使用して接続するには、IAM プロパティのみを指定します。Amazon Redshift RSQL は、「AWS SDK for Java」の「AWS 認証情報の使用」で説明されている順序で認証情報の取得を試みます。チェーン内の少なくとも 1 つのプロバイダーが GetClusterCredentials のアクセス許可を持っている必要があります。これは、ECS コンテナから接続する場合などに便利です。

```
[iamcredentials]
Driver=Default
Host=testcluster.example.com
Database=dev
DbUser=testuser
ClusterID=rsqltestcluster
Region=us-east-1
IAM=1
```

Amazon Redshift RSQL メタコマンド

Amazon Redshift RSQL メタコマンドは、データベース、または特定のデータベースオブジェクトに関する情報レコードを返します。結果には、さまざまな列とメタデータを含めることができます。その他のコマンドは、特定のアクションを実行します。これらのコマンドの先頭にはバックスラッシュが付きます。

\d[S+]

ユーザーが作成したローカルテーブル、レギュラービュー、レイトバインディングビュー、マテリアライズドビューを一覧表示します。\dS も\dと同様に、テーブルとビューを一覧表示しますが、システムオブジェクトは返されたレコードに含まれます。+の結果、一覧表示されたすべてのオブジェクトの追加メタデータ列descriptionが表示されます。次に、コマンドの結果として返されるサンプルレコードを示します。

```
List of relations
```

```
 schema |   name    | type  |  owner
--------+-----------+-------+---------
 public | category  | table | awsuser
 public | date      | table | awsuser
 public | event     | table | awsuser
 public | listing   | table | awsuser
 public | sales     | table | awsuser
 public | users     | table | awsuser
 public | venue     | table | awsuser
(7 rows)
```

\d[S+] 名前

テーブル、ビュー、またはインデックスについて説明します。列の名前と型が含まれます。また、diststyle、バックアップ設定、作成日 (2018 年 10 月以降に作成されたテーブル)、および制約条件を提供します。たとえば、\dS+ sampleはオブジェクトのプロパティを返します。S+を付けると、返されたレコードに含まれる追加の列が表示されます。

```
Table "public.sample"
 Column |            Type             |   Collation    | Nullable | Default Value |
 Encoding  | DistKey | SortKey
--------+-----------------------------+----------------+----------+---------------
+-----------+---------+---------
 col1   | smallint                    |                | NO       |               |
 none      | t       | 1
 col2   | character(100)              | case_sensitive | YES      |               |
 none      | f       | 2
 col3   | character varying(100)      | case_sensitive | YES      |               |
 text32k   | f       | 3
 col4   | timestamp without time zone |                | YES      |               |
 runlength | f       | 0
 col5   | super                       |                | YES      |               |
 zstd      | f       | 0
 col6   | bigint                      |                | YES      |               |
 az64      | f       | 0

Diststyle: KEY
Backup: YES
Created: 2021-07-20 19:47:27.997045
Unique Constraints:
    "sample_pkey" PRIMARY KEY (col1)
    "sample_col2_key" UNIQUE (col2)
Foreign-key constraints:
```

```
    "sample_col2_fkey" FOREIGN KEY (col2) REFERENCES lineitem(l_orderkey)
```

テーブルのディストリビューションスタイル、またはディストスタイルは、KEY、AUTO、EVEN または ALL のいずれかです。

バックアップは、スナップショットの作成時にテーブルがバックアップされるかどうかを示します。有効な値は YES または NO です。

作成は、テーブルが作成された時刻のタイムスタンプです。2018 年 11 月より前に作成された Amazon Redshift テーブルでは、作成日は使用できません。この日付より前に作成されたテーブルには「なし」と表示されます (使用不可) 。

ユニーク制約は、テーブルにユニーク制約とプライマリキー制約を一覧表示します。

外部キー制約は、テーブルに外部キー制約を一覧表示します。

\dC[+] [パターン]

キャストを一覧表示します。ソースタイプ、ターゲットタイプ、およびキャストが暗黙的かどうかが含まれます。

次に、\dC+からの結果のサブセットを示します。

```
List of casts
         source type         |         target type         |      function       |
 implicit?   | description
-----------------------------+-----------------------------+---------------------
+---------------+-------------
 "char"                      | character                   | bpchar              | in
 assignment |
 "char"                      | character varying           | text                | in
 assignment |
 "char"                      | integer                     | int4                | no
          |
 "char"                      | text                        | text                | yes
          |
 "path"                      | point                       | point               | no
          |
 "path"                      | polygon                     | polygon             | in
 assignment |
 abstime                     | date                        | date                | in
 assignment |
```

```
abstime                     | integer                     | (binary coercible)  | no
         |
abstime                     | time without time zone      | time                | in
assignment |
abstime                     | timestamp with time zone    | timestamptz         | yes
         |
abstime                     | timestamp without time zone | timestamp           | yes
         |
bigint                      | bit                         | bit                 | no
         |
bigint                      | boolean                     | bool                | yes
         |
bigint                      | character                   | bpchar              | in
assignment |
bigint                      | character varying           | text                | in
assignment |
bigint                      | double precision            | float8              | yes
         |
bigint                      | integer                     | int4                | in
assignment |
bigint                      | numeric                     | numeric             | yes
         |
bigint                      | oid                         | oid                 | yes
         |
bigint                      | real                        | float4              | yes
         |
bigint                      | regclass                    | oid                 | yes
         |
bigint                      | regoper                     | oid                 | yes
         |
bigint                      | regoperator                 | oid                 | yes
         |
bigint                      | regproc                     | oid                 | yes
         |
bigint                      | regprocedure                | oid                 | yes
         |
bigint                      | regtype                     | oid                 | yes
         |
bigint                      | smallint                    | int2                | in
assignment |
bigint                      | super                       | int8_partiql        | in
assignment |
```

\dd[S] [パターン]

他の場所では表示されないオブジェクトの説明が表示されます。

\de

外部テーブルを一覧表示します。これには、AWS Glue Data Catalog のテーブル、Hive メタストア、Amazon RDS/Aurora MySQL、Amazon RDS/Aurora PostgreSQL、Amazon Redshift のデータ共有テーブルからの連携テーブルが含まれます。

\de 名前

外部テーブルについて説明します。

次のサンプルは、AWS Glue外部テーブルを示しています。

```
# \de spectrum.lineitem
                           Glue External table "spectrum.lineitem"
     Column      | External Type | Redshift Type | Position | Partition Key | Nullable
-----------------+---------------+---------------+----------+---------------+----------
 l_orderkey      | bigint        | bigint        | 1        | 0             |
 l_partkey       | bigint        | bigint        | 2        | 0             |
 l_suppkey       | int           | int           | 3        | 0             |
 l_linenumber    | int           | int           | 4        | 0             |
 l_quantity      | decimal(12,2) | decimal(12,2) | 5        | 0             |
 l_extendedprice | decimal(12,2) | decimal(12,2) | 6        | 0             |
 l_discount      | decimal(12,2) | decimal(12,2) | 7        | 0             |
 l_tax           | decimal(12,2) | decimal(12,2) | 8        | 0             |
 l_returnflag    | char(1)       | char(1)       | 9        | 0             |
 l_linestatus    | char(1)       | char(1)       | 10       | 0             |
 l_shipdate      | date          | date          | 11       | 0             |
 l_commitdate    | date          | date          | 12       | 0             |
 l_receiptdate   | date          | date          | 13       | 0             |
 l_shipinstruct  | char(25)      | char(25)      | 14       | 0             |
 l_shipmode      | char(10)      | char(10)      | 15       | 0             |
 l_comment       | varchar(44)   | varchar(44)   | 16       | 0             |

Location: s3://redshiftbucket/kfhose2019/12/31
Input_format: org.apache.hadoop.mapred.TextInputFormat
Output_format: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
Serialization_lib: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
Serde_parameters: {"field.delim":"|","serialization.format":"|"}
```

```
Parameters:
 {"EXTERNAL":"TRUE","numRows":"178196721475","transient_lastDdlTime":"1577771873"}
```

Hive メタストアテーブル。

```
# \de emr.lineitem
                    Hive Metastore External Table "emr.lineitem"
     Column      | External Type | Redshift Type | Position | Partition Key | Nullable
-----------------+---------------+---------------+----------+---------------+----------
 l_orderkey      | bigint        | bigint        | 1        | 0             |
 l_partkey       | bigint        | bigint        | 2        | 0             |
 l_suppkey       | int           | int           | 3        | 0             |
 l_linenumber    | int           | int           | 4        | 0             |
 l_quantity      | decimal(12,2) | decimal(12,2) | 5        | 0             |
 l_extendedprice | decimal(12,2) | decimal(12,2) | 6        | 0             |
 l_discount      | decimal(12,2) | decimal(12,2) | 7        | 0             |
 l_tax           | decimal(12,2) | decimal(12,2) | 8        | 0             |
 l_returnflag    | char(1)       | char(1)       | 9        | 0             |
 l_linestatus    | char(1)       | char(1)       | 10       | 0             |
 l_commitdate    | date          | date          | 11       | 0             |
 l_receiptdate   | date          | date          | 12       | 0             |
 l_shipinstruct  | char(25)      | char(25)      | 13       | 0             |
 l_shipmode      | char(10)      | char(10)      | 14       | 0             |
 l_comment       | varchar(44)   | varchar(44)   | 15       | 0             |
 l_shipdate      | date          | date          | 16       | 1             |

Location: s3://redshiftbucket/cetas
Input_format: org.apache.hadoop.hive.ql.io.parquet.MapredParquetInputFormat
Output_format: org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat
Serialization_lib: org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe
Serde_parameters: {"serialization.format":"1"}
Parameters: {"EXTERNAL":"TRUE", "numRows":"4307207",
 "transient_lastDdlTime":"1626990007"}
```

PostgreSQL 外部テーブル。

```
# \de pgrsql.alltypes
                                Postgres Federated Table "pgrsql.alltypes"
 Column |        External Type        |        Redshift Type        | Position |
 Partition Key | Nullable
```

```
--------+-----------------------------+-----------------------------+----------
+---------------+----------
 col1   | bigint                      | bigint                      | 1        | 0
        |
 col2   | bigint                      | bigint                      | 2        | 0
        |
 col5   | boolean                     | boolean                     | 3        | 0
        |
 col6   | box                         | varchar(65535)              | 4        | 0
        |
 col7   | bytea                       | varchar(65535)              | 5        | 0
        |
 col8   | character(10)               | character(10)               | 6        | 0
        |
 col9   | character varying(10)       | character varying(10)       | 7        | 0
        |
 col10  | cidr                        | varchar(65535)              | 8        | 0
        |
 col11  | circle                      | varchar(65535)              | 9        | 0
        |
 col12  | date                        | date                        | 10       | 0
        |
 col13  | double precision            | double precision            | 11       | 0
        |
 col14  | inet                        | varchar(65535)              | 12       | 0
        |
 col15  | integer                     | integer                     | 13       | 0
        |
 col16  | interval                    | varchar(65535)              | 14       | 0
        |
 col17  | json                        | varchar(65535)              | 15       | 0
        |
 col18  | jsonb                       | varchar(65535)              | 16       | 0
        |
 col19  | line                        | varchar(65535)              | 17       | 0
        |
 col20  | lseg                        | varchar(65535)              | 18       | 0
        |
 col21  | macaddr                     | varchar(65535)              | 19       | 0
        |
 col22  | macaddr8                    | varchar(65535)              | 20       | 0
        |
 col23  | money                       | varchar(65535)              | 21       | 0
        |
```

```
col24  | numeric                     | numeric(38,20)              | 22       | 0
       |
col25  | path                        | varchar(65535)              | 23       | 0
       |
col26  | pg_lsn                      | varchar(65535)              | 24       | 0
       |
col28  | point                       | varchar(65535)              | 25       | 0
       |
col29  | polygon                     | varchar(65535)              | 26       | 0
       |
col30  | real                        | real                        | 27       | 0
       |
col31  | smallint                    | smallint                    | 28       | 0
       |
col32  | smallint                    | smallint                    | 29       | 0
       |
col33  | integer                     | integer                     | 30       | 0
       |
col34  | text                        | varchar(65535)              | 31       | 0
       |
col35  | time without time zone      | varchar(65535)              | 32       | 0
       |
col36  | time with time zone         | varchar(65535)              | 33       | 0
       |
col37  | timestamp without time zone | timestamp without time zone | 34       | 0
       |
col38  | timestamp with time zone    | timestamp with time zone    | 35       | 0
       |
col39  | tsquery                     | varchar(65535)              | 36       | 0
       |
col40  | tsvector                    | varchar(65535)              | 37       | 0
       |
col41  | txid_snapshot               | varchar(65535)              | 38       | 0
       |
col42  | uuid                        | varchar(65535)              | 39       | 0
       |
col43  | xml                         | varchar(65535)              | 40       | 0
       |
```

\df[anptw][S+] [パターン]

さまざまなタイプの関数を一覧表示します。例えば、コマンド\dfは関数のリストを返します。結果には、名前、返されたデータタイプ、アクセス権、追加のメタデータなどのプロパティが含まれます。関数の種類には、トリガー、ストアドプロシージャ、window 関数、およびその他のものがあります。例えばS+をコマンド\dfantS+に追加すると、owner、security、およびaccess privilegesなどのメタデータ列が追加されます。

\dL[S+] [パターン]

データベースに関連付けられている手続き言語に関するデータを一覧表示します。情報には、plpgsql などの名前と、信頼できるかどうかを含んだ追加のメタデータ、アクセス権限、および説明が含まれます。サンプル呼び出しは、例えば\dLS+であり、言語とそのプロパティが一覧表示されます。S+をコマンドに追加すると、call handlerおよびaccess privilegesなどのメタデータ列が追加されます。

サンプル結果:

```
List of languages
   name    | trusted | internal language |      call handler       |
      validator                          | access privileges |          description

-----------+---------+-------------------+-------------------------
+-------------------------------------------------------------+-------------------
+--------------------------------
 c         | f       | t                 | -                       |
 fmgr_c_validator(oid)                                         |                 |
 Dynamically-loaded C functions
 exfunc    | f       | f                 | exfunc_call_handler()   | -
                                         | rdsdb=U/rdsdb     |
 internal  | f       | t                 | -                       |
 fmgr_internal_validator(oid)                                  |                 |
 Built-in functions
 mlfunc    | f       | f                 | mlfunc_call_handler()   | -
                                         | rdsdb=U/rdsdb     |
 plpgsql   | t       | f                 | plpgsql_call_handler()  |
 plpgsql_validator(oid)                                        |                 |
 plpythonu | f       | f                 | plpython_call_handler() |
 plpython_compiler(cstring,cstring,cstring,cstring,cstring) | rdsdb=U/rdsdb     |
 sql       | t       | t                 | -                       |
 fmgr_sql_validator(oid)                                       | =U/rdsdb          | SQL-
language functions
```

\dm[S+] [パターン]

マテリアライズドビューを一覧表示します。例えば、\dmS+は、マテリアライズドビューとそのプロパティを一覧表示します。S+をコマンドに追加すると、メタデータ列が追加されます。

\dn[S+] [パターン]

スキーマを一覧表示します。例えば、S+を\dnS+のコマンドに追加すると、descriptionおよびaccess privilegesなどの追加のメタデータ列が追加されます。

\dp [パターン]

テーブル、ビューおよびシーケンスのアクセス権限を一覧表示します。

\dt[S+] [パターン]

テーブルの一覧表示 例えば、S+をコマンド\dtS+に追加すると、この場合descriptionのようなメタデータ列が追加されます。

\du

データベースのユーザーを一覧表示します。名前とスーパーユーザーなどのロール、属性が含まれます。

\dv[S+] [パターン]

ビューを一覧表示します。スキーマ、タイプ、および所有者データが含まれます。例えば、S+をコマンド\dvS+に追加すると、メタデータ列が追加されます。

\H

HTML 出力をオンにします。これは、フォーマットされた結果をすばやく返す場合に便利です。例えば、select * from sales; \Hは、売上表の結果を HTML で返します。結果を表形式に戻すには、\qまたは quiet を使用します。

\i

ファイルからコマンドを実行します。例えば、作業ディレクトリに rsql_steps.sql があると仮定すると、次のものがファイルのコマンドを実行します。\i rsql_steps.sql。

\l[+] [パターン]

データベースを一覧表示します。所有者、エンコード、および追加情報が含まれます。

\q

quit または \q コマンドを実行して、データベースセッションからログオフし、RSQL を終了します。

\sv[+] ビュー名

ビューの定義を表示します。

\timing

例えば、クエリのランタイムが表示されます。

\z [パターン]

\dpと同じ出力です。

\?

ヘルプ情報を表示します。オプションのパラメータにより、表示する項目を指定します。

\EXIT

すべてのデータベースセッションからログオフし、Amazon Redshift RSQL を終了します。加えて、オプションの終了コードを指定することも可能です。例えば \EXIT 15 では、Amazon Redshift の RSQL ターミナルを終了し、その際に終了コードとして 15 を返します。

次の例は、接続からの出力と RSQL からの終了を示しています。

```
% rsql -D testuser
DSN Connected
DBMS Name: Amazon Redshift
Driver Name: Amazon Redshift ODBC Driver
Driver Version: 1.4.34.1000
Rsql Version: 1.0.1
Redshift Version: 1.0.29306
Type "help" for help.
```

```
(testcluster) user1@dev=# \exit 15

% echo $?
15
```

\EXPORT

後続の SQL SELECT ステートメントによって返されたデータベース情報を保存するために RSQL が使用するエクスポートファイルの名前を指定します。

export_01.sql

```
\export report file='E:\\accounts.out'
\rset rformat off
\rset width 1500
\rset heading "General Title"
\rset titledashes on
select * from td_dwh.accounts;
\export reset
```

コンソール出力

```
Rformat is off.
Target width is 1500.
Heading is set to: General Title
Titledashes is on.
(exported 40 rows)
```

\LOGON

データベースに接続します。接続パラメータは、位置指定の構文を使って指定することも、接続文字列として指定することもできます。

コマンドの構文は次のとおりです。`\logon {[DBNAME|- USERNAME|- HOST|- PORT|- [PASSWORD]] | conninfo}`

DBNAME は、接続先に指定するデータベースの名前です。USERNAME は、データベースへの接続に使用するユーザー名です。HOST のデフォルト値は `localhost` です。PORT のデフォルト値は 5439 です。

\LOGON コマンドでホスト名を指定した場合、その名前が、追加の \LOGON コマンドでのデフォルトのホスト名になります。デフォルトのホスト名を変更するには、追加の HOST コマンドで新しく \LOGON を指定します。

user1 に対する \LOGON コマンドの出力例を以下に示します。

```
(testcluster) user1@redshiftdb=# \logon dev
DBMS Name: Amazon Redshift
Driver Name: Amazon Redshift ODBC Driver
Driver Version: 1.4.27.1000
Rsql Version: 1.0.1
You are now connected to database "dev" as user "user1".
(testcluster) user1@dev=#
```

user2に対する出力例

```
(testcluster) user1@dev=# \logon dev user2 testcluster2.example.com
Password for user user2:
DBMS Name: Amazon Redshift
Driver Name: Amazon Redshift ODBC Driver
Driver Version: 1.4.27.1000
Rsql Version: 1.0.1
You are now connected to database "dev" as user "user2" on host
 "testcluster2.example.com" at port "5439".
(testcluster2) user2@dev=#
```

\REMARK

これは \echo コマンドの拡張です。\REMARK は、指定された文字列を出力ストリームに表示します。\REMARK　は \echo を拡張して、出力を改行する機能を追加したものです。

次の例は、コマンドからの出力例を示しています。

```
(testcluster) user1@dev=# \remark 'hello//world'
hello
world
```

\RSET

コマンド \rset はコマンドのパラメータと変数を設定します。\rset には、インタラクティブモードとバッチモードの両方があります。-x、または引数 (--<arg> など) といった bash オプションとしてのオプションをサポートしません。

このコマンドは、以下のような変数を設定します。

- ERRORLEVEL
- HEADING および RTITLE
- RFORMAT
- MAXERROR
- TITLEDASHES
- WIDTH

以下の例は、HEADING を指定します。

```
\rset heading "Winter Sales Report"
```

その他の \rset 使用方法例については、Amazon Redshift RSQL 変数トピックに例がいくつか挙げられています。

\RUN

指定されたファイルに含まれる Amazon Redshift RSQL スクリプトを実行します。\RUNは\iコマンドを拡張して、ファイル内のヘッダー行をスキップするオプションを追加します。

ファイル名にカンマ、セミコロン、またはスペースが含まれている場合は、それを一重引用符で囲みます。また、ファイル名の後に続くテキストは、引用符で囲まれます。UNIX では、ファイル名の大文字と小文字が区別されます。Windows では、ファイル名の大文字と小文字は区別されません。

次の例は、コマンドからの出力例を示しています。

```
(testcluster) user1@dev=# \! cat test.sql
select count(*) as lineitem_cnt from lineitem;
select count(*) as customer_cnt from customer;
select count(*) as orders_cnt from orders;



(testcluster) user1@dev=# \run file=test.sql
 lineitem_cnt
--------------
      4307207
```

```
(1 row)

 customer_cnt
--------------
     37796166
(1 row)

 orders_cnt
------------
          0
(1 row)


(testcluster) user1@dev=# \run file=test.sql skip=2
2 records skipped in RUN file.
 orders_cnt
------------
          0
(1 row)
```

\OS

\! コマンドのエイリアスです。\OS は、パラメータとして渡されたオペレーションシステムコマンドを実行します。OS コマンドの実行後、コントロールは Amazon Redshift RSQL に戻されます。例えば、次のコマンドを実行して、現在のシステム日付時刻を表示し、RSQL ターミナルに戻ることができます。\os date。

```
(testcluster) user1@dev=# \os date
Tue Sep 7 20:47:54 UTC 2021
```

\GOTO

Amazon Redshift RSQL 用の新しいコマンドです。\GOTOは、介在するコマンドをすべてスキップし、指定した\LABELで処理を再開します。\LABELは前方リファレンスでなければなりません。辞書的に\GOTOの前にある\LABELにはジャンプできません。

出力例を次に示します。

```
(testcluster) user1@dev=# \! cat test.sql
select count(*) as cnt from lineitem \gset
select :cnt as cnt;
```

```
\if :cnt > 100
    \goto LABELB
\endif

\label LABELA
\remark 'this is label LABELA'
\label LABELB
\remark 'this is label LABELB'


(testcluster) user1@dev=# \i test.sql
   cnt
---------
 4307207
(1 row)

\label LABELA ignored
\label LABELB processed
this is label LABELB
```

\LABEL

Amazon Redshift RSQL 用の新しいコマンドです。\LABELは、\GOTOコマンドのターゲットとして、プログラムを実行するためのエントリーポイントを確立します。

次の例は、コマンドからの出力例を示しています。

```
(testcluster) user1@dev=# \! cat test.sql
select count(*) from lineitem limit 5;
\goto LABELB
\remark "this step was skipped by goto label";
\label LABELA
\remark 'this is label LABELA'
\label LABELB
\remark 'this is label LABELB'



(testcluster) user1@dev=# \i testgoto.sql
  count
 4307193
(1 row)
```

```
\label LABELA ignored
\label LABELB processed
this is label LABELB
```

\IF (\ELSEIF, \ELSE, \ENDIF)

\IF、およびそれと関連するコマンドは、入力スクリプトの一部を条件付きで実行します。PSQL\if(\elif、\else、\endif) コマンドの拡張機能です。\IFと\ELSEIFは、AND、OR、NOTの条件を含むブール表現をサポートします。

次の例は、コマンドからの出力例を示しています。

```
(testcluster) user1@dev=# \! cat test.sql
SELECT query FROM stv_inflight LIMIT 1 \gset
select :query as query;
\if :query > 1000000
    \remark 'Query id is greater than 1000000'
\elseif :query = 1000000
    \remark 'Query id is equal than 1000000'
\else
    \remark 'Query id is less than 1000000'
\endif


(testcluster) user1@dev=# \i test.sql
 query
--------
 994803
(1 row)

Query id is less than 1000000
```

ブランチロジックで ERRORCODE を使用する。

```
\if :'ERRORCODE' = '00000'
    \remark 'The statement was executed without error'
\else
    \remark :LAST_ERROR_MESSAGE
\endif
```

\IF ブロックで \GOTO を使用し、コードの実行方法を制御します。

Amazon Redshift RSQL 変数

キーワードには、RSQL の変数として機能するものがあります。それぞれを特定の値に設定するか、値を再設定することができます。ほとんどは \rset で設定されており、これにはインタラクティブモードとバッチモードがあります。コマンドは、小文字または大文字で定義できます。

ACTIVITYCOUNT

最後に送信されたリクエストの影響を受けた行の数を示します。データ返すリクエストの場合、これはデータベースから RSQL に返された行の数です。この値は 0 または正の整数です。最大値は 18,446,744,073,709,551,615 です。

特別に処理された変数 ACTIVITYCOUNT は変数 ROW_COUNT に似ています。ただし、ROW_COUNTはコマンド完了時に SELECT、COPY、または UNLOAD について影響を受けた行の数をクライアントアプリケーションに報告しませんが、ACTIVITYCOUNT は報告します。

activitycount_01.sql:

```
select viewname, schemaname
from pg_views
where schemaname = 'not_existing_schema';
\if :ACTIVITYCOUNT = 0
\remark 'views do not exist'
\endif
```

コンソール出力:

```
viewname | schemaname
----------+------------
(0 rows)

views do not exist
```

ERRORLEVEL

エラーに重要度レベルを割り当てます。この重大度を使用して、アクションの方針を決定します。ERRORLEVEL コマンドが使用されていない場合、その値はデフォルトで ON になります。

errorlevel_01.sql:

```
\rset errorlevel 42P01 severity 0
```

```
select * from tbl;

select 1 as col;

\echo exit
\quit
```

コンソール出力:

```
Errorlevel is on.
rsql: ERROR: relation "tbl" does not exist
(1 row)

col
1

exit
```

HEADING および RTITLE

ユーザーがレポートの上部に表示されるヘッダーを指定できるようにします。RSET RTITLE コマンドによって指定されたヘッダーには、クライアントコンピュータの現在のシステム日付が自動的に含められます。

rset_heading_rtitle_02.rsql の内容:

```
\remark Starting...
\rset rtitle "Marketing Department||Confidential//Third Quarter//Chicago"
\rset width 70
\rset rformat on
select * from rsql_test.tbl_currency order by id limit 2;
\exit
\remark Finishing...
```

コンソール出力:

```
Starting...
Rtitle is set to: &DATE||Marketing Department||Confidential//Third Quarter//Chicago
 (Changes will take effect after RFORMAT is
switched ON)
Target width is 70.
```

```
Rformat is on.
09/11/20       Marketing       Department Confidential
                  Third Quarter
                     Chicago
id  | bankid  | name |      start_date
100 |       1 | USD | 2020-09-11 10:51:39.106905
110 |       1 | EUR | 2020-09-11 10:51:39.106905
(2 rows)

Press any key to continue . . .
```

MAXERROR

最大エラー重大度レベルを指定します。このレベルを超えると、RSQL がジョブ処理を終了します。リターンコードは、RSQL が各ジョブまたはタスクの完了後にクライアントオペレーティングシステムに返す整数値です。リターンコードの値は、ジョブまたはタスクの完了ステータスを示します。指定された maxerror 値を超えるエラー重大度レベルを生成するステートメントがスクリプトに含まれている場合、RSQL はただちに終了します。従って、エラー重大度レベル 8 で RSQL を終了させるには、RSET MAXERROR 7を使用します。

maxerror_01.sql の内容:

```
\rset maxerror 0

select 1 as col;

\quit
```

コンソール出力:

```
Maxerror is default.
(1 row)

col
1
```

RFORMAT

ユーザーがフォーマットコマンドの設定を適用するかどうかを指定できるようにします。

rset_rformat.rsql の内容:

```
\remark Starting...
\pset border 2
\pset format wrapped
\pset expanded on
\pset title 'Great Title'
select * from rsql_test.tbl_long where id = 500;
\rset rformat
select * from rsql_test.tbl_long where id = 500;
\rset rformat off
select * from rsql_test.tbl_long where id = 500;
\rset rformat on
select * from rsql_test.tbl_long where id = 500;
\exit
\remark Finishing...
```

コンソール出力:

```
Starting...
Border style is 2. (Changes will take effect after RFORMAT is switched ON)
Output format is wrapped. (Changes will take effect after RFORMAT is switched ON)
Expanded display is on. (Changes will take effect after RFORMAT is switched ON)
Title is "Great Title". (Changes will take effect after RFORMAT is switched ON)
id  |                                                                  long_string
500 | In general, the higher the number the more borders and lines the tables will
 have, but details depend on the particular
format.
(1 row)

Rformat is on.
Great Title
+-[ RECORD
 1 ]+-----------------------------------------------------------------------------------------------
-----------+
| id          | 500
|
| long_string | In general, the higher the number the more borders and lines the tables
 will have, but details depend on the
particular format. |
+-------------
+-----------------------------------------------------------------------------------------------
-----------+

Rformat is off.
```

```
id  |                                                                long_string
500 | In general, the higher the number the more borders and lines the tables will
 have, but details depend on the particular format.
(1 row)

Rformat is on.
Great Title
+-[ RECORD
 1 ]+-------------------------------------------------------------------------------------------------
-----------+
| id           | 500
|
| long_string | In general, the higher the number the more borders and lines the tables
 will have, but details depend on the
particular format. |
+-------------
+-------------------------------------------------------------------------------------------------
-----------+
Press any key to continue . . .
```

ROW_COUNT

前回のクエリの影響を受けたレコードの数を取得します。以下のコードフラグメントにあるように、これは結果をチェックするために使用されるのが一般的です。

```
SET result = ROW_COUNT;

IF result = 0
...
```

TITLEDASHES

このコントロールは、ユーザーが SQL ステートメントに返される列データの上にダッシュ文字のラインを出力するかどうかを指定できるようにします。

例:

```
\rset titledashes on
select dept_no, emp_no, salary from rsql_test.EMPLOYEE
where dept_no = 100;
\rset titledashes off
select dept_no, emp_no, salary from rsql_test.EMPLOYEE
```

```
where dept_no = 100;
```

コンソール出力:

```
dept_no      emp_no          salary
------------ ----------- --------------------
100          1000346        1300.00
100          1000245        5000.00
100          1000262        2450.00

dept_no     emp_no          salary
100          1000346        1300.00
100          1000245        5000.00
100          1000262        2450.00
```

WIDTH

出力形式を折り返しに設定し、レポート内の各行のターゲット幅を指定します。パラメータを指定しない場合は、形式とターゲット幅両方の現在の設定を返します。

rset_width_01.rsql の内容:

```
\echo Starting...
\rset width
\rset width 50
\rset width
\quit
\echo Finishing...
```

コンソール出力:

```
Starting...
Target width is 75.
Target width is 50.
Target width is 50.
Press any key to continue . . .
```

パラメータ指定時の例:

```
\echo Starting...
\rset rformat on
\pset format wrapped
```

```
select * from rsql_test.tbl_long where id = 500;
\rset width 50
select * from rsql_test.tbl_long where id = 500;
\quit
\echo Finishing...
```

コンソール出力:

```
Starting...
Rformat is on.
Output format is wrapped.
id  |                                              long_string
500 | In general, the higher the number the more borders and lines the ta.
    |.bles will have, but details depend on the particular format.
(1 row)

Target width is 50.
id  |                                              long_string
500 | In general, the higher the number the more.
    |. borders and lines the tables will have, b.
    |.ut details depend on the particular format.
    |..
(1 row)
Press any key to continue . . .
```

Amazon Redshift RSQL エラーコード

成功メッセージ、警告、例外:

エラーコード	エラークラス	条件名
00000	クラス 00 - 無事修了	successful_completion
01000	クラス 01 - 警告	警告
0100C	クラス 01 - 警告	dynamic_result_sets_returned
01008	クラス 01 - 警告	implicit_zero_bit_padding
01003	クラス 01 - 警告	null_value_eliminated_in_set_function
01007	クラス 01 - 警告	privilege_not_granted

エラーコード	エラークラス	条件名
01006	クラス 01 - 警告	privilege_not_revoked
01004	クラス 01 - 警告	string_data_right_truncation
01P01	クラス 01 - 警告	deprecated_feature
02000	クラス 02 - データなし	no_data
02001	クラス 02 - データなし	no_additional_dynamic_result_sets_returned
03000	クラス 03 - SQL 文がまだ完成していません	sql_statement_not_yet_complete
08000	クラス 08 - 接続例外	connection_exception
08003	クラス 08 - 接続例外	connection_does_not_exist
08006	クラス 08 - 接続例外	connection_failure
08001	クラス 08 - 接続例外	sqlclient_unable_to_establish_sqlconnection
08004	クラス 08 - 接続例外	sqlserver_rejected_establishment_of_sqlconnection
08007	クラス 08 - 接続例外	transaction_resolution_unknown
08P01	クラス 08 - 接続例外	protocol_violation
09000	クラス 09 - トリガーされたアクションの例外	triggered_action_exception
0A000	クラス 0A - 機能はサポートされていません	feature_not_supported
0A000	クラス 0A - 機能はサポートされていません	feature_not_supported
0B000	クラス 0B - 無効なトランザクション開始	invalid_transaction_initiation

エラーコード	エラークラス	条件名
0F000	クラス 0F - ロケータ例外	locator_exception
0F001	クラス 0F - ロケータ例外	invalid_locator_specification
0L000	クラス 0L - 無効な権限付与者	invalid_grantor
0LP01	クラス 0L - 無効な権限付与者	invalid_grant_operation
0P000	クラス 0P - 無効なロール指定	invalid_role_specification
0Z000	クラス 0Z - 診断例外	diagnostics_exception
0Z002	クラス 0Z - 診断例外	stacked_diagnostics_accessed_without_active_handler
20000	クラス 20 - ケースが見つかりません	case_not_found
21000	クラス 21 - カーディナリティ違反	cardinality_violation

データ例外:

エラーコード	エラークラス	条件名
22000	クラス 22 - データ例外	data_exception
2202E	クラス 22 - データ例外	array_subscript_error
22021	クラス 22 - データ例外	character_not_in_repertoire
22008	クラス 22 - データ例外	datetime_field_overflow
22012	クラス 22 - データ例外	division_by_zero
22005	クラス 01 - 警告	error_in_assignment
2200B	クラス 01 - 警告	escape_character_conflict

エラーコード	エラークラス	条件名
22022	クラス 01 - 警告	indicator_overflow
22015	クラス 01 - 警告	interval_field_overflow
2201E	クラス 01 - 警告	invalid_argument_for_logarithm
2201F	クラス 01 - 警告	invalid_argument_for_power_function
2201G	クラス 01 - 警告	invalid_argument_for_width_bucket_function
22018	クラス 01 - 警告	invalid_character_value_for_cast
22007	クラス 01 - 警告	invalid_datetime_format
22019	クラス 01 - 警告	invalid_escape_character
2200D	クラス 01 - 警告	invalid_escape_octet
22025	クラス 01 - 警告	invalid_escape_sequence
22P06	クラス 01 - 警告	nonstandard_use_of_escape_character
22010	クラス 01 - 警告	invalid_indicator_parameter_value
22023	クラス 01 - 警告	invalid_parameter_value
2201B	クラス 01 - 警告	invalid_regular_expression
22009	クラス 01 - 警告	invalid_time_zone_displacement_value
2200C	クラス 01 - 警告	invalid_use_of_escape_character
2200G	クラス 01 - 警告	most_specific_type_mismatch
22004	クラス 01 - 警告	null_value_not_allowed
22002	クラス 01 - 警告	null_value_no_indicator_parameter
22003	クラス 01 - 警告	numeric_value_out_of_range

エラーコード	エラークラス	条件名
22026	クラス 01 - 警告	string_data_length_mismatch
22001	クラス 01 - 警告	string_data_right_truncation
22011	クラス 01 - 警告	substring_error
22027	クラス 01 - 警告	trim_error
22024	クラス 01 - 警告	unterminated_c_string
2200F	クラス 01 - 警告	zero_length_character_string
22P01	クラス 01 - 警告	floating_point_exception
22P02	クラス 01 - 警告	invalid_text_representation
22P03	クラス 01 - 警告	invalid_binary_representation
22P04	クラス 01 - 警告	bad_copy_file_format
22P05	クラス 01 - 警告	untranslatable_character

整合性制約違反:

エラーコード	エラークラス	条件名
23000	クラス 23 - 整合性制約違反	integrity_constraint_violation
23001	クラス 23 - 整合性制約違反	restrict_violation
23502	クラス 23 - 整合性制約違反	not_null_violation
23503	クラス 23 - 整合性制約違反	foreign_key_violation
23505	クラス 23 - 整合性制約違反	unique_violation
23514	クラス 23 - 整合性制約違反	check_violation

エラーコード	エラークラス	条件名
24000	クラス 24 - 無効なカーソル状態	invalid_cursor_state
01004	クラス 01 - 警告	string_data_right_truncation
25000	クラス 25 - 無効なトランザクション状態	invalid_transaction_state
25001	クラス 25 - 無効なトランザクション状態	active_sql_transaction
25002	クラス 25 - 無効なトランザクション状態	invalid_transaction_state
25008	クラス 25 - 無効なトランザクション状態	held_cursor_requires_same_isolation_level
25003	クラス 25 - 無効なトランザクション状態	inappropriate_access_mode_for_branch_transaction
25004	クラス 25 - 無効なトランザクション状態	inappropriate_isolation_level_for_branch_transaction
25005	クラス 25 - 無効なトランザクション状態	no_active_sql_transaction_for_branch_transaction
25006	クラス 25 - 無効なトランザクション状態	read_only_sql_transaction
25007	クラス 25 - 無効なトランザクション状態	no_active_sql_transaction_for_branch_transaction
25P01	クラス 25 - 無効なトランザクション状態	no_active_sql_transaction
25P02	クラス 25 - 無効なトランザクション状態	in_failed_sql_transaction

エラーコード	エラークラス	条件名
26000	クラス 26 - 無効な SQL ステートメント名	invalid_sql_statement_name
28000	クラス 28 - 無効な認可指定	invalid_authorization_specification
2B000	クラス 2B - 従属権限記述子がまだ存在します	dependent_privilege_descriptors_still_exist
2BP01	クラス 2B - 従属権限記述子がまだ存在します	dependent_objects_still_exist
2D000	クラス 2D - 無効なトランザクション終了	invalid_transaction_termination
2F000	クラス 2F - SQL ルーチン例外	sql_routine_exception
2F005	クラス 2F - SQL ルーチン例外	function_executed_no_return_statement
2F002	クラス 2F - SQL ルーチン例外	modifying_sql_data_not_permitted
2F003	クラス 2F - SQL ルーチン例外	prohibited_sql_statement_attempted
2F004	クラス 2F - SQL ルーチン例外	reading_sql_data_not_permitted
34000	クラス 34 - 無効なカーソル名	invalid_cursor_name
38000	クラス 38 - 外部ルーチン例外	external_routine_exception
38001	クラス 38 - 外部ルーチン例外	containing_sql_not_permitted
38002	クラス 38 - 外部ルーチン例外	modifying_sql_data_not_permitted
38003	クラス 38 - 外部ルーチン例外	prohibited_sql_statement_attempted
38004	クラス 38 - 外部ルーチン例外	reading_sql_data_not_permitted
39000	クラス 39 - 外部ルーチン呼び出しの例外	external_routine_invocation_exception

エラーコード	エラークラス	条件名
39001	クラス 39 - 外部ルーチン呼び出しの例外	invalid_sqlstate_returned
39004	クラス 39 - 外部ルーチン呼び出しの例外	null_value_not_allowed
39P01	クラス 39 - 外部ルーチン呼び出しの例外	trigger_protocol_violated
39P02	クラス 39 - 外部ルーチン呼び出しの例外	srf_protocol_violated
3D000	クラス 3D - 無効なカタログ名	invalid_catalog_name
3F000	クラス 3F - 無効なスキーマ名	invalid_schema_name
42000	クラス 42 - 構文エラーまたはアクセス規則違反	syntax_error_or_access_rule_violation
42601	クラス 42 - 構文エラーまたはアクセス規則違反	syntax_error
42501	クラス 42 - 構文エラーまたはアクセス規則違反	insufficient_privilege
42846	クラス 42 - 構文エラーまたはアクセス規則違反	cannot_coerce
42803	クラス 42 - 構文エラーまたはアクセス規則違反	grouping_error
42830	クラス 42 - 構文エラーまたはアクセス規則違反	invalid_foreign_key
42602	クラス 42 - 構文エラーまたはアクセス規則違反	invalid_name

エラーコード	エラークラス	条件名
42622	クラス 42 - 構文エラーまたはアクセス規則違反	name_too_long
42939	クラス 42 - 構文エラーまたはアクセス規則違反	reserved_name
42804	クラス 42 - 構文エラーまたはアクセス規則違反	datatype_mismatch
42P18	クラス 42 - 構文エラーまたはアクセス規則違反	indeterminate_datatype
42809	クラス 42 - 構文エラーまたはアクセス規則違反	wrong_object_type
42703	クラス 42 - 構文エラーまたはアクセス規則違反	undefined_column
42883	クラス 42 - 構文エラーまたはアクセス規則違反	undefined_function
42P01	クラス 42 - 構文エラーまたはアクセス規則違反	undefined_table
42P02	クラス 42 - 構文エラーまたはアクセス規則違反	undefined_parameter
42704	クラス 42 - 構文エラーまたはアクセス規則違反	undefined_object
42701	クラス 42 - 構文エラーまたはアクセス規則違反	duplicate_column
42P03	クラス 42 - 構文エラーまたはアクセス規則違反	duplicate_cursor
42P04	クラス 42 - 構文エラーまたはアクセス規則違反	duplicate_database

エラーコード	エラークラス	条件名
42723	クラス 42 - 構文エラーまたはアクセス規則違反	duplicate_function
42P05	クラス 42 - 構文エラーまたはアクセス規則違反	duplicate_prepared_statement
42P06	クラス 42 - 構文エラーまたはアクセス規則違反	duplicate_schema
42P07	クラス 42 - 構文エラーまたはアクセス規則違反	duplicate_table
42712	クラス 42 - 構文エラーまたはアクセス規則違反	duplicate_alias
42710	クラス 42 - 構文エラーまたはアクセス規則違反	duplicate_object
42702	クラス 42 - 構文エラーまたはアクセス規則違反	ambiguous_column
42725	クラス 42 - 構文エラーまたはアクセス規則違反	ambiguous_function
42P08	クラス 42 - 構文エラーまたはアクセス規則違反	ambiguous_parameter
42P09	クラス 42 - 構文エラーまたはアクセス規則違反	ambiguous_alias
42P10	クラス 42 - 構文エラーまたはアクセス規則違反	invalid_column_reference
42611	クラス 42 - 構文エラーまたはアクセス規則違反	invalid_column_definition
42P11	クラス 42 - 構文エラーまたはアクセス規則違反	invalid_cursor_definition

エラーコード	エラークラス	条件名
42P12	クラス 42 - 構文エラーまたはアクセス規則違反	invalid_database_definition
42P13	クラス 42 - 構文エラーまたはアクセス規則違反	invalid_function_definition
42P14	クラス 42 - 構文エラーまたはアクセス規則違反	invalid_prepared_statement_definition
42P15	クラス 42 - 構文エラーまたはアクセス規則違反	invalid_schema_definition
42P16	クラス 42 - 構文エラーまたはアクセス規則違反	invalid_table_definition
42P17	クラス 42 - 構文エラーまたはアクセス規則違反	invalid_object_definition
44000	クラス 44 - WITH CHECK OPTION 違反	with_check_option_violation
53000	クラス 53 - リソース不足	insufficient_resources
53100	クラス 53 - リソース不足	disk_full
53200	クラス 53 - リソース不足	out_of_memory
53300	クラス 53 - リソース不足	too_many_connections
54000	クラス 54 - プログラム制限を超過しました	program_limit_exceeded
54001	クラス 54 - プログラム制限を超過しました	statement_too_complex
54011	クラス 54 - プログラム制限を超過しました	too_many_columns

エラーコード	エラークラス	条件名
54023	クラス 54 - プログラム制限を超過しました	too_many_arguments
55000	クラス 55 - オブジェクトが前提条件の状態ではありません	object_not_in_prerequisite_state
55006	クラス 55 - オブジェクトが前提条件の状態ではありません	object_in_use
55P02	クラス 55 - オブジェクトが前提条件の状態ではありません	cant_change_runtime_param
55P03	クラス 55 - オブジェクトが前提条件の状態ではありません	lock_not_available
57000	クラス 57 - オペレーターの介入	operator_intervention
57014	クラス 57 - オペレーターの介入	query_canceled
57P01	クラス 57 - オペレーターの介入	admin_shutdown
57P02	クラス 57 - オペレーターの介入	crash_shutdown
57P03	クラス 57 - オペレーターの介入	cannot_connect_now
58000	クラス 58 - システムエラー (PostgreSQL の外部エラー)	system_error
58030	クラス 58 - システムエラー (PostgreSQL の外部エラー)	io_error
58P01	クラス 58 - システムエラー (PostgreSQL の外部エラー)	undefined_file

エラーコード	エラークラス	条件名
58P02	クラス 58 - システムエラー (PostgreSQL の外部エラー)	duplicate_file
F0000	クラス F0 - 設定ファイルエラー	duplicate_file
F0001	クラス F0 - 設定ファイルエラー	lock_file_exists
P0000	クラス P0 - PL/pgSQL エラー	plpgsql_error
P0001	クラス P0 - PL/pgSQL エラー	raise_exception
P0002	クラス P0 - PL/pgSQL エラー	no_data_found
P0003	クラス P0 - PL/pgSQL エラー	too_many_rows
XX000	クラス XX - 内部エラー	internal_error
XX001	クラス XX - 内部エラー	data_corrupted
XX002	クラス XX - 内部エラー	index_corrupted

Amazon Redshift RSQL 環境変数

Amazon Redshift RSQL では、環境変数を使用してデフォルトのパラメータ値を選択できます。

RSPASSWORD

> **Important**
>
> 一部のオペレーティングシステムでは管理者以外のユーザーがプロセスの環境変数を参照できるため、セキュリティ上の理由からこの環境変数を使用することは推奨されません。

Amazon Redshift RSQL が Amazon Redshift に接続する場合に使用するパスワードを設定します。この環境変数を設定するには、Amazon Redshift RSQL 1.0.4 以降がインストールされている必要があります。

RSQL では、RSPASSWORD が設定されている場合、RSPASSWORD を優先します。RSPASSWORD が設定されておらず、DSN を使用して接続する場合、RSQL は DSN ファイルのパラメータからパスワードを取得します。最終的に、RSPASSWORD が設定されておらず、DSN も使用しない場合、RSQL によって接続が試行された後、パスワードプロンプトが表示されます。

RSPASSWORD の設定例を次に示します。

```
export RSPASSWORD=TestPassw0rd
```

認証プロファイルを使用して Amazon Redshift に接続する

Amazon Redshift への接続が多数存在する場合には、すべての接続の設定を管理することが困難な場合があります。多くの場合、各 JDBC 接続または ODBC 接続は、専用の設定オプションを使用しています。認証プロファイルを使用すると、接続オプションをまとめて保存しておくことができます。これにより、ユーザーは接続に使用するプロファイルを選択でき、個々のオプションの設定を管理する必要がなくなります。プロファイルは、さまざまなシナリオやユーザータイプに適用できます。

認証プロファイルが作成されていれば、ユーザーはすぐに使用できるプロファイルを接続文字列に追加できます。これにより、各ロールとユースケースに適した設定で、Amazon Redshift に接続することが可能になります。

Amazon Redshift API の詳細については、「CreateAuthenticationProfile」を参照してください。

認証プロファイルの作成

AWS CLI から `create-authentication-profile` コマンドを使用して、認証プロファイルを作成します。これには、既存の Amazon Redshift クラスターと既存のデータベースがあることが前提です。ユーザーの認証情報には、Amazon Redshift データベースに接続するアクセス許可と、認証プロファイルを取得する権限が必要です。設定オプションは JSON 文字列として指定するか、JSON 文字列を含むファイルの参照を通じて提供します。

```
create-authentication-profile --authentication-profile-name<value: String> --
authentication-profile-content<value: String>
```

次の例では、`ExampleProfileName`という名前のプロファイルを作成します。ここでは、クラスター名やその他のオプション設定を定義するキーと値を、JSON 文字列として追加します。

```
create-authentication-profile --authentication-profile-name "ExampleProfileName"
 --authentication-profile-content "{\"AllowDBUserOverride\":\"1\",\"Client_ID
```

```
\":\"ExampleClientID\",\"App_ID\":\"ExampleAppID\",\"AutoCreate\":false,
\"enableFetchRingBuffer\":true,\"databaseMetadataCurrentDbOnly\":true}"
}
```

このコマンドは、JSON で指定した設定でプロファイルを作成します。次の結果が表示され、プロファイルが作成されたことを示します。

```
{"AuthenticationProfileName": "ExampleProfileName",
"AuthenticationProfileContent": "{\"AllowDBUserOverride\":\"1\",
\"Client_ID\":\"ExampleClientID\",\"App_ID\":\"ExampleAppID\",
\"AutoCreate\":false,\"enableFetchRingBuffer\":true,
\"databaseMetadataCurrentDbOnly\":true}" }
```

認証プロファイルの作成に関する制限事項とクォータ

各カスタマーには、認証プロファイルのクォータが 10 個あります。

認証プロファイルでは、特定のエラーが発生することがあります。例えば、既存の名前で新しいプロファイルを作成した場合や、プロファイルのクォータを超えた場合などに発生します。詳細については、「CreateAuthenticationProfile」を参照してください。

認証プロファイルストアには、JDBC、ODBC、および Python での接続文字列のための、特定のオプションキーと値を保存することはできません。

- `AccessKeyID`
- `access_key_id`
- `SecretAccessKey`
- `secret_access_key_id`
- `PWD`
- `Password`
- `password`

JDBC または ODBC 接続文字列の場合、キーまたは値 `AuthProfile` をプロファイルストアに保存することはできません() Python 接続の場合、`auth_profile`を保存することはできません。

認証プロファイルは Amazon DynamoDB に保存され、AWSにより管理されます。

認証プロファイルを使用した接続

作成した後の認証プロファイルには、JDBC バージョン 2.x の AuthProfile 用の接続オプションとして、プロファイル名を含めることができます。この接続オプションは、保存された設定を取得するために使用します。

```
jdbc:redshift:iam://endpoint:port/database?AuthProfile=<Profile-
Name>&AccessKeyID=<Caller-Access-Key>&SecretAccessKey=<Caller-Secret-Key>
```

JDBC URL 文字列の例を次に示します。

```
jdbc:redshift:iam://examplecluster:us-west-2/dev?
AuthProfile="ExampleProfile"&AccessKeyID="AKIAIOSFODNN7EXAMPLE"&SecretAccessKey="wJalrXUtnFEMI/
K7MDENG/bPxRfiCYEXAMPLEKEY"
```

JDBC URL の中で、AccessKeyIDおよび SecretAccessKey の両方を認証プロファイル名とともに指定します。

また、設定オプションをセミコロン区切り文字で区切ることもできます。例えば、ログ記録のオプションが含まれている次の例のようになります。

```
jdbc:redshift:iam://my_redshift_end_point:5439/dev?LogLevel=6;LogPath=/
tmp;AuthProfile=my_profile;AccessKeyID="AKIAIOSFODNN7EXAMPLE";SecretAccessKey="wJalrXUtnFEMI/
K7MDENG/bPxRfiCYEXAMPLEKEY"
```

> **Note**
>
> 認証プロファイルには機密情報を含めないでください。例えば、AccessKeyID または SecretAccessKey 値は、認証プロファイルには保存しないようにします。認証プロファイルストアには、シークレットキーの格納を禁止するルールがあります。機密情報に関連付けられたキーと値を格納しようとすると、エラーが発生します。

認証プロファイルの取得

既存の認証プロファイルを一覧表示するには、次のコマンドを実行します。

```
describe-authentication-profiles --authentication-profile-name <value: String>
```

2 つのプロファイルが取得された場合の例を以下に示します。プロファイル名を指定しない場合、すべてのプロファイルが返されます。

```
{ "AuthenticationProfiles": [ { "AuthenticationProfileName":
"testProfile1", "AuthenticationProfileContent": "{\"AllowDBUserOverride
\":\"1\",\"Client_ID\":\"ExampleClientID\",\"App_ID\":\"ExampleAppID
\",\"AutoCreate\":false,\"enableFetchRingBuffer\":true,
\"databaseMetadataCurrentDbOnly\":true}" }, { "AuthenticationProfileName":
"testProfile2", "AuthenticationProfileContent": "{\"AllowDBUserOverride
\":\"1\",\"Client_ID\":\"ExampleClientID\",\"App_ID\":\"ExampleAppID
\",\"AutoCreate\":false,\"enableFetchRingBuffer\":true,
\"databaseMetadataCurrentDbOnly\":true}" } ] }
```

Amazon Redshift での接続の問題のトラブルシューティング

SQL クライアントツールからクラスターへの接続で問題が発生した場合は、問題を絞り込むために確認できるいくつかの点があります。SSL またはサーバー証明書を使用している場合、接続の問題をトラブルシューティングしているときにまずこの複雑さを排除します。その後、解決策を見つけたらもう一度これを追加します。詳細については、「接続のセキュリティオプションを設定する」を参照してください。

アプリケーションに影響を与える可能性のある Amazon Redshift 機能の動作の変更については、「Amazon Redshift の動作の変更」を参照してください。

> **Important**
>
> Amazon Redshift は、SSL 証明書の管理方法を変更しました。SSL 接続で問題が発生した場合は、現在の信頼ルート CA 証明書を更新する必要があります。詳細については、「SSL 接続用 ACM 証明書への移行」を参照してください。

次のセクションに、接続の問題のサンプルエラーメッセージと考えられる解決策を示します。SQL クライアントツールによってエラーメッセージが異なるため、このリストは完全ではありませんが、問題のトラブルシューティングに適した開始点です。

Amazon EC2 以外からの接続と、ファイアウォールのタイムアウトの問題

COPY コマンドなどの長いクエリを実行すると、データベースへのクライアント接続がハングまたはタイムアウトしているように見えます。この場合、Amazon Redshift コンソールにはクエリが完了

したと表示されますが、クライアントツール自体はまだクエリを実行しているように見えることがあります。接続がいつ停止したかに応じて、クエリの結果がないか、不完全になる可能性があります。

考えられる解決策

この問題は、Amazon EC2 インスタンス以外のマシンから Amazon Redshift に接続するときに発生します。この場合、アイドル状態の接続は、一定期間非アクティブになった後、ファイアウォールなどの中間ネットワークコンポーネントによって終了します。このような動作は、Virtual Private Network (VPN) やローカルネットワークからログインした場合によく発生します。

このようなタイムアウトを回避するために以下の変更を行うことを推奨します。

- クライアントシステムで TCP/IP タイムアウト値を大きく設定します。この変更は、クラスターへの接続に使用しているコンピュータで行います。クライアントやネットワークに対してタイムアウト期間を調整する必要があります。詳細については、「TCP/IP タイムアウト設定を変更する」を参照してください。
- 必要に応じて、DSN レベルでキープアライブの動作を設定します。詳細については、「DSN のタイムアウト設定を変更する」を参照してください。

TCP/IP タイムアウト設定を変更する

TCP/IP タイムアウト設定を変更するには、クラスターへの接続に使用するオペレーティングシステムに応じて、タイムアウト設定を設定します。

- Linux — クライアントが Linux で動作している場合は、次のコマンドをルートユーザーとして実行し、現在のセッションのタイムアウト設定を変更します。

```
/sbin/sysctl -w net.ipv4.tcp_keepalive_time=200 net.ipv4.tcp_keepalive_intvl=200
 net.ipv4.tcp_keepalive_probes=5
```

  設定を保持するには、次の値を使ってファイル `/etc/sysctl.conf` を作成または変更し、システムを再起動します。

```
net.ipv4.tcp_keepalive_time=200
net.ipv4.tcp_keepalive_intvl=200
net.ipv4.tcp_keepalive_probes=5
```

- Windows — クライアントが Windows で動作している場合、HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\ にある次のレジストリ設定の値を編集します。
  - KeepAliveTime: 30000
  - KeepAliveInterval: 1000
  - TcpMaxDataRetransmissions: 10

  これらの設定は DWORD のデータの種類を使用します。これらがレジストリパスに存在しない場合、設定を作成し、これらの推奨値を指定できます。Windows レジストリの編集の詳細については、Windows のドキュメントを参照してください。

  これらの値を設定したら、コンピュータを再起動して変更を有効にします。

- Mac — クライアントが Mac で動作している場合は、次のコマンドを実行して現在のセッションのタイムアウト設定を変更します。

```
sudo sysctl net.inet.tcp.keepintvl=200000
sudo sysctl net.inet.tcp.keepidle=200000
sudo sysctl net.inet.tcp.keepinit=200000
sudo sysctl net.inet.tcp.always_keepalive=1
```

  設定を保持するには、次の値を使ってファイル /etc/sysctl.conf を作成または変更します。

```
net.inet.tcp.keepidle=200000
net.inet.tcp.keepintvl=200000
net.inet.tcp.keepinit=200000
net.inet.tcp.always_keepalive=1
```

  コンピュータを再起動し、次のコマンドを実行して値が設定されていることを確認します。

```
sysctl net.inet.tcp.keepidle
sysctl net.inet.tcp.keepintvl
sysctl net.inet.tcp.keepinit
sysctl net.inet.tcp.always_keepalive
```

DSN のタイムアウト設定を変更する

必要に応じて、DSN レベルでキープアライブの動作を設定できます。これを行うには、odbc.ini ファイルで以下のパラメータを追加または変更します。

KeepAlivesCount

接続が切断されていると見なされる前に失うことが許容される TCP キープアライブパケットの数。

KeepAlivesIdle

ドライバーが TCP キープアライブパケットを送信する前にアイドル状態である秒数。

KeepAlivesInterval

TCP キープアライブを再送信する間隔の秒数。

これらのパラメータがない場合、または値が 0 である場合、システムは指定されている TCP/IP キープアライブパラメータを使用して DSN キープアライブの動作を決定します。Windows では、`HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\` のレジストリに TCP/IP パラメータ があります。Linux および macOS では、TCP/IP のパラメータは sysctl.conf ファイルにあります。

接続が拒否または失敗する

接続が拒否された場合や失敗した場合、次のいずれかと同様のエラーが表示されることがあります。

- "Failed to establish a connection to *<endpoint>*."
- "Could not connect to server: Connection timed out。Is the server running on host *'<endpoint>'* and accepting TCP/IP connections on port *'<port>'*?"
- "Connection refused。Check that the hostname and port are correct and that the postmaster is accepting TCP/IP connections."

考えられる解決策

通常、接続の確立に失敗したことを示すエラーメッセージが表示された場合は、クラスターにアクセスするためのアクセス許可の問題またはクラスターへのトラフィックネットワークの問題を示しています。

クラスターが属するネットワークの外部にあるクライアントツールからクラスターに接続するには、クラスターのセキュリティグループにインバウンドルールを追加します。ルールの設定は、Amazon Redshift クラスターを仮想プライベートクラウド (VPC) で作成しているかどうかによって異なります。

- Amazon VPC に基づく仮想プライベートクラウド (VPC) で Amazon Redshift クラスターを作成した場合は、Amazon VPC で クライアント CIDR/IP アドレスを指定するインバウンドルールを VPC セキュリティグループに追加します。クラスターの VPC セキュリティグループの設定とパブリックにアクセス可能なオプションの詳細については、「VPC での Redshift リソース」を参照してください。
- VPC の外で Amazon Redshift クラスターを作成した場合は、Amazon Redshift のクラスターセキュリティグループにクライアント CIDR/IP アドレスを追加します。クラスターセキュリティグループの設定の詳細については、「Amazon Redshift セキュリティグループ」を参照してください。

Amazon EC2 インスタンスで実行するクライアントツールからクラスターに接続する場合も、インバウンドルールを追加します。この場合は、クラスターセキュリティグループにルールを追加します。ルールでは、クライアントツールの Amazon EC2 インスタンスに関連付けられた Amazon EC2 セキュリティグループを指定する必要があります。

場合によっては、ファイアウォールなど、クライアントとサーバーの間にレイヤーが存在することがあります。このような場合は、ファイアウォールがクラスター用に設定したポート経由のインバウンド接続を受け入れることを確認します。

クライアントおよびドライバーに互換性がない

クライアントとドライバーに互換性がない場合、「指定された DSN にドライバーとアプリケーション間のアーキテクチャの不一致が含まれている」というエラーが表示されることがあります。

考えられる解決策

接続を試みてアーキテクチャの不一致に関するエラーが発生する場合、クライアントツールとドライバーに互換性がないことを意味します。これは、システムのアーキテクチャが一致しないために発生します。例えば、32 ビットクライアントツールがあるが、ドライバーの 64 ビットバージョンをインストールした場合にこれが発生することがあります。64 ビットクライアントツールが 32 ビットドライバーを使用できることはありますが、64 ビットドライバーで 32 ビットアプリケーションを使用することはできません。ドライバーとクライアントツールが同じバージョンのシステムアーキテクチャを使用していることを確認します。

クエリがハングして、クラスターに達しない場合がある

クエリが完了しないという問題に直面します。クエリは実行されますが、SQL クライアント ツールでハングします。クエリは、システムテーブルや Amazon Redshift コンソールなどでクラスターに表示されない場合があります。

考えられる解決策

この問題は、パケットドロップが原因で発生する可能性があります。この場合、2 つのインターネットプロトコル (IP) ホスト間のネットワークパスの最大伝送ユニット (MTU) サイズに差があります。MTU サイズにより、ネットワーク接続を介して 1 つのイーサネットフレームで転送できるパケットの最大サイズ (バイト単位) が決まります。AWS では、一部の Amazon EC2 インスタンスタイプが 1500 MTU (Ethernet v2 フレーム) をサポートしており、その他のインスタンスタイプは 9001 MTU (TCP/IP ジャンボフレーム) をサポートしています。

MTU サイズの違いで発生する問題を防ぐために、次のいずれかを行うことをお勧めします:

- ご使用のクラスターが EC2-VPC プラットフォームを使用している場合、`Destination Unreachable` を返すインバウンドカスタム Internet Control Message Protocol (ICMP) ルールによって Amazon VPC セキュリティグループを設定します。このルールは、送信側ホストがネットワークパスに沿って最低の MTU サイズを使用するように指示します。この方法の詳細については、「セキュリティグループを設定して ICMP の「Destination Unreachable」を許可する」を参照してください。
- ご使用のクラスターが EC2-Classic プラットフォームを使用しているか、ICMP インバウンドルールを許可できない場合、TCP/IP ジャンボフレームを無効にして、Ethernet v2 フレームを使用します。この方法の詳細については、「インスタンスの MTU の設定」を参照してください。

セキュリティグループを設定して ICMP の「Destination Unreachable」を許可する

2 つのホスト間のネットワークで MTU サイズに違いがある場合、ネットワーク設定がパス MTU 検出 (PMTUD) をブロックしないことをまず確認します。PMTUD は、受信側ホストが次の ICMP メッセージで送信側ホストに応答するのを可能にします: `Destination Unreachable: fragmentation needed and DF set (ICMP Type 3, Code 4)` このメッセージは、送信側ホストがネットワークパスに沿って最低の MTU サイズを使用してリクエストを再送信するように指示します。このネゴシエーションがないと、リクエストが大きすぎて受信側ホストが受け取れないため、パケットドロップが発生する可能性があります。この ICMP メッセージの詳細については、インターネット技術標準化委員会 (IETF) のウェブサイトから RFC792 を参照してください。

この ICMP インバウンドルールを Amazon VPC セキュリティグループのために明示的に設定しない場合、PMTUD はブロックされます。AWS では、セキュリティグループは、インバウンドおよびアウトバウンドトラフィックのルールをインスタンスに指定する仮想ファイアウォールです。Amazon Redshift クラスターセキュリティグループの詳細については、Amazon Redshift セキュリティグループを参照してください。EC2-VPC プラットフォームを使用するクラスターでは、Amazon Redshift はクラスターへのトラフィックを許可または拒否するために VPC セキュリティグループを使用します。デフォルトでは、セキュリティグループはロックされており、すべてのインバウンドトラフィックを拒否します。EC2-Classic インスタンスまたは EC2-VPC インスタンスのインバウンドルールとアウトバウンドルールを設定する方法については、「Amazon EC2 ユーザーガイド」の「EC2-Classic と VPC 内のインスタンスの違い」を参照してください。

VPC セキュリティグループにルールを追加する方法については、「VPC セキュリティグループ」を参照してください。このルールに必要な特定の PMTUD 設定の詳細については、「Amazon EC2 ユーザーガイド」の「パス MTU 検出」を参照してください。

インスタンスの MTU の設定

クラスターで EC2-Classic プラットフォームが使用されている場合や、インバウンドトラフィックにカスタム ICMP ルールを許可できない場合があります。このような場合、Amazon Redshift クラスターに接続する EC2 インスタンスのネットワークインターフェイス (NIC) で MTU を 1500 に調整することをお勧めします。この調整によって TCP/IP ジャンボフレームが無効になるため、同じパケットサイズが一貫して接続に使用されるようになります。ただし、このオプションでは、Amazon Redshift への接続だけでなく、インスタンス全体の最大ネットワークスループットを減らすことに注意してください。詳細については、次の手順を参照してください。

Microsoft Windows のオペレーティング システムで MTU を設定する

クライアントが Microsoft Windows オペレーティングシステムで動作している場合、netshコマンドを使用してイーサネットアダプターの MTU 値を確認し、設定できます。

1. 現在の MTU 値を調べるには、次のコマンドを実行します。

```
netsh interface ipv4 show subinterfaces
```

2. 出力で MTU アダプタの Ethernet の値を確認します。
3. 値が 1500 ではない場合、次のコマンドを実行して設定します。

```
netsh interface ipv4 set subinterface "Ethernet" mtu=1500 store=persistent
```

この値を設定したら、コンピュータを再起動して変更を有効にします。

Linux オペレーティング システムで MTU を設定する

クライアントが Linux オペレーティング システムで動作している場合、ipコマンドを使用して MTU 値を確認し、設定できます。

1. 現在の MTU 値を調べるには、次のコマンドを実行します。

```
$ ip link show eth0
```

2. 出力で次の mtu の値を確認します。
3. 値が 1500 ではない場合、次のコマンドを実行して設定します。

```
$ sudo ip link set dev eth0 mtu 1500
```

Mac オペレーティング システムで MTU を設定する

- How to change the MTU for troubleshooting purposes については macOS サポートサイトの指示に従ってください。詳細については、サポートサイトを検索してください。

JDBC フェッチサイズパラメータの設定

デフォルトでは、Redshift JDBC ドライバーはリングバッファを使用してメモリを効率的に管理し、メモリ不足エラーを防止します。フェッチサイズパラメータは、リングバッファが明示的に無効になっている場合にのみ適用できます。詳細については、「link」を参照してください。この設定では、各バッチで取得される行数を制御するようにフェッチサイズを設定する必要があります。

フェッチサイズを使用するタイミング

次の場合は、フェッチサイズパラメータを使用します。

- 行ベースのバッチ処理をきめ細かく制御する必要がある
- 従来のフェッチサイズ動作を必要とするレガシーアプリケーションを使用している

フェッチサイズの設定

リングバッファが無効な場合、デフォルトでは、JDBC ドライバーはクエリに対して一度にすべての結果を収集します。大きな結果セットを返すクエリは、過剰なメモリを消費する可能性があります。一度にすべてではなくバッチで結果セットを取得するには、アプリケーションで JDBC フェッチサイズパラメータを設定します。

> **Note**
>
> フェッチサイズは ODBC ではサポートされません。

最適なパフォーマンスのためには、メモリ不足エラーが発生しない最大の値にフェッチサイズを設定します。フェッチサイズの値を低く設定すると、サーバートリップが増え、それにより実行時間が長くなります。サーバーは、クライアントが結果セット全体を取得するまで、WLM クエリスロットおよび関連メモリを含むリソースを予約します。そうでない場合、クエリはキャンセルされます。フェッチサイズを適切に調整すると、それらのリソースはより迅速に解放され、他のクエリに利用できるようになります。

> **Note**
>
> 大きなデータセットを抽出する必要がある場合は、UNLOAD ステートメントを使用してデータを Amazon S3 に転送することをお勧めします。UNLOAD を使用するときは、コンピューティングノードは並行してデータの転送を高速化します。

JDBC フェッチサイズパラメータの詳細については、PostgreSQL のドキュメントで「Getting results based on a cursor」を参照してください。

Amazon Redshift Data API の使用

Amazon Redshift Data API は、データベースドライバー、接続、ネットワーク設定、データバッファリング、認証情報などを管理する必要がなくなるため、Amazon Redshift データウェアハウスへのアクセスを簡素化します。AWS SDK で Data API オペレーションを使用して SQL ステートメントを実行できます。Data API オペレーションの詳細については、Amazon Redshift Data API のリファレンスを参照してください。

Data API は、データベースへの永続的な接続を必要としません。代わりに、セキュア HTTP エンドポイントおよび AWS SDK との統合を利用できます。エンドポイントを使用して、接続を管理せず

に SQL ステートメントを実行することができます。Data API の呼び出しは非同期です。Data API では、AWS Secrets Manager に格納された認証情報か、一時的なデータベース認証情報を使用できます。どちらの認可方法でも、API 呼び出しでパスワードを渡す必要はありません。AWS Secrets Manager の詳細については、AWS Secrets Managerユーザーガイドの「AWS Secrets Manager とは」を参照してください。認可に AWS IAM アイデンティティセンター を使用することもできます。

Data API を使用すると、AWS Lambda、Amazon SageMaker AI ノートブック、AWS Cloud9 などのウェブサービスベースのアプリケーションで、Amazon Redshift のデータにプログラムによってアクセスできます。これらのアプリケーションの詳細については、「AWS Lambda」、「Amazon SageMaker AI」、「AWS Cloud9」を参照してください。

Data API の詳細については、「AWS Big Data ブログ」の「Get started with the Amazon Redshift Data API」を参照してください。

Amazon Redshift Data API の操作

Amazon Redshift データ API を使用する前に、以下の手順を確認してください。

1. データ API の呼び出し元として承認されているかどうかを確認します。認可の詳細については、「Amazon Redshift Data API へのアクセスの認可」を参照してください。
2. Secrets Manager から認証情報を使用して Data API を呼び出すか、一時的な認証情報を使用するか、または AWS IAM アイデンティティセンター を使用するかを決定します。詳細については、「Amazon Redshift Data API を呼び出すときのデータベース認証用の認証情報の選択」を参照してください。
3. 認証情報に Secrets Manager を使用する場合は、シークレットを設定します。詳細については、「AWS Secrets Manager へのデータベース認証情報の保存」を参照してください。
4. Data API を呼び出す際の考慮事項と制限事項を確認してください。詳細については、「Amazon Redshift Data API を呼び出す際の考慮事項」を参照してください。
5. Data API は、AWS Command Line Interface(AWS CLI) や独自のコードから、または Amazon Redshift コンソールのクエリエディタを使用して呼び出します。AWS CLI からの呼び出しの例については、「Data API の呼び出し」を参照してください。

Amazon Redshift Data API を呼び出す際の考慮事項

Data API を呼び出すときは、以下について検討します。

- Amazon Redshift Data API は、Amazon Redshift のプロビジョニング済みクラスターと Redshift Serverless ワークグループのデータベースにアクセスできます。Redshift Data API を利用可能な AWS リージョン の一覧については、Amazon Web Services 全般のリファレンスの「Redshift Data API」のエンドポイントのリストをご覧ください。
- クエリの最大期間は 24 時間です。
- アクティブなクエリの最大数 (STARTED および SUBMITTED クエリ) は、Amazon Redshift クラスターあたり 500 です。
- クエリ結果の最大サイズは 500 MB (gzip 圧縮後) です。500 MB を超えるレスポンスデータが返されると、その呼び出しは終了します。
- クエリ結果の最大保持時間は 24 時間です。
- クエリステートメントの最大サイズは 200 KB です。
- Data API は、すべての Serverless ワークグループ、および次のノードタイプのプロビジョニング済み単一ノードおよび複数ノードのクラスターをクエリするために使用できます。
  - dc2.large
  - dc2.8xlarge
  - rg.large
  - rg.xlarge
  - rg.4xlarge
  - rg.12xlarge
  - ra3.large
  - ra3.xlplus
  - ra3.4xlarge
  - ra3.16xlarge
- クラスターは、Amazon VPC サービスに基づいて Virtual Private Cloud (VPC) で作成する必要があります。
- デフォルトでは、ExecuteStatement または BatchExecuteStatement API オペレーションの実行者と同じ IAM ロールを持つユーザーは CancelStatement、DescribeStatement、GetStatementResult、GetStatementResultV2、および ListStatements API オペレーションで同じステートメントを操作できます。別のユーザーから同じ SQL ステートメントを操作する場合、そのユーザーは、SQL ステートメントを実行したユーザーの IAM ロールを引き受ける必要があります。ロールを割り当てる方法については、Amazon Redshift Data API へのアクセスの認可を参照してください。

- BatchExecuteStatement API オペレーションの Sqls パラメータの SQL ステートメントは、配列の順序で連続して実行されます。後続の SQL ステートメントは、前のステートメントが完了するまで開始されません。デフォルトでは、すべての SQL ステートメントは単一のトランザクションとして実行されます。いずれかの SQL ステートメントが失敗すると、すべての処理がロールバックされます。ExecutionMode パラメータを使用して、トランザクションの動作を制御します。
  - TRANSACTION (デフォルト) – サービスはすべての SQL ステートメントを単一のトランザクションとして実行し、すべてを一括してコミットまたはロールバックします。
  - AUTO_COMMIT – 各 SQL ステートメントは個別にコミットされます。あるステートメントの失敗は、他のステートメントには影響しません。

  例については、複数の SQL ステートメントを実行するを参照してください。
- ExecuteStatement または BatchExecuteStatement API オペレーションで使用されるクライアントトークンの最大保持時間は 8 時間です。
- オプションの WaitTimeSeconds パラメータを使用すると、オペレーションが新しいクエリを送信するとき、または進行中のクエリに対して、Data API が最大 30 秒待機します。これにより、ポーリングリクエストの数が減少し、ラウンドトリップが少なくなり、レイテンシーが低くなります。詳細については、「ロングポーリングによる API コールの削減」を参照してください。
- Amazon Redshift でプロビジョニングされたクラスターと Redshift Serverless ワークグループがカスタマーマネージドキーを使用して暗号化されている場合、Redshift は Redshift Data API がオペレーションにキーを使用できるようにする許可を作成します。詳細については、 Amazon Redshift Data API で AWS KMS を使用する を参照してください。
- Redshift Data API の各 API には、リクエストのスロットリング前の 1 秒あたりのトランザクション割り当てがあります。クォータについては、「Amazon Redshift Data API のクォータ」を参照してください。リクエスト率がクォータを超えると、HTTP ステータスコード: 400 の ThrottlingException が返されます。スロットリングに対応するには、「AWS SDK とツールリファレンスガイド」の「再試行動作」で説明されている再試行戦略を使用します。AWS SDK によっては、この戦略が HTTP 400 エラー用に自動的に実装されています。

> **Note**
>
> AWS Step Functions では、再試行がデフォルトでは有効になっていません。Step Functions ステートマシンで Redshift データ API を呼び出す場合は、Redshift データ API コールに ClientToken 冪等性パラメータを含める必要があります。この ClientToken の値は、再試行の間も維持する必要があります。次の ExecuteStatement API へのリクエストの例では、States.ArrayGetItem(States.StringSplit($

$.Execution.Id, ':'), 7) 式は、組み込み関数を使用してステートマシンが実行されるたびに一意となる $$.Execution.Id の UUID 部分を抽出します。詳細については、「AWS Step Functions デベロッパーガイド」の「組み込み関数」を参照してください。

```
{
  "Database": "dev",
  "Sql": "select 1;",
  "ClusterIdentifier": "MyCluster",
  "ClientToken.$": "States.ArrayGetItem(States.StringSplit($$.Execution.Id,
 ':'), 7)"
}
```

Amazon Redshift Data API を呼び出すときのデータベース認証用の認証情報の選択

データ API を呼び出すと、一部の API 操作で次のいずれかの認証方法を使用します。各メソッドでは、異なるパラメータの組み合わせが必要です。

AWS IAM アイデンティティセンター

Data API は、AWS IAM アイデンティティセンター に登録されたシングルサインオンユーザーでアクセスできます。IAM アイデンティティセンターの設定手順については、「信頼できる ID の伝播で Data API を使用する」を参照してください。

AWS Secrets Manager

この方法では、username および password を持つ AWS Secrets Manager が格納されるシークレットの secret-arn を指定します。指定されたシークレットには、指定する database に接続するための認証情報が含まれます。クラスターに接続するときは、データベース名も指定します。クラスター識別子 (dbClusterIdentifier) を指定する場合は、シークレット内に格納されているクラスター識別子と一致する必要があります。サーバーレスワークグループに接続する場合は、データベース名も指定します。詳細については、「AWS Secrets Manager へのデータベース認証情報の保存」を参照してください。

またこの方法では、データが配置されている AWS リージョン を特定する region 値も指定できます。

一時的な認証情報

この方法を実行する場合は、次のいずれかのオプションを選択します。

- サーバーレスワークグループに接続する場合は、ワークグループ名とデータベース名を指定します。データベースユーザー名は IAM ID から取得されます。例えば、arn:iam::123456789012:user:foo のデータベースユーザー名は IAM:foo です。また、redshift-serverless:GetCredentials オペレーションを呼び出す許可も必要です。
- IAM ID としてクラスターに接続するときは、クラスター識別子とデータベース名を指定します。データベースユーザー名は IAM ID から取得されます。例えば、arn:iam::123456789012:user:foo のデータベースユーザー名は IAM:foo です。また、redshift:GetClusterCredentialsWithIAM オペレーションを呼び出す許可も必要です。
- データベースユーザーとしてクラスターに接続するときは、クラスター識別子、データベース名、データベースユーザー名を指定します。また、redshift:GetClusterCredentials オペレーションを呼び出す許可も必要です。この方法で接続するときにデータベースグループに参加する方法については、「クラスターへの接続時にデータベースグループに参加する」を参照してください。

またこの方法では、データが配置されている AWS リージョン を特定する region 値も指定できます。

Amazon Redshift データ API を呼び出すときの JDBC データ型のマッピング

次の表は、Data API 呼び出しで指定したデータ型に Java Database Connectivity (JDBC) データ型をマッピングしたものです。

JDBC データ型	Data API のデータ型
INTEGER, SMALLINT, BIGINT	LONG
FLOAT, REAL, DOUBLE	DOUBLE
DECIMAL	STRING
BOOLEAN, BIT	BOOLEAN

JDBC データ型	Data API のデータ型
BLOB, BINARY, LONGVARBINARY	BLOB
VARBINARY	STRING
CLOB	STRING
その他の型 (日時に関する型も含む)	STRING

文字列値は Amazon Redshift データベースに渡され、暗黙的にデータベースのデータ型に変換されます。

> **Note**
>
> 現在、Data API はユニバーサル固有識別子 (UUID) の配列をサポートしていません。

Amazon Redshift Data API を呼び出す際にパラメータを使用した SQL ステートメントを実行する

SQL ステートメントの一部にパラメータを使用して Data API オペレーションを呼び出し、データベースエンジンに送信される SQL テキストを制御できます。名前付きパラメータを使用すると、SQL テキストでハードコーディングすることなく、柔軟な方法でパラメータを渡すことができます。これらは、SQL テキストを再利用し、SQL インジェクションの問題を回避するのに役立ちます。

次の例は、AWS CLI コマンド `execute-statement` または `batch-execute-statement` の `parameters` フィールドの名前付きパラメータを示しています。

```
--parameters "[{\"name\": \"id\", \"value\": \"1\"},{\"name\": \"address\", \"value\":
 \"Seattle\"}]"
```

名前付きパラメータを使用する際は、次について検討します。

- 名前付きパラメータは、SQL ステートメントの値の置換にのみ使用できます。
  - `INSERT INTO mytable VALUES(:val1)` など、INSERT ステートメントの値は置換することができます。

名前付きパラメータは任意の順序で指定でき、パラメータは SQL テキストで複数回使用できます。前の例で示したパラメータオプションでは、値 1 および Seattle がテーブル列 id および address に挿入されます。SQL テキストでは、次のように名前付きパラメータを指定します。

```
--sql "insert into mytable values (:id, :address)"
```

- WHERE attr >= :val1、WHERE attr BETWEEN :val1 AND :val2、HAVING COUNT(attr) > :val など、条件句の値は置換することができます。
- SELECT column-name、ORDER BY column-name、または GROUP BY column-name など、SQL ステートメントの列名は置換できません。

  例えば、次の SELECT ステートメントは無効な構文なので失敗します。

  ```
  --sql "SELECT :colname, FROM event" --parameters "[{\"name\": \"colname\", \"value\": \"eventname\"}]"
  ```

  構文エラーのあるステートメントを記述 (describe-statement オペレーション) した場合、返される QueryString はパラメータ ("QueryString": "SELECT :colname, FROM event") の列名を置換せず、エラーが報告されます (ERROR: syntax error at or near \"FROM \"\n Position: 12)。
- COUNT(column-name)、AVG(column-name)、または SUM(column-name) などの集計関数では、列名を置換できません。
- JOIN 句の列名は置換できません。

- SQL が実行されると、データは暗黙的にデータ型にキャストされます。データ型のキャストのついての詳細は、Amazon Redshift データベースデベロッパーガイドの「データ型」を参照してください。
- 値を NULL に設定することはできません。データ API では、これはリテラル文字列 NULL として解釈されます。次の例では、idがリテラル文字列 null に置き換えられます。SQL NULL 値ではありません。

  ```
  --parameters "[{\"name\": \"id\", \"value\": \"null\"}]"
  ```
- 長さにゼロの値を設定することはできません。Data API の SQL ステートメントが失敗します。次の例では、idの長さをゼロの値に設定しようとしているため、SQL ステートメントは失敗します。

```
--parameters "[{\"name\": \"id\", \"value\": \"\"}]"
```

- パラメータを使用して、SQL ステートメントにテーブル名を設定することはできません。Data API では、JDBC PreparedStatement のルールに従います。
- describe-statement オペレーションの出力により、SQL ステートメントのクエリパラメータが返されます。
- execute-statement オペレーションと batch-execute-statement オペレーションは、どちらもパラメータを使用した SQL ステートメントをサポートしています。batch-execute-statement を使用する場合、パラメータはバッチ内のすべての SQL ステートメント間で共有されます。各 SQL ステートメントは、指定されたパラメータのサブセットを参照できますが、どのパラメータも最低 1 回はどこかの SQL ステートメントで使用する必要があります。

Amazon Redshift Data API を呼び出す際に冪等性トークンで SQL ステートメントを実行する

変異する API リクエストを行うと、通常、リクエストはオペレーションの非同期ワークフローが完了する前に結果を返します。リクエストが既に結果を返している場合でも、操作が完了する前にタイムアウトしたり、その他のサーバーの問題が発生したりすることもあります。これにより、リクエストが成功したかどうかを判断するのが難しくなり、操作を正常に完了するために複数回の再試行が行われることがあります。ただし、元のリクエストとその後の再試行が成功すると、操作は複数回完了します。つまり、意図したよりも多くのリソースを更新する可能性があります。

冪等性とは、API リクエストが 1 回だけ完了することを保証するものです。冪等性リクエストでは、元のリクエストが正常に完了した場合、その後の再試行は追加のアクションを実行せずに正しく完了します。データ API ExecuteStatement と BatchExecuteStatement オペレーションには、オプションの ClientToken 冪等性パラメータがあります。ClientToken は 8 時間後に期限切れになります。

> **Important**
>
> AWS SDK から ExecuteStatement および BatchExecuteStatement オペレーションを呼び出すと、再試行時に使用するクライアントトークンが自動的に生成されます。この場合、ExecuteStatement および BatchExecuteStatement オペレーションで client-token パラメータを使用することはお勧めしません。CloudTrail の口グを表示す

ると ClientToken を確認できます。CloudTrail ログの例については、「Amazon Redshift Data API の例」を参照してください。

次の execute-statement AWS CLI コマンドは、冪等性のオプション client-token パラメータを示しています。

```
aws redshift-data execute-statement
    --secret-arn arn:aws:secretsmanager:us-west-2:123456789012:secret:myuser-secret-
hKgPWn
    --cluster-identifier mycluster-test
    --sql "select * from stl_query limit 1"
    --database dev
    --client-token b855dced-259b-444c-bc7b-d3e8e33f94g1
```

次の表は、冪等性 API リクエストに対して返される一般的な応答と、再試行の推奨事項を示しています。

応答	推奨事項	コメント
200 (OK)	再試行しないでください	元のリクエストは正しく完了しています。それ以降に再試行しても正常に戻ります。
400 シリーズ応答コード	再試行しないでください	次のうち、リクエストに問題があります。 • 無効なパラメータまたはパラメータの組み合わせが含まれています。 • 権限のないアクションまたはリソースを使用しています。 • 状態の変更処理中のリソースを使用しています。 リクエストに状態の変更処理中のリソースが含まれている場合、リクエストを再試行すると成功する可能性があります。

応答	推奨事項	コメント
500 シリーズ応答コード	再試行	このエラーは AWS サーバー側の問題によって発生し、通常は一時的なものです。適切なバックオフ戦略でリクエストを繰り返してください。

Amazon Redshift 応答コードの詳細については、「Amazon Redshift API リファレンス」から「一般的なエラー」を参照してください。

Amazon Redshift Data API を呼び出す際にセッションの再利用によって SQL ステートメントを実行する

API リクエストを実行して SQL ステートメントを実行すると、通常、SQL が実行されているセッションは SQL の完了時に終了します。指定された秒数だけセッションをアクティブにするために、Data API `ExecuteStatement` および `BatchExecuteStatement` オペレーションにはオプションの `SessionKeepAliveSeconds` パラメータがあります。`SessionId` レスポンスフィールドにはセッションのアイデンティティが含まれており、後続の `ExecuteStatement` および `BatchExecuteStatement` オペレーションで使用できます。それ以降の呼び出しでは、別の `SessionKeepAliveSeconds` を指定してアイドルタイムアウト時間を変更できます。`SessionKeepAliveSeconds` が変更されていない場合、最初のアイドルタイムアウト設定は残ります。セッションの再利用を使用する場合は、次の点を考慮してください。

- `SessionKeepAliveSeconds` の最大値は 24 時間です。
- セッションは最大 24 時間持続します。24 時間後、セッションは強制的に閉じられ、進行中のクエリは終了します。
- Amazon Redshift クラスターまたは Redshift Serverless ワークグループあたりのセッションの最大数は 500 です。
- セッションごとに 1 つのクエリしか実行できません。同じセッションで次のクエリを実行するには、クエリが完了するまで待つ必要があります。つまり、指定されたセッションでクエリを並行して実行することはできません。
- Data API は、特定のセッションのクエリをキューに入れることはできません。

`ExecuteStatement` および `BatchExecuteStatement` オペレーションの呼び出しで使用される `SessionId` を取得するには、`DescribeStatement` および `ListStatements` オペレーションを

呼び出します。セッションとそのステータスを一覧表示するには、ListSessions オペレーションを使用します。詳細については、「セッションを一覧表示する」を参照してください。

次の例は、SessionKeepAliveSeconds および SessionId パラメータを使用してセッションを存続させ、再利用する方法を示しています。まず、オプションの session-keep-alive-seconds パラメータを 2 に設定して execute-statement AWS CLI コマンドを呼び出します。

```
aws redshift-data execute-statement
    --session-keep-alive-seconds 2
    --sql "select 1"
    --database dev
    --workgroup-name mywg
```

応答にはセッションの識別子が含まれます。

```
{
    "WorkgroupName": "mywg",
    "CreatedAt": 1703022996.436,
    "Database": "dev",
    "DbUser": "awsuser",
    "Id": "07c5ffea-76d6-4786-b62c-4fe3ef529680",
    "SessionId": "5a254dc6-4fc2-4203-87a8-551155432ee4"
}
```

次に、最初の呼び出しから返された SessionId を使用して execute-statement AWS CLI コマンドを呼び出します。オプションで、session-keep-alive-seconds パラメータを 10 に設定してアイドルタイムアウト値を変更します。

```
aws redshift-data execute-statement
    --sql "select 1"
    --session-id 5a254dc6-4fc2-4203-87a8-551155432ee4
    --session-keep-alive-seconds 10
```

SQL ステートメントの結果を取得する

結果の形式に応じて異なる Data API オペレーションを使用して、SQL 結果を取得します。ExecuteStatement および BatchExecuteStatement オペレーションを呼び出すときに、結果の形式を JSON と CSV のどちらにするかを指定できます。未指定の場合は、デフォルトで

JSON になります。JSON 形式で結果を取得するには、GetStatementResult オペレーションを使用します。CSV 形式で結果を取得するには、GetStatementResultV2 オペレーションを使用します。

JSON 形式で返される結果は、各列に関するメタデータを含むレコードです。各レコードは JSON 形式です。例えば、GetStatementResult からのレスポンスは次のようになります。

```
{
   "ColumnMetadata": [
      {
         "isCaseSensitive": false,
         "isCurrency": false,
         "isSigned": true,
         "label": "?column?",
         "name": "?column?",
         "nullable": 1,
         "precision": 10,
         "scale": 0,
         "schemaName": "",
         "tableName": "",
         "typeName": "int4",
         "length": 0
      }
   ],
   "NextToken": "<token>",
   "Records": [
        [
            {
                "longValue": 1
            }
        ]
    ],
   "TotalNumRows": <number>
}
```

CSV 形式で返される結果は、各列に関するメタデータを含むレコードです。結果は 1 MB のチャンクで返され、各チャンクには任意の数の行が CSV 形式で含まれます。各リクエストは、最大 15 MB 分の結果を返します。結果が 15 MB を上回る場合は、結果の取得を続行できるように、次のページトークンが返されます。例えば、GetStatementResultV2 からのレスポンスは次のようになります。

```
{
```

```
    "ColumnMetadata": [
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": true,
            "label": "?column?",
            "name": "?column?",
            "nullable": 1,
            "precision": 10,
            "scale": 0,
            "schemaName": "",
            "tableName": "",
            "typeName": "int4",
            "length": 0
        },
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": true,
            "label": "?column?",
            "name": "?column?",
            "nullable": 1,
            "precision": 10,
            "scale": 0,
            "schemaName": "",
            "tableName": "",
            "typeName": "int4",
            "length": 0
        },
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": true,
            "label": "?column?",
            "name": "?column?",
            "nullable": 1,
            "precision": 10,
            "scale": 0,
            "schemaName": "",
            "tableName": "",
            "typeName": "int4",
            "length": 0
        }
    ],
```

```
    "NextToken": "<token>",
    "Records": [
        [
            {
                "CSVRecords":"1,2,3\r\n4,5,6\r\n7,8,9\r\n, ... 1MB" // First 1MB Chunk
            },
            {
                "CSVRecords":"1025,1026,1027\r\n1028,1029,1030\r\n...2MB" // Second 1MB
 chunk
            }
            ...
        ]
    ],
    "ResultFormat" : "CSV",
    "TotalNumRows": <number>
}
```

Amazon Redshift Data API へのアクセスの認可

ユーザーは Data API へのアクセスが許可されている必要があります。事前定義済みの AWS Identity and Access Management (IAM) ポリシーである管理ポリシーをユーザーに追加して、ユーザーに Data API へのアクセスを許可できます。ベストプラクティスとして、アクセス許可ポリシーを IAM ロールにアタッチし、それを必要に応じてユーザーやグループに割り当てることをお勧めします。詳細については、「Amazon Redshift での Identity and Access Management」を参照してください。マネージドポリシーによって許可および拒否されるアクセス許可を確認するには、IAM コンソール (https://console.aws.amazon.com/iam/) 参照してください。

IAM 許可の設定

Amazon Redshift は、`AmazonRedshiftDataFullAccess`マネージドポリシーを提供します。このポリシーは、Amazon Redshift Data API オペレーションへのフルアクセスを提供します。このポリシーでは、Amazon Redshift クラスターまたは Redshift Serverless ワークグループの認証とアクセスに必要な特定の Amazon Redshift、AWS Secrets Manager、IAM API オペレーションへのスコープ付きアクセスも許可されます。

特定のリソースへのアクセスを許可する独自の IAM ポリシーを作成することもできます。ポリシーを作成するには、`AmazonRedshiftDataFullAccess`ポリシーを開始テンプレートとして使用します。作成したポリシーは、Data API にアクセスする必要がある各ユーザーに追加します。

ユーザーに関連付けられた IAM ポリシーの以下の要件を考慮してください。

- AWS Secrets Manager を使用して認証する場合、ポリシーでは secretsmanager:GetSecretValue アクションの使用が許可され、キー RedshiftDataFullAccess でタグ付けされたシークレットを取得できることを確認します。
- 一時的な認証情報を使用してクラスターに対して認証する場合、ポリシーでは、クラスター内の任意のデータベースのデータベースユーザー名 redshift_data_api_user に対する redshift:GetClusterCredentials アクションの使用が許可されることを確認します。このユーザー名は、データベースにすでに作成されている必要があります。
- 一時的な認証情報を使用してサーバーレスワークグループへの認証を行う場合は、ポリシーで redshift-serverless:GetCredentials キーでタグ付けされたワークグループを取得するための RedshiftDataFullAccess アクションの使用が許可されていることを確認します。データベースユーザーが、ソース AWS Identity and Access Management (IAM) ID に対して 1:1 にマップされています。例えば、ユーザー sample_user はデータベースユーザー IAM:sample_user にマップされ、IAM ロール sample_role は IAMR:sample_role にマップされます。IAM ID の詳細については、「IAM ユーザーガイド」の「IAM ID (ユーザー、ユーザーグループ、ロール)」を参照してください。
- IAM アクション redshift-data:GetStatementResult は、GetStatementResult および GetStatementResultV2 API オペレーションの両方へのアクセスを許可します。

次のリンクから、IAM ユーザーガイドで AWS Identity and Access Management についての詳細情報をご覧いただけます。

- IAM ロールの作成については、「IAM ロールの作成」を参照してください。
- IAM ポリシーの作成については、「IAM ポリシーの作成」を参照してください。
- IAM ポリシーをユーザーに追加する方法については、「IAM ID のアクセス許可の追加および削除」を参照してください。

別のアカウントが所有するクラスターでクエリを実行する

別のアカウントが所有するクラスターでクエリを実行するには、呼び出すアカウントで Data API がロールを引き受けることができる IAM ロールを所有アカウントより提供してもらう必要があります。例えば、アカウント A がアクセスする必要のあるクラスターを、アカウント B が所有しているとします。アカウント B は、AWS マネージドポリシーの AmazonRedshiftDataFullAccess をアカウント B の IAM ロールにアタッチできます。次に、アカウント B は、次のような信頼ポリシーを使用してアカウント A を信頼します:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:role/someRoleA"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

最後に、アカウント A の IAM ロールでアカウント B の IAM ロールを引き受けることができなければなりません。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "sts:AssumeRole",
        "Resource": "arn:aws:iam::111122223333:role/someRoleB"
    }
}
```

AWS アカウント 内のリソースを Amazon Redshift Serverless ワークグループおよび Amazon Redshift クラスターに制限する IAM ロールを指定します。

アイデンティティベースのポリシーでリソース ARN を指定して、AWS アカウントの Redshift Serverless ワークグループと Amazon Redshift クラスターへのアクセスを制御できます。この例では、指定した AWS アカウントのワークグループとクラスターのみに対して Data API へのアクセスを許可するポリシーを作成する方法を示します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "redshift-data:CancelStatement",
                "redshift-data:DescribeStatement",
                "redshift-data:GetStatementResult",
                "redshift-data:ListStatements"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "redshift-data:*",
            "Resource": [
                "arn:aws:redshift:us-east-1:111122223333:workgroup/*",
                "arn:aws:redshift:us-east-1:111122223333:cluster:*"
            ]
        }
    ]
}
```

SQL ステートメント情報へのアクセスをステートメント所有者のみに制限する IAM ポリシーを設定する

デフォルトでは、Amazon Redshift Data API は、`ExecuteStatement` および `BatchExecuteStatement` を呼び出すときに使用される IAM ロールを SQL ステートメントの所有者として扱います。ロールを引き受けることが許可されているユーザーは、結果を含む SQL ステートメントに関する情報にアクセスできます。特定の所有者との IAM ロールセッションへの SQL ステートメント情報アクセスを制限するには、条件 `redshift-data:statement-owner-iam-userid: "${aws:userid}"` を追加します。次の IAM ポリシーは、アクセスを制限します。

JSON

```
{
```

```
"Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "redshift-data:CancelStatement",
                "redshift-data:DescribeStatement",
                "redshift-data:GetStatementResult",
                "redshift-data:ListStatements"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "redshift-data:statement-owner-iam-userid": "${aws:userid}"
                }
            }
        }
    ]
}
```

statement-owner-iam-userid 条件は CancelStatement、DescribeStatement、GetStatementResult、および ListStatements で使用することができます。詳細については、「Amazon Redshift Data API で定義されるアクション」を参照してください。

SQL 結果へのアクセスをセッション所有者のみに制限する IAM ポリシーを設定する

デフォルトでは、Amazon Redshift Data API は、ExecuteStatement と BatchExecuteStatement を呼び出すときに使用される IAM ロールを、SQL ステートメントを実行するデータベースセッションの所有者として扱います。ロールを引き受けることが許可されているユーザーは、データベースセッションにクエリを送信できます。特定の所有者との IAM ロールセッションへのセッションアクセスを制限するには、条件 redshift-data:session-owner-iam-userid: "${aws:userid}" を追加します。次の IAM ポリシーは、アクセスを制限します。

次の IAM ポリシーでは、セッション所有者のみがステートメントの結果を取得できます。条件 session-owner-iam-userid は、指定された userid へのリソースアクセスを制限するために使用されます。

JSON

```
{
"Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "redshift-data:ExecuteStatement",
                "redshift-data:BatchExecuteStatement"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "redshift-data:session-owner-iam-userid": "${aws:userid}"
                }
            }
        }
    ]
}
```

`ExecuteStatement` および `BatchExecuteStatement` で `session-owner-iam-userid` 条件を使用することができます。詳細については、「Amazon Redshift Data API で定義されるアクション」を参照してください。

AWS Secrets Manager へのデータベース認証情報の保存

Data API を呼び出すと、AWS Secrets Manager のシークレットを使用してクラスターまたはサーバーレスワークグループの認証情報を渡すことができます。この方法で認証情報を渡すには、シークレットの名前またはシークレットの Amazon リソースネーム (ARN) を指定します。

Secrets Manager で認証情報を保存するには、`SecretManagerReadWrite`マネージドポリシー権限が必要です。最小のアクセス許可についての詳細は、AWS Secrets Managerユーザーガイドの「Creating and Managing Secrets with AWS Secrets Manager」を参照してください。

Amazon Redshift クラスターのシークレットに認証情報を保存するには

1. AWS Secrets Manager コンソールを使用して、クラスターの認証情報が含まれるシークレットを作成します。

- [Store a new secret (新しいシークレットを保存する)] を選択する場合は、[Credentials for Redshift cluster (Redshift クラスターの認証情報)]を選択します。
- [User name] (ユーザー名) (データベースユーザー)、[Password] (パスワード)、および [DB cluster] (DB クラスター) (クラスター識別子) の値をシークレットに保存します。
- キー `RedshiftDataFullAccess` でシークレットにタグを付けます。AWS マネージドポリシーの `AmazonRedshiftDataFullAccess` は、キー `RedshiftDataFullAccess` でタグ付けされたシークレットに対してのみアクション `secretsmanager:GetSecretValue` を許可します。

手順については、AWS Secrets Managerユーザーガイド の「Creating a Basic Secret」を参照してください。

2. AWS Secrets Manager コンソールを使用して、作成したシークレットの詳細を表示するか、`aws secretsmanager describe-secret`の AWS CLI コマンドを実行します。

   シークレットの名前と ARN を書き留めます。これらは、Data API への呼び出しで使用できます。

サーバーレスワークグループのシークレットに認証情報を保存するには

1. サーバーレスワークグループの認証情報を含むシークレットを保存するには、AWS Secrets Manager AWS CLI コマンドを使用します。

   - ファイルにシークレットを作成します。例: `mycreds.json` という名前の JSON ファイル。ファイルで [ユーザー名] (データベースユーザー) と [パスワード] の値を指定します。

     ```
     {
           "username": "myusername",
           "password": "mypassword"
     }
     ```

   - 値をシークレットに保存し、シークレットに `RedshiftDataFullAccess` キーでタグを付けます。

     ```
     aws secretsmanager create-secret --name MyRedshiftSecret  --tags
      Key="RedshiftDataFullAccess",Value="serverless" --secret-string file://
     mycreds.json
     ```

出力は以下のようになります。

```
{
    "ARN":
 "arn:aws:secretsmanager:region:accountId:secret:MyRedshiftSecret-mvLHxf",
    "Name": "MyRedshiftSecret",
    "VersionId": "a1603925-e8ea-4739-9ae9-e509eEXAMPLE"
}
```

詳細については、AWS Secrets Manager ユーザーガイドの「AWS CLI を使用したベーシックシークレットの作成」を参照してください。

2. AWS Secrets Manager コンソールを使用して、作成したシークレットの詳細を表示するか、`aws secretsmanager describe-secret`の AWS CLI コマンドを実行します。

   シークレットの名前と ARN を書き留めます。これらは、Data API への呼び出しで使用できます。

Data API に Amazon VPC エンドポイント (AWS PrivateLink) を作成する

Amazon Virtual Private Cloud (Amazon VPC) を使用すると、Amazon Redshift クラスターやアプリケーションなどの AWS リソースを仮想プライベートクラウド (VPC) 内に起動できます。AWS PrivateLink は、仮想プライベートクラウド (VPC) と AWS サービス間のプライベートな接続を Amazon ネットワーク上で安全に提供します。AWS PrivateLink を使用すると、VPC エンドポイントを作成できます。これにより、Amazon VPC に基づいて、異なるアカウントや VPC のサービスに接続できます。AWS PrivateLink の詳細については、Amazon Virtual Private Cloud ユーザーガイドの「VPC エンドポイントサービス (AWS PrivateLink)」を参照してください。

Amazon VPC エンドポイントを使用して Data API を呼び出すことができます。Amazon VPC エンドポイントを使用することで、パブリック IP アドレスなしで Amazon VPC 内のアプリケーションと AWS ネットワーク内の Data API 間のトラフィックを維持できます。Amazon VPC エンドポイントは、公共のインターネット接続の制限に関連するコンプライアンスおよび規制要件を満たすのに役立ちます。例えば、Amazon VPC エンドポイントを使用する場合、Amazon EC2 インスタンスで実行されているアプリケーションと、それらを含む VPC 内の Data API 間のトラフィックを維持できます。

Amazon VPC エンドポイントを作成したら、アプリケーションでコードや設定を変更せずに、エンドポイントの使用をスタートできます。

Data API に Amazon VPC エンドポイントを作成するには

1. AWS マネジメントコンソール にサインインして、Amazon VPC コンソール (https://console.aws.amazon.com/vpc/) を開きます。
2. [エンドポイント] を選択し、[エンドポイントの作成] を選択します。
3. [エンドポイントの作成] ページの [サービスカテゴリ] で [AWS サービス] を選択します。を使用する場合[サービス名] で、[redshift Data(`com.amazonaws.region.redshift-data`)]。
4. [VPC] の場合は、VPC を選択してエンドポイントを作成します。

   Data API コールを行うアプリケーションを含む VPC を選択します。
5. [サブネット] で、アプリケーションを実行している AWS のサービスによって使用される各アベイラビリティーゾーン (AZ) のサブネットを選択します。

   Amazon VPC エンドポイントを作成するには、エンドポイントにアクセスできるプライベート IP アドレスの範囲を指定します。これを行うには、各アベイラビリティーゾーンのサブネットを選択します。これにより、VPC エンドポイントは各アベイラビリティーゾーンに固有のプライベート IP アドレス範囲に制限され、各アベイラビリティーゾーンに Amazon VPC エンドポイントが作成されます。
6. [DNS 名を有効にする] で、[このエンドポイントで有効にする] を選択します。

   プライベート DNS は、標準の Data API DNS ホスト名 (`https://redshift-data.region.amazonaws.com`) を、Amazon VPC エンドポイントに固有の DNS ホスト名に関連付けられたプライベート IP アドレスに解決します。その結果、Data API エンドポイント URL を更新するためのコードや設定を変更せずに、AWS CLIまたは AWS SDK を使用して Data API VPC エンドポイントにアクセスできます。
7. セキュリティグループで、Amazon VPC エンドポイントに関連付けるセキュリティグループを選択します。

   アプリケーションを実行している AWS のサービスへのアクセスを許可するセキュリティグループを選択します。例えば、Amazon EC2 インスタンスでアプリケーションを実行している場合は、Amazon EC2 インスタンスへのアクセスを許可するセキュリティグループを選択します。セキュリティグループを使用すると、VPC 内のリソースから Amazon VPC エンドポイントへのトラフィックを制御できます。
8. [Create endpoint (エンドポイントの作成)] を選択します。

エンドポイントを作成したら、AWS マネジメントコンソールでリンクを選択して、エンドポイントの詳細を表示します。

エンドポイントの [Details (詳細)] タブには、Amazon VPC エンドポイントの作成中に生成された DNS ホスト名が表示されます。

標準エンドポイント (`redshift-data.region.amazonaws.com`) または VPC 固有のエンドポイントの 1 つを使用して、Amazon VPC 内の Data API を呼び出すことができます。スタンダード Data API エンドポイントは、自動的に Amazon VPC エンドポイントにルーティングします。このルーティングは、Amazon VPC エンドポイントの作成時にプライベート DNS ホスト名が有効になったために発生します。

Data API コールで Amazon VPC エンドポイントを使用すると、アプリケーションと Data API 間のすべてのトラフィックは、それらを含む Amazon VPC に残ります。Amazon VPC エンドポイントは、任意のタイプの Data API コールに使用できます。Data API を呼び出す方法については、「Amazon Redshift Data API を呼び出す際の考慮事項」を参照してください。

クラスターへの接続時にデータベースグループに参加する

データベースグループは、データベースユーザーのコレクションです。データベース権限はグループに付与できます。管理者は、SQL を Data API で実行するときに、これらのデータベースグループを考慮に入れて IAM ロールを設定できます。データベースグループの詳細については、「Amazon Redshift データベース開発者ガイド」の「グループ」を参照してください。

Data API がクラスターに接続したときに、呼び出しで指定したデータベースユーザーがデータベースグループに参加するように、Data API 呼び出し元の IAM ロールを設定できます。この機能は、プロビジョニングされたクラスターに接続する場合にのみサポートされます。Redshift Serverless ワークグループに接続する場合はサポートされません。Data API の呼び出し元の IAM ロールでは、`redshift:JoinGroup` アクションも許可する必要があります。

これを設定するには、IAM ロールにタグを追加します。呼び出し元の IAM ロールの管理者は、キー `RedshiftDbGroups` およびデータベースグループのリストのキー値を使用してタグを追加します。値は、コロン (:) で区切られたデータベースグループの名前のリストで、全長は最大 256 文字です。データベースグループは、接続されたデータベースで事前に定義しておく必要があります。指定したグループがデータベースに見つからない場合、そのグループは無視されます。例えば、データベースグループ `accounting` および `retail` の場合、キー値は `accounting:retail` です。タグのキーおよび値のペア `{"Key":"RedshiftDbGroups","Value":"accounting:retail"}` は、Data API への呼び出しで、どのデータベースグループが指定のデータベースユーザーに関連付けられているかを Data API で判断するために使用します。

データベースグループに参加するには

1. AWS マネジメントコンソール にサインインして、IAM コンソール https://console.aws.amazon.com/iam/ を開きます。
2. コンソールのナビゲーションペインで、ロール を選択し、編集するロールの名前を選択します。
3. [タグ] タブ、[タグを管理] の順に選択します。
4. [タグを追加] を選択し、キー RedshiftDbGroups と値 (*database-groups-colon-separated* のリスト) を追加します。
5. [Save changes] (変更の保存) をクリックします。

   これで、この IAM ロールがアタッチされた IAM プリンシパルから Data API を呼び出すと、指定したデータベースユーザーが IAM ロールに指定されたデータベースグループに参加します。

IAM ロールや IAM ユーザーなど、プリンシパルにタグをアタッチする方法については、「IAM ユーザーガイド」の「IAM リソースのタグ付け」を参照してください。

信頼できる ID の伝播で Data API を使用する

Amazon Redshift アカウント管理者は、Amazon Redshift クラスターまたはワークグループを AWS IAM アイデンティティセンター と統合できます。これにより、シングルサインオンで Amazon Redshift へのワークフォースアクセスを管理できます。詳細については、「Amazon Redshift での AWS IAM アイデンティティセンターの統合の設定」を参照してください。Amazon Redshift Data API は、IAM アイデンティティセンターのユーザー ID を Amazon Redshift クラスターまたはワークグループ、および AWS Lake Formation などの他の サービスに連鎖的に伝播することをサポートしています。「Access AWS services programmatically using trusted identity propagation」の手順に従って、Data API を使用した設定およびクエリを実行することができます。

ID 拡張 IAM ロールセッションから IAM アイデンティティセンターのユーザー ID を使用して Data API を呼び出す場合、結果のステートメントとステートメントの結果には、同じ IAM アイデンティティセンターユーザーを使用してのみアクセスできます。例えば、次の AWS CLI コマンドは `execute-statement` オペレーションを呼び出して、信頼できる ID の伝播で SQL コマンドを実行します。

```
aws redshift-data execute-statement
--sql "select current_user;"
--cluster-id mycluster
```

```
--database dev
```

次の AWS CLI コマンドは、batch-execute-statement オペレーションを呼び出して 2 つの SQL コマンドを実行します。

```
aws redshift-data batch-execute-statement
--sqls  "select current_user;"  "select current_date;"
--cluster-id mycluster
--database dev
```

ID 拡張 IAM ロールセッションによって送信された cancel-statement、describe-statement、get-statement-result、および get-statement-result-v2 のステートメントにアクセスするには、IAM アイデンティティセンターのユーザーと IAM ロールが、execute-statement または batch-execute-statement の実行に使用される認証情報と一致する必要があります。例えば、次の AWS CLI コマンドは SQL ステートメントの結果を取得します。

```
aws redshift-data get-statement-result
--id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

ステートメントを一覧表示するには、割り当てられている Amazon Redshift IAM アイデンティティセンターアプリケーションのみに IAM アイデンティティセンターのユーザーがアクセスできるように、cluster-identifier または workgroup-name パラメータを指定する必要があります。例えば、次の AWS CLI コマンドは、特定のクラスターのステートメントを一覧表示します。

```
aws redshift-data list-statements
--cluster-identifier mycluster
```

信頼できる ID の伝播を使用して、クラスターまたはワークグループ内のデータベースオブジェクトにアクセスする Data API オペレーションを呼び出すこともできます。これには list-databases、list-schemas、list-tables、および describe-table オペレーションが含まれます。

IAM アイデンティティセンターのユーザーによって行われた API コールは、AWS CloudTrail で追跡できます。CloudTrail イベントの onBehalfOf セクションには、IAM アイデンティティセンターのユーザー ID と ID ストア ARN が表示されます。次の例は、IAM アイデンティティセンターのユーザー ID a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 と ID ストア ARN arn:aws:identitystore::123456789012:identitystore/d-9067bc44d2 を含む onBehalfOf セクションを表示している CloudTrail イベントのスニペットを示しています。

```
{
            "eventVersion":"1.10",
            "userIdentity":{
            "type":"AssumedRole",
            ...
            },
            "onBehalfOf":{
            "userId":"a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "identityStoreArn":"arn:aws:identitystore::123456789012:identitystore/
d-9067bc44d2"
            }
            },
            "eventTime":"2025-01-13T04:46:27Z",
            "eventSource":"redshift-data.amazonaws.com",
            "eventName":"ExecuteStatement",
            "awsRegion":"us-east-1"
            }
```

次の SQL コマンドを実行して、IAM アイデンティティセンターのユーザーが送信したクエリを確認できます。この例では、アイデンティティセンターに登録されている E メールは username@example.com です。

```
SELECT
    h.query_id,
    h.database_name,
    h.status,
    h.query_text,
    u.usename,
    h.start_time,
    h.end_time
FROM
    sys_query_history h
LEFT JOIN
    pg_user u
ON
    h.user_id = u.usesysid
where u.usename='awsidc:username@example.com'
ORDER BY
    h.start_time DESC;
```

Data API の呼び出し

Data API または AWS CLI を呼び出して、クラスターまたはサーバーレスワークグループで SQL ステートメントを実行できます。SQL ステートメントを実行する主なオペレーションは、Amazon Redshift Data API リファレンスの `ExecuteStatement` と `BatchExecuteStatement` です。Data API は、AWSSDK でサポートされているプログラミング言語をサポートしています。詳細については、「AWS での構築ツール」を参照してください。

Data API を呼び出すコードの例については、GitHub の「Getting Started with Redshift Data API」を参照してください。このリポジトリには、Amazon EC2、AWS Glue Data Catalog または Amazon SageMaker ランタイムからの Amazon Redshift データへのアクセスに AWS Lambda を使用する例があります。プログラミング言語の例としては、Python、Go、Java、Javascript があります。

Data API は、AWS CLIを使用して呼び出すことができます。

次の例では、AWS CLIを使用して Data API を呼び出します。例を実行するには、環境に合わせてパラメータ値を編集します。多くの例では、クラスターに対して実行するため、`cluster-identifier` が指定されています。サーバーレスワークグループに対して実行する場合は、代わりに `workgroup-name` を指定します。これらの例は、データ API 操作の一部を示しています。詳細については、AWS CLIコマンドリファレンスを参照してください。

次の例のコマンドは分割され、読みやすい形式になっています。すべてのパラメータとレスポンスがすべての例で表示されるとは限りません。完全なリクエスト構文、リクエストパラメータ、レスポンス構文、レスポンス要素の API 定義については、「Amazon Redshift Data API リファレンス」を参照してください。

SQL ステートメントを Amazon Redshift データウェアハウスに渡す

このページの例は、SQL ステートメントをデータウェアハウスに渡すさまざまな方法について説明しています。

SQL ステートメントを実行する

SQL ステートメントを実行するには、`aws redshift-data execute-statement` AWS CLI コマンドを使用します。

必要に応じて、`WaitTimeSeconds` パラメータを使用して、API がレスポンスを返す前にステートメントが完了するまでの追加の時間を確保します。ステートメントが完了すると、レスポンスには `Status`、`RedshiftPid`、および `HasResultSet` も含まれます。詳細については、「ロングポーリングによる API コールの削減」を参照してください。

次の AWS CLI コマンドは、クラスターに対して SQL ステートメントを実行し、結果を取得する識別子を返します。この例では、AWS Secrets Manager の認証方法を使用します。

```
aws redshift-data execute-statement
    --secret-arn arn:aws:secretsmanager:us-west-2:123456789012:secret:myuser-secret-
hKgPWn
    --cluster-identifier mycluster-test
    --sql "select * from stl_query limit 1"
    --database dev
```

次は、レスポンスの例です。

```
{
    "ClusterIdentifier": "mycluster-test",
    "CreatedAt": 1598323175.823,
    "Database": "dev",
    "Id": "c016234e-5c6c-4bc5-bb16-2c5b8ff61814",
    "SecretArn": "arn:aws:secretsmanager:us-west-2:123456789012:secret:myuser-secret-
hKgPWn"
}
```

次の AWS CLI コマンドは、クラスターに対して SQL ステートメントを実行し、結果を取得する識別子を返します。この例では、一時的な認証情報認証方法を使用します。

```
aws redshift-data execute-statement
    --db-user myuser
    --cluster-identifier mycluster-test
    --database dev
    --sql "select * from stl_query limit 1"
```

次は、レスポンスの例です。

```
{
    "ClusterIdentifier": "mycluster-test",
    "CreatedAt": 1598306924.632,
    "Database": "dev",
    "DbUser": "myuser",
    "Id": "d9b6c0c9-0747-4bf4-b142-e8883122f766"
}
```

次の AWS CLI コマンドは、サーバーレスワークグループに対して SQL ステートメントを実行し、結果を取得する識別子を返します。この例では、一時的な認証情報認証方法を使用します。

```
aws redshift-data execute-statement
    --database dev
    --workgroup-name myworkgroup
    --sql "select 1;"
```

次は、レスポンスの例です。

```
{
 "CreatedAt": "2022-02-11T06:25:28.748000+00:00",
 "Database": "dev",
 "DbUser": "IAMR:RoleName",
 "Id": "89dd91f5-2d43-43d3-8461-f33aa093c41e",
 "WorkgroupName": "myworkgroup"
}
```

次の AWS CLI コマンドは、クラスターに対して SQL ステートメントを実行し、結果を取得する識別子を返します。この例では、AWS Secrets Manager 認証方法と冪等性トークンを使用しています。

```
aws redshift-data execute-statement
    --secret-arn arn:aws:secretsmanager:us-west-2:123456789012:secret:myuser-secret-
hKgPWn
    --cluster-identifier mycluster-test
    --sql "select * from stl_query limit 1"
    --database dev
    --client-token b855dced-259b-444c-bc7b-d3e8e33f94g1
```

次は、レスポンスの例です。

```
{
    "ClusterIdentifier": "mycluster-test",
    "CreatedAt": 1598323175.823,
    "Database": "dev",
    "Id": "c016234e-5c6c-4bc5-bb16-2c5b8ff61814",
    "SecretArn": "arn:aws:secretsmanager:us-west-2:123456789012:secret:myuser-secret-
hKgPWn"
```

```
}
```

複数の SQL ステートメントを実行する

1 つのコマンドで複数の SQL ステートメントを実行するには、`aws redshift-data batch-execute-statement`AWS CLI コマンドを実行します。

必要に応じて、`WaitTimeSeconds` パラメータを使用して、API がレスポンスを返す前にバッチ全体が完了するまでの追加の時間を確保します。詳細については、「ロングポーリングによる API コールの削減」を参照してください。

次の AWS CLI コマンドは、クラスターに対して 3 つの SQL ステートメントを実行し、結果を取得する識別子を返します。この例では、一時的な認証情報認証方法を使用します。

```
aws redshift-data batch-execute-statement
    --db-user myuser
    --cluster-identifier mycluster-test
    --database dev
    --sqls "set timezone to BST" "select * from mytable" "select * from another_table"
```

次は、レスポンスの例です。

```
{
    "ClusterIdentifier": "mycluster-test",
    "CreatedAt": 1598306924.632,
    "Database": "dev",
    "DbUser": "myuser",
    "Id": "d9b6c0c9-0747-4bf4-b142-e8883122f766"
}
```

次の AWS CLI コマンドは、`AUTO_COMMIT` 実行モードを使用して 3 つの SQL ステートメントを別々のトランザクションで実行します。各ステートメントは個別にコミットされます。あるステートメントが失敗しても、残りのステートメントの実行が停止したり、既に完了したステートメントに影響を与えたりすることはありません。

```
aws redshift-data batch-execute-statement
    --db-user myuser
    --cluster-identifier mycluster-test
    --database dev
```

```
    --sqls "insert into mytable values (1)" "invalid sql statement" "insert into
 mytable values (3)"
    --execution-mode AUTO_COMMIT
```

次は、レスポンスの例です。

```
{
    "ClusterIdentifier": "mycluster-test",
    "CreatedAt": 1623979777.126,
    "Duration": 6591877,
    "HasResultSet": false,
    "Id": "b2906c76-fa6e-4cdf-8c5f-4de1ff9b7652",
    "RedshiftPid": 31459,
    "RedshiftQueryId": 0,
    "Status": "FAILED",
    "SubStatements": [
        {
            "CreatedAt": 1623979777.274,
            "Duration": 3396637,
            "HasResultSet": false,
            "Id": "b2906c76-fa6e-4cdf-8c5f-4de1ff9b7652:1",
            "QueryString": "insert into mytable values (1)",
            "RedshiftQueryId": -1,
            "Status": "FINISHED",
            "UpdatedAt": 1623979777.903
        },
        {
            "CreatedAt": 1623979777.274,
            "Duration": 1195240,
            "HasResultSet": false,
            "Id": "b2906c76-fa6e-4cdf-8c5f-4de1ff9b7652:2",
            "QueryString": "invalid sql statement",
            "RedshiftQueryId": -1,
            "Error": "ERROR: syntax error at or near \"invalid\"",
            "Status": "FAILED",
            "UpdatedAt": 1623979777.503
        },
        {
            "CreatedAt": 1623979777.274,
            "Duration": 1095981,
            "HasResultSet": false,
            "Id": "b2906c76-fa6e-4cdf-8c5f-4de1ff9b7652:3",
            "QueryString": "insert into mytable values (3)",
```

```
            "RedshiftQueryId": -1,
            "Status": "FINISHED",
            "UpdatedAt": 1623979777.903
        }
    ],
    "UpdatedAt": 1623979778.183
}
```

パラメータを使用してステートメントを実行する

名前付きパラメータは、execute-statement オペレーションと batch-execute-statement オペレーションの両方で使用できます。batch-execute-statement を使用する場合、パラメータはバッチ内のすべての SQL ステートメント間で共有されます。各 SQL ステートメントは、指定されたパラメータのサブセットを参照できますが、どのパラメータも最低 1 回はどこかの SQL ステートメントで使用する必要があります。

次の AWS CLI コマンドは、クラスターに対して SQL ステートメントを実行し、結果を取得する識別子を返します。この例では、AWS Secrets Manager の認証方法を使用します。SQL テキストには、名前付きパラメータ distance があります。この場合、述部で使用される距離は 5 です。SELECT ステートメントでは、列名の名前付きパラメータは述部でのみ使用できます。SQL ステートメントの名前付きパラメータの値は、parametersオプションで指定できます。

```
aws redshift-data execute-statement
    --secret-arn arn:aws:secretsmanager:us-west-2:123456789012:secret:myuser-secret-
hKgPWn
    --cluster-identifier mycluster-test
    --sql "SELECT ratecode FROM demo_table WHERE trip_distance > :distance"
    --parameters "[{\"name\": \"distance\", \"value\": \"5\"}]"
    --database dev
```

次は、レスポンスの例です。

```
{
    "ClusterIdentifier": "mycluster-test",
    "CreatedAt": 1598323175.823,
    "Database": "dev",
    "Id": "c016234e-5c6c-4bc5-bb16-2c5b8ff61814",
    "SecretArn": "arn:aws:secretsmanager:us-west-2:123456789012:secret:myuser-secret-
hKgPWn"
}
```

次の例では、サンプルデータベースの EVENT テーブルを使用します。詳細については、Amazon Redshift データベースデベロッパーガイドの「EVENT テーブル」を参照してください。

まだデータベースに EVENT テーブルにない場合、Data API を使用して次のように作成できます。

```
aws redshift-data execute-statement
--database dev
--cluster-id mycluster-test
--db-user awsuser
--sql "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)"
```

次のコマンドは、1 つの行を EVENT テーブルに挿入します。

```
aws redshift-data execute-statement
--database dev
--cluster-id mycluster-test
--db-user awsuser
--sql "insert into event
 values(:eventid, :venueid::smallint, :catid, :dateid, :eventname, :starttime)"
--parameters "[{\"name\": \"eventid\", \"value\": \"1\"}, {\"name\": \"venueid\",
 \"value\": \"1\"},
               {\"name\": \"catid\", \"value\": \"1\"},
               {\"name\": \"dateid\", \"value\": \"1\"},
               {\"name\": \"eventname\", \"value\": \"event 1\"},
               {\"name\": \"starttime\", \"value\": \"2022-02-22\"}]"
```

次のコマンドは、2 つ目の行を EVENT テーブルに挿入します。この例は以下を示します。

- id という名前のパラメータは SQL テキストで 4 回使用されます。
- 暗黙的な型変換は、パラメータ starttime を挿入する際に自動的に適用されます。
- venueid 列は、SMALLINT データ型にタイプキャストされます。
- DATE データ型を表す文字列は、TIMESTAMP データ型に暗黙的に変換されます。
- コメントは SQL テキスト内で使用できます。

```
aws redshift-data execute-statement
--database dev
--cluster-id mycluster-test
--db-user awsuser
--sql "insert into event values(:id, :id::smallint, :id, :id, :eventname, :starttime) /
*this is comment, and it won't apply parameterization for :id, :eventname or :starttime
 here*/"
--parameters "[{\"name\": \"eventname\", \"value\": \"event 2\"},
              {\"name\": \"starttime\", \"value\": \"2022-02-22\"},
              {\"name\": \"id\", \"value\": \"2\"}]"
```

次に、挿入された 2 つの行を示します。

```
 eventid | venueid | catid | dateid | eventname |      starttime
---------+---------+-------+--------+-----------+---------------------
       1 |       1 |     1 |      1 | event 1   | 2022-02-22 00:00:00
       2 |       2 |     2 |      2 | event 2   | 2022-02-22 00:00:00
```

次のコマンドでは、WHERE 句で名前付きパラメータを使用して、eventidが 1 である行を取得します。

```
aws redshift-data execute-statement
--database dev
--cluster-id mycluster-test
--db-user awsuser
--sql "select * from event where eventid=:id"
--parameters "[{\"name\": \"id\", \"value\": \"1\"}]"
```

次のコマンドを実行して、前の SQL ステートメントの SQL 結果を取得します。

```
aws redshift-data get-statement-result --id 7529ad05-b905-4d71-9ec6-8b333836eb5a
```

これにより次の結果が示されます。

```
{
    "Records": [
        [
            {
                "longValue": 1
            },
            {
                "longValue": 1
            },
            {
                "longValue": 1
            },
            {
                "longValue": 1
            },
            {
                "stringValue": "event 1"
            },
            {
                "stringValue": "2022-02-22 00:00:00.0"
            }
        ]
    ],
    "ColumnMetadata": [
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": true,
            "label": "eventid",
            "length": 0,
            "name": "eventid",
            "nullable": 0,
            "precision": 10,
            "scale": 0,
            "schemaName": "public",
            "tableName": "event",
            "typeName": "int4"
        },
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": true,
```

```
            "label": "venueid",
            "length": 0,
            "name": "venueid",
            "nullable": 0,
            "precision": 5,
            "scale": 0,
            "schemaName": "public",
            "tableName": "event",
            "typeName": "int2"
        },
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": true,
            "label": "catid",
            "length": 0,
            "name": "catid",
            "nullable": 0,
            "precision": 5,
            "scale": 0,
            "schemaName": "public",
            "tableName": "event",
            "typeName": "int2"
        },
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": true,
            "label": "dateid",
            "length": 0,
            "name": "dateid",
            "nullable": 0,
            "precision": 5,
            "scale": 0,
            "schemaName": "public",
            "tableName": "event",
            "typeName": "int2"
        },
        {
            "isCaseSensitive": true,
            "isCurrency": false,
            "isSigned": false,
            "label": "eventname",
            "length": 0,
```

```
            "name": "eventname",
            "nullable": 1,
            "precision": 200,
            "scale": 0,
            "schemaName": "public",
            "tableName": "event",
            "typeName": "varchar"
        },
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": false,
            "label": "starttime",
            "length": 0,
            "name": "starttime",
            "nullable": 1,
            "precision": 29,
            "scale": 6,
            "schemaName": "public",
            "tableName": "event",
            "typeName": "timestamp"
        }
    ],
    "TotalNumRows": 1
}
```

次の AWS CLI コマンドでは、batch-execute-statement を使用して、名前付きパラメータを共有する 2 つの SQL ステートメントをクラスターに対して実行します。SQL テキストには、名前付きパラメータ id および name があります。最初の SQL ステートメントでは id パラメータを使用し、2 番目の SQL ステートメントでは name パラメータを使用します。

```
aws redshift-data batch-execute-statement
    --db-user myuser
    --cluster-identifier mycluster-test
    --database dev
    --sqls "SELECT * FROM mytable WHERE id = :id" "SELECT * FROM mytable WHERE name
 = :name"
    --parameters "[{\"name\": \"id\", \"value\": \"1\"},{\"name\": \"name\", \"value\":
 \"Alice\"}]"
```

次は、レスポンスの例です。

```
{
    "ClusterIdentifier": "mycluster-test",
    "CreatedAt": 1598306924.632,
    "Database": "dev",
    "DbUser": "myuser",
    "Id": "d9b6c0c9-0747-4bf4-b142-e8883122f766"
}
```

次の AWS CLI コマンドでは 2 つの SQL ステートメントを実行し、両方のステートメントで同じパラメータを使用します。id という名前のパラメータは、INSERT ステートメントと SELECT ステートメントで使用されています。

```
aws redshift-data batch-execute-statement
    --db-user myuser
    --cluster-identifier mycluster-test
    --database dev
    --sqls "INSERT INTO mytable VALUES(:id, :name, :address)" "SELECT * FROM mytable
 WHERE id = :id"
    --parameters "[{\"name\": \"id\", \"value\": \"1\"},{\"name\": \"name\", \"value\":
 \"Alice\"},{\"name\": \"address\", \"value\": \"Seattle\"}]"
```

次は、レスポンスの例です。

```
{
    "ClusterIdentifier": "mycluster-test",
    "CreatedAt": 1598306924.632,
    "Database": "dev",
    "DbUser": "myuser",
    "Id": "d9b6c0c9-0747-4bf4-b142-e8883122f766"
}
```

SQL ステートメントに関するメタデータを一覧表示する

`aws redshift-data list-statements` AWS CLI コマンドを使用して、SQL ステートメントに関するメタデータを一覧表示します。このコマンドの実行の承認は、呼び出し元の IAM アクセス許可に基づいています。

次の AWS CLI コマンドは、実行された SQL ステートメントを示しています。

```
aws redshift-data list-statements
    --status ALL
```

次は、レスポンスの例です。

```
{
    "Statements": [
        {
            "CreatedAt": 1598306924.632,
            "Id": "d9b6c0c9-0747-4bf4-b142-e8883122f766",
            "QueryString": "select * from stl_query limit 1",
            "Status": "FINISHED",
            "UpdatedAt": 1598306926.667
        },
        {
            "CreatedAt": 1598311717.437,
            "Id": "e0ebd578-58b3-46cc-8e52-8163fd7e01aa",
            "QueryString": "select * from stl_query limit 1",
            "Status": "FAILED",
            "UpdatedAt": 1598311719.008
        },
        {
            "CreatedAt": 1598313683.65,
            "Id": "c361d4f7-8c53-4343-8c45-6b2b1166330c",
            "QueryString": "select * from stl_query limit 1",
            "Status": "ABORTED",
            "UpdatedAt": 1598313685.495
        },
        {
            "CreatedAt": 1598306653.333,
            "Id": "a512b7bd-98c7-45d5-985b-a715f3cfde7f",
            "QueryString": "select 1",
            "Status": "FINISHED",
            "UpdatedAt": 1598306653.992
        }
    ]
}
```

セッションを一覧表示する

過去 24 時間に作成したセッションを一覧表示するには、aws redshift-data list-sessions AWS CLI コマンドを使用します。デフォルトでは、コマンドはステータスが AVAILABLE または BUSY のセッションのみを返します。セッションステータス、コンピューティングターゲット (クラ

スターまたはサーバーレスワークグループ)、またはデータベースで結果をフィルタリングできます。IAM アクセス許可によって、このコマンドを実行できるかどうかが決まります。セッションの詳細については、「Amazon Redshift Data API を呼び出す際にセッションの再利用によって SQL ステートメントを実行する」を参照してください。

次の AWS CLI コマンドは、クラスターで実行されたセッションを一覧表示します。

```
aws redshift-data list-sessions
   --cluster-identifier mycluster-test
```

次は、レスポンスの例です。

```
{
    "Sessions": [
        {
            "SessionId": "5a254dc6-4fc2-4203-87a8-551155432ee4",
            "Status": "AVAILABLE",
            "CreatedAt": 1703022996.436,
            "UpdatedAt": 1703023010.221,
            "Database": "dev",
            "DbUser": "awsuser",
            "ClusterIdentifier": "mycluster-test",
            "SessionAliveSeconds": 10,
            "SessionTtl": 1703023020.436
        },
        {
            "SessionId": "8b365ed7-5gd7-5314-98b9-662266543ff5",
            "Status": "BUSY",
            "CreatedAt": 1703023100.512,
            "UpdatedAt": 1703023150.874,
            "Database": "dev",
            "DbUser": "awsuser",
            "ClusterIdentifier": "mycluster-test",
            "SessionAliveSeconds": 60,
            "SessionTtl": 1703023210.512,
            "CurrentStatementId": "c016234e-5c6c-4bc5-bb16-2c5b8ff61814"
        }
    ]
}
```

次の AWS CLI コマンドは、セッションステータスでフィルタリングして、サーバーレスワークグループの CLOSED セッションを一覧表示します。

```
aws redshift-data list-sessions
    --workgroup-name myworkgroup
    --status CLOSED
```

次は、レスポンスの例です。

```
{
    "Sessions": [
        {
            "SessionId": "07c5ffea-76d6-4786-b62c-4fe3ef529680",
            "Status": "CLOSED",
            "CreatedAt": 1703019996.436,
            "UpdatedAt": 1703020056.436,
            "Database": "dev",
            "DbUser": "IAMR:RoleName",
            "WorkgroupName": "myworkgroup",
            "SessionAliveSeconds": 60,
            "SessionTtl": 1703020056.436
        }
    ]
}
```

> **Note**
>
> ID 拡張 IAM ロールセッションを使用する場合は、`cluster-identifier` または `workgroup-name` パラメータのいずれかを指定する必要があります。このパラメータを使用すると、AWS IAM アイデンティティセンター ユーザーは割り当てられた Amazon Redshift IAM アイデンティティセンターアプリケーションにのみアクセスできます。詳細については、「信頼できる ID の伝播で Data API を使用する」を参照してください。

SQL ステートメントに関するメタデータの記述を取得する

SQL ステートメントのメタデータの記述を取得するには、`aws redshift-data describe-statement` AWS CLI コマンドを使用します。このコマンドを使用して、サブステートメント ID を

指定してバッチ内の個々のサブステートメントを記述することもできます。このコマンドの実行の承認は、呼び出し元の IAM アクセス許可に基づいています。

必要に応じて、ポーリングリクエストの数を減らすには、WaitTimeSeconds パラメータを使用して、この API コール内でオペレーションのステータスが更新されるまでの追加の時間を確保します。詳細については、「ロングポーリングによる API コールの削減」を参照してください。

次の AWS CLI コマンドは、SQL ステートメントを記述します。

```
aws redshift-data describe-statement
    --id d9b6c0c9-0747-4bf4-b142-e8883122f766
```

次は、レスポンスの例です。

```
{
    "ClusterIdentifier": "mycluster-test",
    "CreatedAt": 1598306924.632,
    "Duration": 1095981511,
    "Id": "d9b6c0c9-0747-4bf4-b142-e8883122f766",
    "QueryString": "select * from stl_query limit 1",
    "RedshiftPid": 20859,
    "RedshiftQueryId": 48879,
    "ResultRows": 1,
    "ResultSize": 4489,
    "Status": "FINISHED",
    "UpdatedAt": 1598306926.667
}
```

次に、複数の SQL ステートメントで batch-execute-statement コマンドを実行した後の describe-statement レスポンスの例を示します。

```
{
    "ClusterIdentifier": "mayo",
    "CreatedAt": 1623979777.126,
    "Duration": 6591877,
    "HasResultSet": true,
    "Id": "b2906c76-fa6e-4cdf-8c5f-4de1ff9b7652",
    "RedshiftPid": 31459,
    "RedshiftQueryId": 0,
    "ResultRows": 2,
    "ResultSize": 22,
```

```
    "Status": "FINISHED",
    "SubStatements": [
        {
            "CreatedAt": 1623979777.274,
            "Duration": 3396637,
            "HasResultSet": true,
            "Id": "b2906c76-fa6e-4cdf-8c5f-4de1ff9b7652:1",
            "QueryString": "select 1;",
            "RedshiftQueryId": -1,
            "ResultRows": 1,
            "ResultSize": 11,
            "Status": "FINISHED",
            "UpdatedAt": 1623979777.903
        },
        {
            "CreatedAt": 1623979777.274,
            "Duration": 3195240,
            "HasResultSet": true,
            "Id": "b2906c76-fa6e-4cdf-8c5f-4de1ff9b7652:2",
            "QueryString": "select 2;",
            "RedshiftQueryId": -1,
            "ResultRows": 1,
            "ResultSize": 11,
            "Status": "FINISHED",
            "UpdatedAt": 1623979778.076
        }
    ],
    "UpdatedAt": 1623979778.183
}
```

次の AWS CLI コマンドは、サブステートメント ID を指定して、バッチ内の特定のサブステートメントを記述します。

```
aws redshift-data describe-statement
    --id b2906c76-fa6e-4cdf-8c5f-4de1ff9b7652:2
```

次は、レスポンスの例です。バッチがまだ完了していないため、親ステートメントのステータスは STARTED です。SubStatements 配列には、リクエストされたサブステートメントのみが含まれます。

```
{
```

```
    "ClusterIdentifier": "mayo",
    "CreatedAt": 1623979777.126,
    "Duration": 0,
    "HasResultSet": false,
    "Id": "b2906c76-fa6e-4cdf-8c5f-4de1ff9b7652",
    "RedshiftPid": 31459,
    "RedshiftQueryId": 0,
    "Status": "STARTED",
    "SubStatements": [
        {
            "CreatedAt": 1623979777.274,
            "Duration": 3195240,
            "HasResultSet": true,
            "Id": "b2906c76-fa6e-4cdf-8c5f-4de1ff9b7652:2",
            "QueryString": "select 2;",
            "RedshiftQueryId": -1,
            "ResultRows": 1,
            "ResultSize": 11,
            "Status": "FINISHED",
            "UpdatedAt": 1623979778.076
        }
    ],
    "UpdatedAt": 1623979777.503
}
```

SQL ステートメントの結果を取得する

実行された SQL ステートメントの結果を取得するには、`redshift-data get-statement-result` または `redshift-data get-statement-result-v2` AWS CLI コマンドを使用します。`get-statement-result` の結果は JSON 形式です。`get-statement-result-v2` の結果は CSV 形式です。レスポンスで受け取った `Id` を、`execute-statement`または`batch-execute-statement` に提供できます。`batch-execute-statement` によって実行された SQL ステートメントの `Id` 値は、`describe-statement` の結果で取得できます。この値には、`b2906c76-fa6e-4cdf-8c5f-4de1ff9b7652:2` のように、コロンとシーケンス番号が末尾に付きます。`batch-execute-statement` で複数の SQL ステートメントを実行する場合、各 SQL ステートメントには `describe-statement` に示されているような `Id` 値があります。このコマンドの実行の承認は、呼び出し元の IAM アクセス許可に基づいています。

必要に応じて、ポーリングリクエストの数を減らすには、`WaitTimeSeconds` パラメータを使用して、API がステートメント結果を返す前にステートメントが完了するまでの追加の時間を確保します。詳細については、「ロングポーリングによる API コールの削減」を参照してください。

次のステートメントは、execute-statement によって実行された SQL ステートメントの結果を返します。ResultFormat はデフォルトで JSON に設定されます。結果を取得するには、get-statement-result オペレーションを呼び出します。

```
aws redshift-data get-statement-result
    --id d9b6c0c9-0747-4bf4-b142-e8883122f766
```

次のステートメントは、batch-execute-statementによって実行された 2 番目の SQL ステートメントの結果を返します。

```
aws redshift-data get-statement-result
    --id b2906c76-fa6e-4cdf-8c5f-4de1ff9b7652:2
```

以下は、get-statement-result への呼び出しに対するレスポンスの例です。SQL 結果は、レスポンスの Records キーに JSON 形式で返されます。

```
{
    "ColumnMetadata": [
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": true,
            "label": "userid",
            "length": 0,
            "name": "userid",
            "nullable": 0,
            "precision": 10,
            "scale": 0,
            "schemaName": "",
            "tableName": "stll_query",
            "typeName": "int4"
        },
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": true,
            "label": "query",
            "length": 0,
            "name": "query",
            "nullable": 0,
```

```
            "precision": 10,
            "scale": 0,
            "schemaName": "",
            "tableName": "stll_query",
            "typeName": "int4"
        },
        {
            "isCaseSensitive": true,
            "isCurrency": false,
            "isSigned": false,
            "label": "label",
            "length": 0,
            "name": "label",
            "nullable": 0,
            "precision": 320,
            "scale": 0,
            "schemaName": "",
            "tableName": "stll_query",
            "typeName": "bpchar"
        },
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": true,
            "label": "xid",
            "length": 0,
            "name": "xid",
            "nullable": 0,
            "precision": 19,
            "scale": 0,
            "schemaName": "",
            "tableName": "stll_query",
            "typeName": "int8"
        },
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": true,
            "label": "pid",
            "length": 0,
            "name": "pid",
            "nullable": 0,
            "precision": 10,
            "scale": 0,
```

```
            "schemaName": "",
            "tableName": "stll_query",
            "typeName": "int4"
        },
        {
            "isCaseSensitive": true,
            "isCurrency": false,
            "isSigned": false,
            "label": "database",
            "length": 0,
            "name": "database",
            "nullable": 0,
            "precision": 32,
            "scale": 0,
            "schemaName": "",
            "tableName": "stll_query",
            "typeName": "bpchar"
        },
        {
            "isCaseSensitive": true,
            "isCurrency": false,
            "isSigned": false,
            "label": "querytxt",
            "length": 0,
            "name": "querytxt",
            "nullable": 0,
            "precision": 4000,
            "scale": 0,
            "schemaName": "",
            "tableName": "stll_query",
            "typeName": "bpchar"
        },
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": false,
            "label": "starttime",
            "length": 0,
            "name": "starttime",
            "nullable": 0,
            "precision": 29,
            "scale": 6,
            "schemaName": "",
            "tableName": "stll_query",
```

```
            "typeName": "timestamp"
        },
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": false,
            "label": "endtime",
            "length": 0,
            "name": "endtime",
            "nullable": 0,
            "precision": 29,
            "scale": 6,
            "schemaName": "",
            "tableName": "stll_query",
            "type": 93,
            "typeName": "timestamp"
        },
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": true,
            "label": "aborted",
            "length": 0,
            "name": "aborted",
            "nullable": 0,
            "precision": 10,
            "scale": 0,
            "schemaName": "",
            "tableName": "stll_query",
            "typeName": "int4"
        },
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": true,
            "label": "insert_pristine",
            "length": 0,
            "name": "insert_pristine",
            "nullable": 0,
            "precision": 10,
            "scale": 0,
            "schemaName": "",
            "tableName": "stll_query",
            "typeName": "int4"
```

```
        },
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": true,
            "label": "concurrency_scaling_status",
            "length": 0,
            "name": "concurrency_scaling_status",
            "nullable": 0,
            "precision": 10,
            "scale": 0,
            "schemaName": "",
            "tableName": "stll_query",
            "typeName": "int4"
        }
    ],
    "Records": [
        [
            {
                "longValue": 1
            },
            {
                "longValue": 3
            },
            {
                "stringValue": "health"
            },
            {
                "longValue": 1023
            },
            {
                "longValue": 15279
            },
            {
                "stringValue": "dev"
            },
            {
                "stringValue": "select system_status from stv_gui_status;"
            },
            {
                "stringValue": "2020-08-21 17:33:51.88712"
            },
            {
                "stringValue": "2020-08-21 17:33:52.974306"
```

```
            },
            {
                "longValue": 0
            },
            {
                "longValue": 0
            },
            {
                "longValue": 6
            }
        ]
    ],
    "TotalNumRows": 1
}
```

次の例は、execute-statement によって実行され、結果を JSON 形式で返す SQL ステートメントを示しています。テーブル testingtable には整数列が 3 つ (col1、col2、col3) あり、3 つの行に値 (1、2、3)、(4、5、6)、および (7、8、9) が含まれています。

```
aws redshift-data execute-statement
    --database dev
    --sql "SELECT col1, col2, col3 FROM testingtable"
    --cluster-id mycluster-test
    --result-format JSON
```

```
{
    "ClusterIdentifier": "mycluster-test",
    "CreatedAt": "2024-04-02T16:45:25.144000+00:00",
    "Database": "dev",
    "DbUser": "IAMR:Administrator",
    "Id": "d468d942-6df9-4f85-8ae3-bac01a61aec3"
}
```

以下は、get-statement-result への呼び出しに対するレスポンスの例です。SQL 結果は、レスポンスの Records キーに JSON 形式で返されます。

```
aws redshift-data get-statement-result
    --id d468d942-6df9-4f85-8ae3-bac01a61aec3
```

```
{
    "Records": [
        [
            {
                "longValue": 1
            },
            {
                "longValue": 2
            },
            {
                "longValue": 3
            }
        ],
        [
            {
                "longValue": 4
            },
            {
                "longValue": 5
            },
            {
                "longValue": 6
            }
        ],
        [
            {
                "longValue": 7
            },
            {
                "longValue": 8
            },
            {
                "longValue": 9
            }
        ]
    ],
    "ColumnMetadata": [
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": true,
            "label": "col1",
            "name": "col1",
```

```
            "nullable": 1,
            "precision": 10,
            "scale": 0,
            "schemaName": "public",
            "tableName": "testingtable",
            "typeName": "int4",
            "length": 0
        },
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": true,
            "label": "col2",
            "name": "col2",
            "nullable": 1,
            "precision": 10,
            "scale": 0,
            "schemaName": "public",
            "tableName": "testingtable",
            "typeName": "int4",
            "length": 0
        },
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": true,
            "label": "col3",
            "name": "col3",
            "nullable": 1,
            "precision": 10,
            "scale": 0,
            "schemaName": "public",
            "tableName": "testingtable",
            "typeName": "int4",
            "length": 0
        }
    ],
    "TotalNumRows": 3
}
```

次の例は、execute-statement によって実行され、結果を CSV 形式で返す SQL ステートメントを示しています。テーブル testingtable には整数列が 3 つ (col1、col2、col3) あり、3 つの行に値 (1、2、3)、(4、5、6)、および (7、8、9) が含まれています。

```
aws redshift-data execute-statement
    --database dev
    --sql "SELECT col1, col2, col3 FROM testingtable"
    --cluster-id mycluster-test
    --result-format CSV
```

```
{
    "ClusterIdentifier": "mycluster-test",
    "CreatedAt": "2024-04-02T16:45:25.144000+00:00",
    "Database": "dev",
    "DbUser": "IAMR:Administrator",
    "Id": "d468d942-6df9-4f85-8ae3-bac01a61aec3"
}
```

以下は、get-statement-result-v2 への呼び出しに対するレスポンスの例です。SQL 結果は、レスポンスの Records キーに CSV 形式で返されます。行はキャリッジリターンと改行 (\r\n) で区切られます。Records で返される最初の行は列ヘッダーです。CSV 形式で返される結果は 1 MB 単位で返され、各チャンクには最大 1MB 分の任意の数の行が含まれます。

```
aws redshift-data get-statement-result-v2
    --id d468d942-6df9-4f85-8ae3-bac01a61aec3
```

```
{
    "Records": [
        {
            "CSVRecords": "col1,col2,col3\r\n1,2,3\r\n4,5,6\r\n7,8,9\r\n"
        }
    ],
    "ColumnMetadata": [
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": true,
            "label": "col1",
            "name": "col1",
            "nullable": 1,
            "precision": 10,
            "scale": 0,
            "schemaName": "public",
```

```
            "tableName": "testingtable",
            "typeName": "int4",
            "length": 0
        },
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": true,
            "label": "col2",
            "name": "col2",
            "nullable": 1,
            "precision": 10,
            "scale": 0,
            "schemaName": "public",
            "tableName": "testingtable",
            "typeName": "int4",
            "length": 0
        },
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": true,
            "label": "col3",
            "name": "col3",
            "nullable": 1,
            "precision": 10,
            "scale": 0,
            "schemaName": "public",
            "tableName": "testingtable",
            "typeName": "int4",
            "length": 0
        }
    ],
    "TotalNumRows": 3,
    "ResultFormat": "csv"
}
```

ロングポーリングによる API コールの削減

デフォルトでは、Data API オペレーションは即座にレスポンスを返します。オペレーションが進行中のステートメントに対するものである場合は、ステートメントが終了状態になるまで繰り返しポーリングする必要があります。ロングポーリングでは、`WaitTimeSeconds` を 1～30 の値に設定しま

す。API は、ステートメントの終了または WaitTimeSeconds への到達のいずれか早い方まで、同期レスポンスの返信を遅らせます。

ロングポーリング
は、DescribeStatement、ExecuteStatement、BatchExecuteStatement、GetStatementResult、GetStatemer
でサポートされています。

オペレーションが WaitTimeSeconds を使用して新しいステートメントを送信するか、既存のステートメントを待機する場合。

- ステートメントが待機時間内に完了した場合、レスポンスには、オペレーションに応じて終了ステータスまたは結果データが含まれます。
- 待機時間が経過してもステートメントがまだ実行中の場合、レスポンスは現在の進行中のステータスを返します。
- ロングポーリングリクエストが行われる前にステートメントが既に完了している場合、API は直ちに終了ステータスを返します。
- WaitTimeSeconds を省略した場合、デフォルトの非同期動作が保持されます。

一般的なユースケース

ロングポーリングが役立つ場合の例を次に示します。

- 短いクエリを送信して完了を待つ – WaitTimeSeconds を指定して ExecuteStatement を使用します。クエリが終了するか、待機時間が経過すると、レスポンスはステートメントの Status、RedshiftPid、および HasResultSet を返します。完全なステートメントメタデータを取得するには、DescribeStatement を呼び出します。HasResultSet が true の場合、GetStatementResult を呼び出して結果を取得します。
- ステータスのロングポーリング – ExecuteStatement を使用してクエリを送信し、WaitTimeSeconds を指定して DescribeStatement を呼び出します。クエリが終了すると、DescribeStatement はステータス、所要時間、エラーの詳細を含むステートメントの完全なメタデータを返します。
- 結果のロングポーリング – ExecuteStatement を使用してクエリを送信し、WaitTimeSeconds を指定して GetStatementResult を呼び出します。クエリが終了すると、結果データが直接返されます。待機時間が経過してもクエリがまだ実行されている場合、結果がまだ利用できないことを示す ResourceNotFoundException が返されます。

- 複数のクエリを送信し、バッチ全体が完了するまで待機する – WaitTimeSeconds を指定して BatchExecuteStatement を使用します。呼び出しは、すべてのサブステートメントが完了するまで維持され、バッチの親 ID と全体的なステータスが返されます。
- 複数のクエリを送信し、サブステートメントのステータスをロングポーリングする – サブステートメント ID と WaitTimeSeconds を指定して DescribeStatement または GetStatementResult を使用します。そのサブステートメントが終了すると、バッチの残りの処理を待たずに、すぐに結果を返します。

テーブルの説明

aws redshift-data describe-table AWS CLI コマンドを使用して、テーブルを説明するメタデータを取得します。

次の AWS CLI コマンドは、クラスターに対して SQL ステートメントを実行し、テーブルを記述するメタデータを返します。この例では、AWS Secrets Manager の認証方法を使用します。

```
aws redshift-data describe-table
    --cluster-identifier mycluster-test
    --database dev
    --schema information_schema
    --table sql_features
    --secret-arn arn:aws:secretsmanager:us-west-2:123456789012:secret:myuser-secret-
hKgPWn
```

次は、レスポンスの例です。

```
{
    "ColumnList": [
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": false,
            "length": 2147483647,
            "name": "feature_id",
            "nullable": 1,
            "precision": 2147483647,
            "scale": 0,
            "schemaName": "information_schema",
            "tableName": "sql_features",
            "typeName": "character_data"
```

```
        },
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": false,
            "length": 2147483647,
            "name": "feature_name",
            "nullable": 1,
            "precision": 2147483647,
            "scale": 0,
            "schemaName": "information_schema",
            "tableName": "sql_features",
            "typeName": "character_data"
        }
    ]
}
```

次の AWS CLI コマンドは、テーブルを記述するクラスターに対して SQL ステートメントを実行します。この例では、一時的な認証情報認証方法を使用します。

```
aws redshift-data describe-table
    --db-user myuser
    --cluster-identifier mycluster-test
    --database dev
    --schema information_schema
    --table sql_features
```

次は、レスポンスの例です。

```
{
    "ColumnList": [
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": false,
            "length": 2147483647,
            "name": "feature_id",
            "nullable": 1,
            "precision": 2147483647,
            "scale": 0,
            "schemaName": "information_schema",
            "tableName": "sql_features",
```

```
            "typeName": "character_data"
        },
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": false,
            "length": 2147483647,
            "name": "feature_name",
            "nullable": 1,
            "precision": 2147483647,
            "scale": 0,
            "schemaName": "information_schema",
            "tableName": "sql_features",
            "typeName": "character_data"
        },
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": false,
            "length": 2147483647,
            "name": "sub_feature_id",
            "nullable": 1,
            "precision": 2147483647,
            "scale": 0,
            "schemaName": "information_schema",
            "tableName": "sql_features",
            "typeName": "character_data"
        },
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": false,
            "length": 2147483647,
            "name": "sub_feature_name",
            "nullable": 1,
            "precision": 2147483647,
            "scale": 0,
            "schemaName": "information_schema",
            "tableName": "sql_features",
            "typeName": "character_data"
        },
        {
            "isCaseSensitive": false,
            "isCurrency": false,
```

```
            "isSigned": false,
            "length": 2147483647,
            "name": "is_supported",
            "nullable": 1,
            "precision": 2147483647,
            "scale": 0,
            "schemaName": "information_schema",
            "tableName": "sql_features",
            "typeName": "character_data"
        },
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": false,
            "length": 2147483647,
            "name": "is_verified_by",
            "nullable": 1,
            "precision": 2147483647,
            "scale": 0,
            "schemaName": "information_schema",
            "tableName": "sql_features",
            "typeName": "character_data"
        },
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": false,
            "length": 2147483647,
            "name": "comments",
            "nullable": 1,
            "precision": 2147483647,
            "scale": 0,
            "schemaName": "information_schema",
            "tableName": "sql_features",
            "typeName": "character_data"
        }
    ]
}
```

クラスター内のデータベースの一覧表示

`aws redshift-data list-databases` AWS CLI コマンドを使用して、クラスター内のデータベースを一覧表示します。

次の AWS CLI コマンドは、クラスターに対して SQL ステートメントを実行し、データベースを一覧表示します。この例では、AWS Secrets Manager の認証方法を使用します。

```
aws redshift-data list-databases

    --secret-arn arn:aws:secretsmanager:us-west-2:123456789012:secret:myuser-secret-
hKgPWn
    --cluster-identifier mycluster-test
    --database dev
```

次は、レスポンスの例です。

```
{
    "Databases": [
        "dev"
    ]
}
```

次の AWS CLI コマンドは、クラスターに対して SQL ステートメントを実行し、データベースを一覧表示します。この例では、一時的な認証情報認証方法を使用します。

```
aws redshift-data list-databases
    --db-user myuser
    --cluster-identifier mycluster-test
    --database dev
```

次は、レスポンスの例です。

```
{
    "Databases": [
        "dev"
    ]
}
```

データベース内のスキーマの一覧表示

`aws redshift-data list-schemas` AWS CLI コマンドを使用して、データベース内のスキーマを一覧表示します。

次の AWS CLI コマンドは、クラスターに対して SQL ステートメントを実行し、データベース内のスキーマを一覧表示します。この例では、AWS Secrets Manager の認証方法を使用します。

```
aws redshift-data list-schemas
    --secret-arn arn:aws:secretsmanager:us-west-2:123456789012:secret:myuser-secret-
hKgPWn
    --cluster-identifier mycluster-test
    --database dev
```

次は、レスポンスの例です。

```
{
    "Schemas": [
        "information_schema",
        "pg_catalog",
        "pg_internal",
        "public"
    ]
}
```

次の AWS CLI コマンドは、クラスターに対して SQL ステートメントを実行し、データベース内のスキーマを一覧表示します。この例では、一時的な認証情報認証方法を使用します。

```
aws redshift-data list-schemas
    --db-user mysuser
    --cluster-identifier mycluster-test
    --database dev
```

次は、レスポンスの例です。

```
{
    "Schemas": [
        "information_schema",
        "pg_catalog",
        "pg_internal",
        "public"
    ]
}
```

データベース内のテーブルの一覧表示

`aws redshift-data list-tables` AWS CLI コマンドを使用して、データベース内のテーブルを一覧表示します。

次の AWS CLI コマンドは、クラスターに対して SQL ステートメントを実行し、データベース内のテーブルを一覧表示します。この例では、AWS Secrets Manager の認証方法を使用します。

```
aws redshift-data list-tables
    --secret-arn arn:aws:secretsmanager:us-west-2:123456789012:secret:myuser-secret-
hKgPWn
    --cluster-identifier mycluster-test
    --database dev
    --schema information_schema
```

次は、レスポンスの例です。

```
{
    "Tables": [
        {
            "name": "sql_features",
            "schema": "information_schema",
            "type": "SYSTEM TABLE"
        },
        {
            "name": "sql_implementation_info",
            "schema": "information_schema",
            "type": "SYSTEM TABLE"
        }
}
```

次の AWS CLI コマンドは、クラスターに対して SQL ステートメントを実行し、データベース内のテーブルを一覧表示します。この例では、一時的な認証情報認証方法を使用します。

```
aws redshift-data list-tables

     --db-user myuser
     --cluster-identifier mycluster-test
```

```
    --database dev
    --schema information_schema
```

次は、レスポンスの例です。

```
{
    "Tables": [
        {
            "name": "sql_features",
            "schema": "information_schema",
            "type": "SYSTEM TABLE"
        },
        {
            "name": "sql_implementation_info",
            "schema": "information_schema",
            "type": "SYSTEM TABLE"
        }
    ]
}
```

Amazon Redshift Data API のトラブルシューティングに関する問題

Data API に関する問題のトラブルシューティングには、「一般的なエラーメッセージ」というタイトルの以下のセクションを使用します。

トピック

- クエリのパケットが大きすぎる
- データベース応答がサイズ制限を超えている

クエリのパケットが大きすぎる

クエリのパケットが大きすぎることを示すエラーが表示された場合は、通常、ロードに対して返される結果セットが大きすぎます。Data API のサイズ制限は、データベースから返る結果セットの 1 行あたり 64 KB です。

この問題を解決するには、結果セットの各行が 64 KB 以下であることを確認します。

データベース応答がサイズ制限を超えている

データベース応答がサイズ制限を超えていることを示すエラーが表示される場合、通常、データベースから返る結果セットのサイズが大きすぎます。Data API の制限は、データベースより返る結果セットで 500 MB です。

この問題を解決するには、Data API への呼び出しで返るデータが 500 MB 以下になるようにします。500 MB を超えるデータを返す必要がある場合は、クエリで `LIMIT` 句を使用して、複数のステートメント呼び出しを行います。

Amazon EventBridge を使用した Amazon Redshift Data API オペレーションのスケジューリング

選択したイベントを照合し、ターゲットにルーティングしてアクションを実行するためのルールを作成できます。ルールを使用して、あらかじめ決められたスケジュールに従ってアクションを実行することもできます。詳細については、Amazon EventBridge ユーザーガイドを参照してください。

EventBridge でデータ API オペレーションをスケジュールするには、関連付けられた IAM ロールが CloudWatch Events (events.amazonaws.com) のプリンシパルを信頼する必要があります。このロールには、管理対象ポリシー `AmazonEventBridgeFullAccess` に相当するものがアタッチされている必要があります。また、Data API によって管理される `AmazonRedshiftDataFullAccess` ポリシー権限も必要です。IAM コンソールで、これらのアクセス許可を持つ IAM ロールを作成できます。IAM コンソールでロールを作成するときは、CloudWatch Events 用に AWS サービスの信頼できるエンティティを選択します。EventBridge ターゲットの `RoleArn` JSON 値に IAM ロールを指定します。IAM ロールの作成の詳細については、IAM ユーザーガイドの「AWS のサービス用ロールの作成 (コンソール)」を参照してください。

Amazon EventBridge で作成するルールの `name` は、`RedshiftDataParameters` の `StatementName` と一致する必要があります。

以下の例は、単一または複数の SQL ステートメントを使用し、データウェアハウスとして Amazon Redshift クラスターまたは Amazon Redshift Serverless ワークグループを使用して EventBridge ルールを作成するバリエーションを示しています。

単一 SQL ステートメントとクラスターによる呼び出し

次の例では、AWS CLI を使用して、Amazon Redshift クラスターに対する 1 つの SQL ステートメントの実行に使用する EventBridge ルールを作成します。

```
aws events put-rule
--name test-redshift-cluster-data
--schedule-expression "rate(1 minute)"
```

次に、ルールで指定されたスケジュールで実行する EventBridge ターゲットが作成されます。

```
aws events put-targets
--cli-input-json file://data.json
```

入力 data.json ファイルは次のとおりです。Sql JSON キーは、単一 SQL ステートメントがあることを示しています。Arn JSON 値にはクラスター識別子が含まれます。RoleArn JSON 値には、前述のように SQL を実行するために使用される IAM ロールが含まれます。

```
{
    "Rule": "test-redshift-cluster-data",
    "EventBusName": "default",
    "Targets": [
        {
            "Id": "2",
            "Arn": "arn:aws:redshift:us-east-1:123456789012:cluster:mycluster",
            "RoleArn": "arn:aws:iam::123456789012:role/Administrator",
            "RedshiftDataParameters": {
                "Database": "dev",
                "DbUser": "root",
                "Sql": "select 1;",
                "StatementName": "test-redshift-cluster-data",
                "WithEvent": true
            }
        }
    ]
}
```

単一 SQL ステートメントおよびワークグループによる呼び出し

次の例では、AWS CLI を使用して、Amazon Redshift Serverless ワークグループに対する 1 つの SQL ステートメントの実行に使用する EventBridge ルールを作成します。

```
aws events put-rule
--name  test-redshift-serverless-workgroup-data
--schedule-expression "rate(1 minute)"
```

次に、ルールで指定されたスケジュールで実行する EventBridge ターゲットが作成されます。

```
aws events put-targets
--cli-input-json file://data.json
```

入力 data.json ファイルは次のとおりです。Sql JSON キーは、単一 SQL ステートメントがあることを示しています。Arn JSON 値にはワークグループ名が含まれます。RoleArn JSON 値には、前述のように SQL を実行するために使用される IAM ロールが含まれます。

```
{
    "Rule": "test-redshift-serverless-workgroup-data",
    "EventBusName": "default",
    "Targets": [
        {
            "Id": "2",
            "Arn": "arn:aws:redshift-serverless:us-east-1:123456789012:workgroup/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "RoleArn": "arn:aws:iam::123456789012:role/Administrator",
            "RedshiftDataParameters": {
                "Database": "dev",
                "Sql": "select 1;",
                "StatementName": "test-redshift-serverless-workgroup-data",
                "WithEvent": true
            }
        }
    ]
}
```

複数の SQL ステートメントおよびクラスターによる呼び出し

次の例では、AWS CLI を使用して、Amazon Redshift クラスターに対する複数の SQL ステートメントの実行に使用する EventBridge ルールを作成します。

```
aws events put-rule
--name  test-redshift-cluster-data
--schedule-expression "rate(1 minute)"
```

次に、ルールで指定されたスケジュールで実行する EventBridge ターゲットが作成されます。

```
aws events put-targets
--cli-input-json file://data.json
```

入力 data.json ファイルは次のとおりです。Sqls JSON キーは、複数の SQL ステートメントがあることを示しています。Arn JSON 値にはクラスター識別子が含まれます。RoleArn JSON 値には、前述のように SQL を実行するために使用される IAM ロールが含まれます。

```
{
    "Rule": "test-redshift-cluster-data",
    "EventBusName": "default",
    "Targets": [
        {
            "Id": "2",
            "Arn": "arn:aws:redshift:us-east-1:123456789012:cluster:mycluster",
            "RoleArn": "arn:aws:iam::123456789012:role/Administrator",
            "RedshiftDataParameters": {
                "Database": "dev",
                "Sqls": ["select 1;", "select 2;", "select 3;"],
                "StatementName": "test-redshift-cluster-data",
                "WithEvent": true
            }
        }
    ]
}
```

複数の SQL ステートメントおよびワークグループによる呼び出し

次の例では、AWS CLI を使用して、Amazon Redshift Serverless ワークグループに対する複数の SQL ステートメントの実行に使用する EventBridge ルールを作成します。

```
aws events put-rule
--name  test-redshift-serverless-workgroup-data
--schedule-expression "rate(1 minute)"
```

次に、ルールで指定されたスケジュールで実行する EventBridge ターゲットが作成されます。

```
aws events put-targets
--cli-input-json file://data.json
```

入力 data.json ファイルは次のとおりです。Sqls JSON キーは、複数の SQL ステートメントがあることを示しています。Arn JSON 値にはワークグループ名が含まれます。RoleArn JSON 値には、前述のように SQL を実行するために使用される IAM ロールが含まれます。

```
{
```

```
    "Rule": "test-redshift-serverless-workgroup-data",
    "EventBusName": "default",
    "Targets": [
        {
            "Id": "2",
            "Arn": "arn:aws:redshift-serverless:us-east-1:123456789012:workgroup/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "RoleArn": "arn:aws:iam::123456789012:role/Administrator",
            "RedshiftDataParameters": {
                "Database": "dev",
                "Sqls": ["select 1;", "select 2;", "select 3;"],
                "StatementName": "test-redshift-serverless-workgroup-data",
                "WithEvent": true
            }
        }
    ]
}
```

データ API のモニタリング

モニタリングは、Data API およびその他の AWS ソリューションの信頼性、可用性、およびパフォーマンスを維持する上で重要な部分です。Data API を監視したり、問題が発生したときに報告したり、必要に応じて自動アクションを実行するため、AWSには次のモニタリングツールが用意されています。

- Amazon EventBridge を使用して、AWSのサービスを自動化し、アプリケーションの可用性の問題やリソースの変更などのシステムイベントに自動的に対応できます。AWS サービスからのイベントは、ほぼリアルタイムで EventBridge に配信されます。簡単なルールを記述して、注目するイベントと、イベントがルールに一致した場合に自動的に実行するアクションを指定できます。詳細については、Amazon EventBridge ユーザーガイドを参照してください。
- AWS CloudTrail は、AWS アカウントにより、またはそのアカウントに代わって行われた、API コールおよび関連イベントを取得し、指定した Amazon S3 バケットにログファイルを配信します。AWS を呼び出したユーザーとアカウント、呼び出し元の IP アドレス、および呼び出し日時を特定できます。Amazon Redshift と AWS CloudTrail の統合の詳細については、「CloudTrail による口グ記録」を参照してください。CloudTrail の詳細については、「AWS CloudTrail ユーザーガイド」を参照してください。

トピック

- Amazon EventBridge での Amazon Redshift データ API のイベントのモニタリング

Amazon EventBridge での Amazon Redshift データ API のイベントのモニタリング

お客様独自のアプリケーション、Software as a Service (SaaS) アプリケーション、AWSサービスからのリアルタイムデータのストリームを配信する EventBridge で、Data API イベントをモニタリングできます。EventBridge では、データは AWS Lambda や Amazon SNS などのターゲットにルーティングされます。これらのイベントは、CloudWatch Events に表示されるイベントと同じで、AWSリソースの変更を記述するシステムイベントのストリームをほぼリアルタイムで配信します。イベントは、Amazon Redshift データベースを含むアカウントに送信されます。例えば、別のアカウントでロールを引き受ける場合、イベントはそのアカウントに送信されます。詳細については、Amazon EventBridge ユーザーガイドの「Amazon EventBridge events」を参照してください。

データ API イベントは、`ExecuteStatement`または `BatchExecuteStatement` API オペレーションが `WithEvent` オプションを `true` に設定するときに送信されます。`state` フィールドには次のいずれかの値が含まれます。

- ABORTED – クエリの実行がユーザーによって停止されました。
- FAILED – クエリの実行に失敗しました。
- FINISHED – クエリの実行が終了しました。

イベントは保証に基づいて配信されます。詳細については、Amazon EventBridge ユーザーガイドの「Events from AWS services」を参照してください。

データ API 終了イベントの例

以下の例は、`ExecuteStatement`API オペレーションが終了したときの Data API のイベントを示しています。この例では、`test.testtable`という名前のステートメントの実行が終了しました。

```
{
    "version": "0",
    "id": "18e7079c-dd4b-dd64-caf9-e2a31640dab0",
    "detail-type": "Redshift Data Statement Status Change",
    "source": "aws.redshift-data",
    "account": "123456789012",
    "time": "2020-10-01T21:14:26Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:redshift:us-east-1:123456789012:cluster:redshift-cluster-1"
    ],
    "detail": {
        "principal": "arn:aws:iam::123456789012:user/myuser",
```

```
        "statementName": "test.testtable",
        "statementId": "dd2e1ec9-2ee3-49a0-819f-905fa7d75a4a",
        "redshiftQueryId": -1,
        "state": "FINISHED",
        "rows": 1,
        "expireAt": 1601673265
    }
}
```

Amazon Redshift Data API で AWS KMS を使用する

Amazon Redshift クラスターまたは Redshift Serverless ワークグループをカスタマーマネージドキーで暗号化すると、Amazon Redshift Data API は同じカスタマーマネージドキーを使用してクエリと結果を保存および暗号化します。

Data API は、デフォルトでデータを暗号化して、クエリテキストやクエリ結果などの機密情報を保護します。この保護には、AWS が所有する AWS KMS 暗号化キーを使用します。

保管中のデータをデフォルトで暗号化して、機密データの保護に伴う運用上のオーバーヘッドと複雑な作業を軽減します。このアプローチにより、厳格な暗号化のコンプライアンスと規制要件に対応する安全なアプリケーションを構築できます。

AWS KMS での権限の使用

Data API でカスタマーマネージドキーを使用するには権限が必要です。

カスタマーマネージドキーで暗号化されたクラスターに対して `ExecuteStatement` または `BatchExecuteStatement` を呼び出すと、Amazon Redshift は AWS KMS に `CreateGrant` リクエストを送信してユーザーに代わって権限を作成します。AWS KMS は権限を使用して、アカウント内の KMS キーへのアクセス権を Data API に付与します。

DATA API では、以下のオペレーションでカスタマーマネージドキーを使用するのに権限が必要です。

- `Encrypt` リクエストを AWS KMS に送信して、カスタマーマネージドキーを使用してクエリメタデータを暗号化する。
- `GenerateDataKey` リクエストを AWS KMS に送信して、カスタマーマネージドキーで暗号化されたデータキーを生成する。
- `Decrypt` リクエストを AWS KMS に送信して、暗号化されたデータキーを復号して、データを暗号化できるようにする。

権限へのアクセス許可の取り消し、またはカスタマーマネージドキーへの Amazon Redshift のアクセス許可の削除は、いつでも実行できます。これを行うと、DATA API はカスタマーマネージドキーによって暗号化されたデータにアクセスできなくなり、そのデータに依存しているオペレーションが影響を受けます。例えば、権限の取り消し後にクエリ結果を取得したり、クエリのステータスを追跡しようとすると、Data API は `AccessDeniedException` を返します。

カスタマーマネージドキーの KMS キーポリシー

キーポリシーは、カスタマーマネージドキーへのアクセスを制御します。すべてのカスタマーマネージドキーには、キーポリシーが 1 つだけ必要です。このポリシーには、そのキーを使用できるユーザーとその使用方法を決定するステートメントが含まれています。キーポリシーは、カスタマーマネージドキーの作成時に指定できます。詳細については、「AWS Key Management Service デベロッパーガイド」の「カスタマーマネージドキー」を参照してください。

Data API でカスタマーマネージドキーを使用するには、まず Amazon Redshift へのアクセスを許可する必要があります。キーポリシーで次の API オペレーションを許可する必要があります。

- `kms:CreateGrant` - カスタマーマネージドキーに許可を追加します。この権限は、指定された AWS KMS キーへのアクセスを制御します。これにより、必要な許可オペレーションに対し Amazon Redshift がアクセスできるようにします。詳細については、「AWS KMS での権限の使用」を参照してください。

以下は、キーポリシーの例です。

```
"Statement":[
   {
      "Sid":"Allow access to principals authorized to use Amazon Redshift",
      "Effect":"Allow",
      "Principal":{
         "AWS":"*"
      },
      "Action":[
         "kms:DescribeKey",
         "kms:CreateGrant"
      ],
      "Resource":"*",
      "Condition":{
         "StringEquals":{
            "kms:ViaService":"redshift.amazonaws.com",
            "kms:CallerAccount":"111122223333"
         }
```

```
      }
   },
   {
      "Sid":"AllowKeyAdministratorsAccess",
      "Effect":"Allow",
      "Principal":{
         "AWS":"arn:aws:iam::111122223333:role/ExampleAdminRole"
      },
      "Action":"kms:*",
      "Resource":"*"
   },
   {
      "Sid":"AllowKeyUseForExampleRole",
      "Effect":"Allow",
      "Principal":{
         "AWS":"arn:aws:iam::111122223333:role/ExampleUserRole"
      },
      "Action":[
         "kms:Encrypt",
         "kms:Decrypt",
         "kms:ReEncrypt*",
         "kms:GenerateDataKey*",
         "kms:DescribeKey"
      ],
      "Resource":"*"
   }
]
```

データ API 暗号化コンテキスト

暗号化コンテキストとは、データに関する追加のコンテキスト情報を含む、(オプションの) キーと値のペアのセットです。AWS KMS は、暗号化コンテキストを追加の認証済みデータとして使用し、認証付き暗号化をサポートします。データの暗号化リクエストに暗号化コンテキストを組み込むと、AWS KMS は暗号化コンテキストを暗号化後のデータにバインドします。データを復号するには、そのリクエストに同じ暗号化コンテキストを含める必要があります。

Data API は、プロビジョニングされたクラスターのすべての AWS KMS 暗号化オペレーションで同じ 3 つの暗号化コンテキストのキーと値のペアを使用します。

- `aws:redshift:arn` – クラスターの Amazon リソースネーム (ARN) (文字列)
- `aws:redshift:createtime` – クラスターの作成をリクエストしたときのタイムスタンプ
- `serviceName` – `RedshiftDataAPI`

```
"EncryptionContextSubset": {
    "aws:redshift:arn": "arn:aws:redshift:us-east-1:123456789012:cluster:redshift-
cluster",
    "aws:redshift:createtime": "20250815T0000Z",
    "serviceName": "RedshiftDataAPI",
}
```

Data API は、サーバーレスワークグループのすべての AWS KMS 暗号化オペレーションで 2 つの暗号化コンテキストのキーと値のペアを使用します。

- `aws:redshift-serverless:arn` – 名前空間の Amazon リソースネーム (ARN)
- `serviceName` – RedshiftDataAPI

```
"EncryptionContextSubset": {
    "aws:redshift-serverless:arn": "arn:aws:redshift-serverless:us-
east-1:123456789012:namespace:12345678-1234-1234-1234-123456789012",
    "serviceName": "RedshiftDataAPI"
}
```

暗号化の詳細については、「AWS KMS の暗号化の詳細の概要」を参照してください。Amazon Redshift と AWS KMS 統合の詳細については、「Amazon Redshift が AWS KMS を使用する方法」を参照してください。

Amazon Sagemaker Unified Studio を使用して Amazon Redshift と SageMaker Lakehouse のデータベースをクエリする

Amazon SageMaker Unified Studio は、オフコンソールの開発環境を提供し、SageMaker Lakehouse、Amazon Redshift、および SQL 分析用の Amazon Athena のデータに対する SQL 分析をサポートします。管理者の URL を使用して Amazon SageMaker Unified Studio に移動し、SSO または AWS 認証情報を使用してログインします。最初のプロジェクトの設定の詳細については、「Amazon SageMaker Unified Studio ユーザーガイド」の「Getting started」を参照してください。

Amazon SageMaker Unified Studio では、クエリエディタを使用して Amazon Redshift と Amazon Athena を実行して SQL 分析を実行できます。クエリエディタを使用して、クエリの記述と実行、結果の表示、チームとの作業の共有ができます。AWS アカウント (同じアカウント内および他の AWS アカウント 間で) の Redshift データウェアハウスに対してクエリを実行し、同じインターフェイスを使用して Redshift と Athena の両方の SQL クエリを構築し、Amazon Managed Workflows for

Apache Airflow を使用して SQL クエリをスケジュールします。Amazon Q 生成 SQL を使用して、自然言語から SQL を生成することもできます。

Amazon Redshift パラメータグループを作成します。

Amazon Redshift で、作成する各クラスターにパラメータグループを関連付けます。パラメータグループは、クラスター内で作成したデータベースに適用されるパラメータのグループです。これらのパラメータは、クエリのタイムアウトやデータスタイルなどのデータベース設定を定義します。クラスターを起動する場合、パラメータグループに関連付ける必要があります。パラメータグループを後で変更する場合は、クラスターを変更し、別のパラメータグループを選択します。

各パラメータグループには、データベースの設定を定義する複数のパラメータがあります。使用できるパラメータのリストは、パラメータグループが属しているパラメータグループファミリーによって異なります。デフォルトのパラメータグループファミリーは `redshift-2.0` です。

Amazon Redshift は、パラメータグループファミリーごとに 1 つのデフォルトパラメータグループを提供しています。デフォルトパラメータグループにはパラメータごとのプリセット値があり、これを変更することはできません。デフォルトのパラメータグループ名の形式は `default.`*`parameter_group_family`* です。例えば、`redshift-2.0` パラメータグループファミリーのデフォルトのパラメータグループは、`default.redshift-2.0` です。

デフォルトパラメータグループとは異なるパラメータ値を使用するには、カスタムパラメータグループを作成してクラスターを関連付ける必要があります。カスタムパラメータグループ内のパラメータ値は、最初はデフォルトパラメータグループ内のものと同じです。これらの値は Amazon Redshift によってプリセットされているため、すべてのパラメータの最初の `source` は `engine-default` です。パラメータ値を変更した後、`source` は `user` に変わり、値がデフォルト値から変更されたことを示します。

> **Note**
>
> Amazon Redshift コンソールには各パラメータの `source` は表示されません。`source` を表示するには、Amazon Redshift API、AWS CLI、または AWS SDK の 1 つを使用する必要があります。

ユーザーが作成したパラメータグループについては、パラメータ値はいつでも変更できます。またはすべてのパラメータ値をデフォルトにリセットすることもできます。また、異なるパラメータグループをクラスターに関連付けることもできます。場合によっては、クラスターに既に関連付けられているパラメータグループのパラメータ値を変更したり、別のパラメータグループをクラスターに関連付けたりすることもできます。このような場合、更新したパラメータ値を有効にするためにクラスター

の再起動が必要になることがあります。クラスターに障害が発生し、Amazon Redshift により再起動された場合、変更内容はそのときに適用されます。メンテナンス中にクラスターが再起動された場合、変更は適用されません。詳細については、「WLM の動的プロパティと静的プロパティ」を参照してください。

デフォルトパラメータ値

> **Note**
>
> 2025 年 1 月 10 日現在、`require_ssl` パラメータのデフォルト値は true です。クラスターで SSL を必須にしない場合は、クラスターの作成時にカスタムパラメータグループを使用するか、デフォルト設定でクラスターを作成した後で、カスタムパラメータグループに関連付けるように、クラスターを変更することができます。

次の表はデフォルトパラメータ値の一覧とともに、各パラメータの詳細な情報へのリンクを示しています。これらは `redshift-2.0` パラメータグループファミリーのデフォルト値です。

パラメータ名	値	詳細情報
auto_analyze	true	Amazon Redshift データベースデベロッパーガイドの「auto_analyze」
auto_mv	true	Amazon Redshift データベースデベロッパーガイドの「自動マテリアライズドビュー」
datestyle	ISO, MDY	Amazon Redshift データベースデベロッパーガイドの「datestyle」
enable_case_sensitive_identifier	false	Amazon Redshift データベースデベロッパーガイドの「enable_case_sensitive_identifier」
enable_user_activity_logging	false	データベース監査ログ作成このガイドの「」
extra_float_digits	0	Amazon Redshift データベースデベロッパーガイドの「extra_float_digits」

パラメータ名	値	詳細情報
max_concurrency_scaling_clusters	1	Amazon Redshift データベースデベロッパーガイドの「max_concurrency_scaling_clusters」
query_group	default	Amazon Redshift データベースデベロッパーガイドの「query_group」
require_ssl	true	接続のセキュリティオプションを設定するこのガイドの「」
search_path	$user, public	Amazon Redshift データベースデベロッパーガイドの「search_path」
statement_timeout	0	Amazon Redshift データベースデベロッパーガイドの「statement_timeout」
wlm_json_configuration	[{"auto_wlm":true}]	ワークロード管理このガイドの「」
use_fips_ssl	false	システムが FIPS に準拠する必要がある場合のみ、FIPS 準拠の SSL モードを有効にします。

> **Note**
>
> `max_cursor_result_set_size` パラメータは廃止されました。カーソル結果セットのサイズの詳細については、Amazon Redshift データベースデベロッパーガイドの「カーソルの制約」を参照してください。

データベースで SET コマンドを使用して一時的にパラメータを上書きすることもできます。SET コマンドは、現在のセッションの期間だけパラメータを上書きします。前の表で示されたパラメータに加えて、データベースで `wlm_query_slot_count` を設定することで、一時的にスロットカウントを調整することもできます。`wlm_query_slot_count` パラメータは、パラメータグループでの設定に使用することはできません。スロット数の調整の詳細については、Amazon Redshift データベースデベロッパーガイドの「wlm_query_slot_count」を参照してください。他のパラメータを一時的に上書きする方法の詳細については、Amazon Redshift データベースデベロッパーガイドから「サーバー設定の変更」を参照してください。

ワークロード管理

Amazon Redshift では、ワークロード管理 (WLM) を使用して、使用可能なクエリキューの数と、処理するためにクエリをそれらのキューにルーティングする方法を定義します。WLM は、パラメータグループ設定の一部です。クラスターは、関連付けられているパラメータグループで指定された WLM 設定を使用します。

パラメータグループを作成すると、デフォルトの WLM 設定には、最大 5 つのクエリを同時に実行できる 1 つのキューが含まれます。クエリの処理をさらに細かく制御する場合、キューを追加して、各キューの WLM プロパティを設定できます。追加した各キューのデフォルトの WLM 設定は、そのプロパティを設定するまで同じです。

付加的なキューを追加する場合、設定内の最後のキューがデフォルトキューになります。クエリが WLM 設定の条件に基づいて別のキューにルーティングされる場合を除いて、デフォルトキューにより処理されます。デフォルトキューのモードおよび同時実行レベル (クエリスロット) を指定できますが、デフォルトキューのユーザーグループやクエリグループを指定することはできません。

他のパラメータと同様、デフォルトパラメータグループで WLM 設定を変更することはできません。デフォルトパラメータグループに関連付けられたクラスターは、常にデフォルトの WLM 設定を使用します。WLM 設定を変更するには、新しいパラメータグループを作成してからカスタム WLM 設定を必要とするクラスターとそのパラメータグループを関連付けます。

WLM の動的プロパティと静的プロパティ

WLM 設定プロパティは動的または静的のいずれかです。動的なプロパティは、クラスターを再起動することなくデータベースに適用できますが、静的プロパティで変更を有効にするには、クラスターの再起動が必要です。静的および動的プロパティの詳細については、「WLM の動的設定プロパティと静的設定プロパティ」を参照してください。

WLM 設定パラメータのプロパティ

WLM は、Amazon Redshift コンソール、AWS CLI、Amazon Redshift API、または AWS SDK の 1 つを使用して設定できます。WLM 設定では、キュー間でのメモリ割り当て、キューで同時に実行できるクエリの数などのキューの動作を定義するいくつかのプロパティが使用されます。

> **Note**
>
> 以下のプロパティは Amazon Redshift コンソール名とともに表示され、対応する JSON プロパティは説明で示されています。

次の表は、プロパティが自動 WLM または手動 WLM に適用可能かどうかをまとめたものです。

WLM プロパティ	自動 WLM	手動 WLM
自動 WLM	はい	はい
ショートクエリアクセラレーションの有効化	はい	はい
ショートクエリの最大実行時間	はい	はい
優先度	はい	いいえ
キューのタイプ	はい	はい
キュー名	はい	はい
同時実行スケーリングモード	はい	はい
Concurrency	いいえ	はい
ユーザーグループ	はい	はい
User Group Wildcard	はい	はい
クエリグループ	はい	はい
Query Group Wildcard	はい	はい
ユーザーロール	可能	はい
ユーザーロールワイルドカード	可能	はい
タイムアウト	いいえ	廃止
メモリ	いいえ	はい
クエリのモニタリングルール	はい	はい

次の一覧では、設定できる WLM プロパティについて説明します。

自動 WLM

自動 WLM を `true` に設定すると、自動 WLM が有効になります。[Automatic WLM] (自動 WLM) では、[Concurrency on main] (メインでの同時実行数) と [Memory (%)] (メモリ (%))の値が `Auto` に設定されます。Amazon Redshift は、クエリの同時実行数とメモリの割り当てを管理します。デフォルトは `true` です。

JSON プロパティ: `auto_wlm`

ショートクエリアクセラレーションの有効化

ショートクエリアクセラレーション (SQA) は、実行時間が短い一部のクエリを、実行時間が長いクエリよりも優先します。SQA では実行時間が短いクエリを専用領域で実行します。このため SQA クエリは、実行時間が長いクエリをキューで待機するよう強制されません。SQA によって実行時間が短いクエリの実行開始が早くなり、ユーザーへの結果表示も早くなります。SQA を有効にすると、ショートクエリの最大実行時間も指定できます。SQA を有効にするには、`true` を指定します。デフォルトは `false` です。この設定は、キューではなく各パラメータグループに適用されます。

JSON プロパティ: `short_query_queue`

ショートクエリの最大実行時間

SQA を有効にした場合は、0 に指定することで、WLM でショートクエリの最大実行時間を動的に設定することができます。または、固定値 (1 20 秒) を指定します。デフォルト値は `0` です。

JSON プロパティ: `max_execution_time`

優先度

優先度は、キューで実行されるクエリの優先度を設定します。優先度を設定するには、[WLM モード] を [Auto WLM] に設定する必要があります。つまり、`auto_wlm` が `true` でなければなりません。有効な値は、`highest`、`high`、`normal`、`low`、および `lowest` です。デフォルトは `normal` です。

JSON プロパティ: `priority`

キューのタイプ

キュータイプは、[自動 WLM] または [手動 WLM] で使用されるキューを指定します。`queue_type` を `auto` または `manual` に設定します。指定されなかった場合、デフォルト値は `manual` です。

JSON プロパティ: queue_type

キュー名

キューの名前。キュー名は、ビジネスニーズに基づいて設定できます。キュー名は WLM 設定内で一意である必要があります。また、最大 64 文字の英数字、アンダースコア、またはスペースを使用します。引用符を含めることはできません。たとえば、ETL クエリのキューがある場合は、ETL queue と名前を付けることができます。この名前は、メトリクス、システムテーブルの値、Amazon Redshift コンソールでキューを識別するために使用されます。これらのソースの名前を使用するクエリとレポートは、名前の変更に対応できる必要があります。以前は、キュー名は Amazon Redshift によって生成されていました。キューのデフォルト名は、Queue 1、Queue 2 の順に最後の Default queue という名前のキューまで続きます。

> **Important**
>
> キュー名を変更すると、WLM キューメトリクスの QueueName ディメンションの値 (WLMQueueLength、WLMQueueWaitTime、WLMQueriesCompletedPerSecond、WLMQueryDura など) も変更されます。そのため、キュー名を変更する場合は、設定した CloudWatch のアラームを変更する必要がある場合があります。

JSON プロパティ: name

同時実行スケーリングモード

キューで同時実行スケーリングを有効にするには、[同時実行スケーリングモード] を auto に設定します。キューにルーティングされたクエリの数がキューに設定された同時実行数を超えると、対象クエリがスケーリングクラスターに送信されます。スロットが使用可能になると、クエリはメインクラスターで実行されます。デフォルトは off です。

JSON プロパティ: concurrency_scaling

同時実行

手動 WLM キューで同時に実行できるクエリの数。このプロパティは、手動 WLM にのみ適用されます。同時実行スケーリングが有効な場合、キューが同時実行数レベル (クエリスロット) に達すると、対象のクエリがスケーリングクラスターに送信されます。同時実行スケーリングモードが無効な場合、スロットが使用可能になるまで、クエリはキューで待機します。範囲は 1～50 です。

JSON プロパティ: query_concurrency

ユーザーグループ

ユーザーグループ名のカンマ区切りリスト。ユーザーグループのメンバーがデータベースでクエリを実行すると、そのメンバーのクエリはユーザーグループに関連付けられたキューにルーティングされます。

JSON プロパティ: `user_group`

ユーザーグループのワイルドカード

ユーザーグループでワイルドカードを有効にするかどうかを示すブール値。この値が 0 の場合、ワイルドカードは無効です。この値が 1 の場合、ワイルドカードは有効です。ワイルドカードを有効にすると、「*」または「?」を使用できます。クエリを実行するときに複数のユーザーグループを指定します。詳細については、ワイルドカードを参照してください。

JSON プロパティ: `user_group_wild_card`

クエリグループ

クエリグループのカンマ区切りリスト。クエリグループのメンバーがデータベースでクエリを実行すると、そのメンバーのクエリはクエリグループに関連付けられたキューにルーティングされます。

JSON プロパティ: `query_group`

クエリグループのワイルドカード

クエリグループでワイルドカードを有効にするかどうかを示すブール値。この値が 0 の場合、ワイルドカードは無効です。この値が 1 の場合、ワイルドカードは有効です。ワイルドカードを有効にすると、「*」または「?」を使用できます。クエリを実行するときに複数のクエリグループを指定します。詳細については、ワイルドカードを参照してください。

JSON プロパティ: `query_group_wild_card`

ユーザーロール

ユーザーロールのカンマ区切りリスト。そのユーザーロールのメンバーがデータベースでクエリを実行すると、そのメンバーのクエリはユーザーロールに関連付けられたキューにルーティングされます。ユーザーロールの詳細については、「ロールベースのアクセスコントロール (RBAC)」を参照してください。

JSON プロパティ: `user_role`

ユーザーロールワイルドカード

クエリグループでワイルドカードを有効にするかどうかを示すブール値。この値が 0 の場合、ワイルドカードは無効です。この値が 1 の場合、ワイルドカードは有効です。ワイルドカードを有効にすると、「*」または「？」を使用できます。クエリを実行するときに複数のクエリグループを指定します。詳細については、ワイルドカードを参照してください。

JSON プロパティ: `user_role_wild_card`

タイムアウト (ms)

WLM タイムアウト (`max_execution_time`) は廃止されました。自動 WLM を使用している場合は使用できません。代わりに、`query_execution_time` を使用してクエリモニタリングルール (QMR) を作成して、経過したクエリ実行時間を制限します。詳細については、「WLM クエリモニタリングルール」を参照してください。

クエリがキャンセルされるまで実行できる最大時間 (ミリ秒単位)。場合によっては、読み取り専用クエリ (SELECT ステートメントなど) が WLM タイムアウトに伴ってキャンセルされることもあります。このような場合、WLM は、WLM キュー割り当てルールに基づいて次に一致するキューへのクエリのルーティングを試みます。クエリが他のいずれのキュー定義にも一致しない場合、クエリはキャンセルされ、デフォルトキューには割り当てられません。詳細については、「WLM クエリキューのホッピング」を参照してください。WLM タイムアウトは `returning` 状態に達したクエリには適用されません。クエリの状態を表示するには、STV_WLM_QUERY_STATE システムテーブルを参照してください。

JSON プロパティ: `max_execution_time`

メモリ (%)

キューに割り当てるメモリの割合。少なくとも 1 つキューに対してメモリの割合を指定する場合は、他のすべてのキューに対しても割合を指定する必要があります。合計は 100 パーセント以下とします。すべてのキューのメモリ割り当ての合計が 100% 未満である場合、未割り当て分のメモリはサービスによって管理されます。サービスは、この未割り当てメモリを、処理用メモリの追加をリクエストしたキューに一時的に割り当てることができます。

JSON プロパティ: `memory_percent_to_use`

クエリのモニタリングルール

WLM クエリモニタリングルールを使用することで、指定した基準または述語に基づいてクエリの WLM キューを継続的にモニタリングできます。たとえば、システムリソースを過剰に消費す

る傾向があるクエリをモニタリングし、指定したパフォーマンスの境界をクエリを超えたときに指定のアクションを開始することができます。

> **Note**
>
> プログラムでルールを作成する場合は、パラメータグループ定義に含める JSON をコンソールを使用して生成することを強くおすすめします。

特定のクエリキューにクエリモニタリングルールを関連付けます。キューごとの 25 個までルールを指定できます。すべてのキューでのルールは合計 25 個までです。

JSON プロパティ: `rules`

JSON プロパティの階層:

```
rules
    rule_name
    predicate
        metric_name
        operator
        value
    action
        value
```

ルールごとに、以下のプロパティを指定します。

- `rule_name` – ルール名は WLM 設定内で一意である必要があります。ルール名には最大で 32 文字の英数字または下線を使用できます。スペースまたは疑問符を含めることはできません。
  - `predicate` – 述語は、ルールごとに最大で 3 つまで設定できます。述語ごとに、以下のプロパティを指定します。
    - `metric_name` – メトリクスの一覧については、Amazon Redshift データベースデベロッパーガイドの「クエリモニタリングのメトリクス」を参照してください。
    - `operator` – オペレーションは =、<、および > です。
    - `value` – アクションをトリガーする指定されたメトリクスのしきい値。
- `action` – 各ルールは 1 つのアクションに関連付けられます。有効なアクションは以下のとおりです。
  - `log`
  - hop (手動 WLM でのみ利用可能)

- abort
- change_query_priority (自動 WLM でのみ利用可能)

次の例は、2 つの述語と rule_1 いうアクションを持つ hop という名前の WLM クエリモニタリングルール用の JSON を示しています。

```
"rules": [
          {
            "rule_name": "rule_1",
            "predicate": [
              {
                "metric_name": "query_execution_time",
                "operator": ">",
                "value": 100000
              },
              {
                "metric_name": "query_blocks_read",
                "operator": ">",
                "value": 1000
              }
            ],
            "action": "hop"
          }
        ]
```

これらのプロパティとクエリキュー設定の戦略に関する詳細については、Amazon Redshift データベースデベロッパーガイドからワークロード管理の実装を参照してください。

AWS CLI を使用した WLM パラメータの設定

WLM を設定するには、wlm_json_configuration パラメータを変更します。wlm_json_configuration プロパティ値の最大サイズは 8,000 文字です。この値は JavaScript Object Notation (JSON) でフォーマットされています。AWS CLI、Amazon Redshift API、または AWS SDK の 1 つを使用して WLM を設定する場合、このセクションの残りの部分で wlm_json_configuration パラメータの JSON 構造を構築する方法を確認してください。

> **Note**
>
> Amazon Redshift を使用して WLM を設定する場合、コンソールでは簡単にキューを追加してそのプロパティを設定できるため、JSON 形式について理解する必要はありません。コン

ソールを使用して WLM を設定する方法の詳細については、「パラメータグループを変更する」を参照してください。

例

次の例は、自動 WLM で 1 つのキューを定義するデフォルトの WLM 設定です。

```
{
   "auto_wlm": true
}
```

例

次の例は、同時実行数レベル (クエリスロット) が 5 である 1 つの手動 WLM キューを定義するカスタムの WLM 設定です。

```
{
   "query_concurrency":5
}
```

構文

デフォルトの WLM 設定は非常にシンプルで、キューが 1 つとプロパティが 1 つしかありません。JSON 構造でキューを追加し、キューごとに複数のプロパティを設定することができます。次の構文は、複数のプロパティを持つ複数のキューを設定する場合に使用する JSON 構造を表しています。

```
[
   {
      "ParameterName":"wlm_json_configuration", "ParameterValue":
         "[
             {
                "q1_first_property_name":"q1_first_property_value",
                "q1_second_property_name":"q1_second_property_value",
                ...
             },
             {
                "q2_first_property_name":"q2_first_property_value",
                "q2_second_property_name":"q2_second_property_value",
```

```
                ...
            }
            ...

        ]"
    }
]
```

前の例では、q1 で始まる代表的なプロパティは、最初のキューの配列にあるオブジェクトです。これらの各オブジェクトは名前と値のペアです。name と value を組み合わせて最初のキューに WLM プロパティを設定します。q2 で始まる代表的なプロパティは、2 番目のキューの配列にあるオブジェクトです。より多くのキューが必要な場合、追加の各キューに別のアレイを追加し、各オブジェクトのプロパティを設定します。

WLM の設定を変更する際は、キュー内の 1 つのプロパティだけを変更する場合であっても、キューの構造全体を含める必要があります。これは、wlm_json_configuration パラメータの値として、JSON 構造全体が文字列として渡されるためです。

AWS CLI コマンドのフォーマット

wlm_json_configuration を使用する場合、AWS CLI パラメータには特定の形式が必要です。使用する形式は、クライアントのオペレーティングシステムによって異なります。コマンドラインから正しく渡されるように JSON 構造を囲む方法は、オペレーティングシステムによってさまざまです。Linux、macOS、Windows の各オペレーティングシステムで適切なコマンドを作成する方法の詳細については、以下のセクションを参照してください。一般的な AWS CLI での JSON データ構造の違いについては、AWS Command Line Interface ユーザーガイドの「Quoting strings」を参照してください。

例

以下のコマンド例では、example-parameter-group という名前のパラメータグループの手動 WLM を設定します。この設定により、ショートクエリアクセラレーションでショートクエリの最大実行時間が 0 に設定されます。これにより、値を動的に設定することを WLM に指示します。ApplyType 設定は dynamic です。この設定では、このパラメータ内の動的プロパティに対して行われたすべての変更は、設定に対して他の静的な変更が行われていない限り、速やかに適用されます。この設定では、次の 3 つのキューを定義しています。

- 最初のキューにより、ユーザーは自身のキューで report をラベルとして (query_group プロパティで) 指定し、そのキューへのクエリのルーティングをスムーズにすることができま

す。report* ではワイルドカード検索が有効にされているため、クエリをキューにルーティングするためにラベルを正しい順序にする必要はありません。たとえば、reports と reporting のいずれも、このクエリグループに一致します。このキューにはすべてのキューの合計メモリのうち 25 パーセントが割り当てられ、最大 4 つのクエリを同時に実行できます。クエリの時間は、最大 20,000 ミリ秒 (ms) に制限されています。モードが自動に設定されているため、キューのクエリスロットがいっぱいになると、対象のクエリがスケーリングクラスターに送信されます。

- 2 番目のキューにより、データベース内の admin または dba グループのメンバーであるユーザーは、自身のクエリを処理用にそのキューにルーティングすることができます。ユーザーグループではワイルドカード検索が無効になっているため、ユーザーは自身のクエリがそのキューにルーティングされるようにデータベース内のグループと正確に一致する必要があります。このキューにはすべてのキューの合計メモリのうち 40% が割り当てられ、最大 5 つのクエリを同時に実行できます。モードはオフに設定されているため、admin または dba グループのメンバーによって送信されたクエリはすべて、メインクラスターで実行されます。
- 設定内の最後のキューはデフォルトキューです。このキューにはすべてのキューの合計メモリのうち 35 パーセントが割り当てられ、最大 5 つのクエリを同時に処理できます。モードは自動に設定されます。

> **Note**
>
> デモンストレーションの目的で、例は複数行で示されています。実際のコマンドに改行を入れることはできません。

```
aws redshift modify-cluster-parameter-group
--parameter-group-name example-parameter-group
--parameters
'[
  {
    "query_concurrency": 4,
    "max_execution_time": 20000,
    "memory_percent_to_use": 25,
    "query_group": ["report"],
    "query_group_wild_card": 1,
    "user_group": [],
    "user_group_wild_card": 0,
    "user_role": [],
    "user_role_wild_card": 0,
    "concurrency_scaling": "auto",
```

```
    "queue_type": "manual"
  },
  {
    "query_concurrency": 5,
    "memory_percent_to_use": 40,
    "query_group": [],
    "query_group_wild_card": 0,
    "user_group": [
      "admin",
      "dba"
    ],
    "user_group_wild_card": 0,
    "user_role": [],
    "user_role_wild_card": 0,
    "concurrency_scaling": "off",
    "queue_type": "manual"
  },
  {
    "query_concurrency": 5,
    "query_group": [],
    "query_group_wild_card": 0,
    "user_group": [],
    "user_group_wild_card": 0,
    "user_role": [],
    "user_role_wild_card": 0,
    "concurrency_scaling": "auto",
    "queue_type": "manual"
  },
  {"short_query_queue": true}
]'
```

以下に、自動 WLM 設定で、WLM クエリモニタリングルールを設定する例を示します。この例では、example-monitoring-rules という名前のパラメータグループを作成します。設定で前の例と同じ 3 つのキューを定義しますが、query_concurrency および memory_percent_to_use は、指定されません。この設定では、次のルールとクエリの優先度も追加されます。

- 最初のキューは rule_1 というルールを定義します。ルールには 2 つの述語があります。query_cpu_time > 10000000 と query_blocks_read > 1000 です。ルールのアクションは log です。このキューの優先度は Normal です。
- 2 つめのキューは rule_2 というルールを定義します。ルールには 2 つの述語があります。query_execution_time > 600000000 と scan_row_count > 1000000000 です。ルールのアクションは abort です。このキューの優先度は Highest です。

- 設定内の最後のキューはデフォルトキューです。このキューの優先度は Low です。

> **Note**
>
> デモンストレーションの目的で、例は複数行で示されています。実際のコマンドに改行を入れることはできません。

```
aws redshift modify-cluster-parameter-group
--parameter-group-name example-monitoring-rules
--parameters
'[ {
  "query_group" : [ "report" ],
  "query_group_wild_card" : 1,
  "user_group" : [ ],
  "user_group_wild_card" : 0,
  "user_role": [ ],
  "user_role_wild_card": 0,
  "concurrency_scaling" : "auto",
  "rules" : [{
    "rule_name": "rule_1",
    "predicate": [{
      "metric_name": "query_cpu_time",
      "operator": ">",
      "value": 1000000 },
      { "metric_name": "query_blocks_read",
      "operator": ">",
      "value": 1000
    } ],
    "action" : "log"
  } ],
   "priority": "normal",
   "queue_type": "auto"
}, {
  "query_group" : [ ],
  "query_group_wild_card" : 0,
  "user_group" : [ "admin", "dba" ],
  "user_group_wild_card" : 0,
  "user_role": [ ],
  "user_role_wild_card": 0,
  "concurrency_scaling" : "off",
```

```
  "rules" : [ {
    "rule_name": "rule_2",
    "predicate": [
      {"metric_name": "query_execution_time",
      "operator": ">",
      "value": 600000000},
      {"metric_name": "scan_row_count",
      "operator": ">",
      "value": 1000000000}],
      "action": "abort"}],
   "priority": "high",
   "queue_type": "auto"

}, {
  "query_group" : [ ],
  "query_group_wild_card" : 0,
  "user_group" : [ ],
  "user_group_wild_card" : 0,
  "user_role": [ ],
  "user_role_wild_card": 0,
  "concurrency_scaling" : "auto",
  "priority": "low",
  "queue_type": "auto",
  "auto_wlm": true
}, {
  "short_query_queue" : true
} ]'
```

JSON ファイルを指定してコマンドラインで AWS CLI を使用して WLM を設定する

`wlm_json_configuration` を使用して AWS CLI パラメータを変更し、JSON ファイルとして `parameters` 引数の値で渡すことができます。

```
aws redshift modify-cluster-parameter-group --parameter-group-name
 myclusterparaametergroup --parameters file://modify_pg.json
```

`--parameters` の引数は、`modify_pg.json` に保存されます。ファイルの場所は、オペレーティングシステムの形式で指定されます。詳細については、「ファイルからパラメータをロードする」を参照してください。`modify_pg.json` JSON ファイルの内容の例を以下に示します。

```
[
```

```
    {
        "ParameterName": "wlm_json_configuration",
        "ParameterValue": "[{\"user_group\":\"example_user_group1\",\"query_group\":
 \"example_query_group1\", \"query_concurrency\":7},{\"query_concurrency\":5}]"
    }
]
```

```
[
    {
        "ParameterName": "wlm_json_configuration",
        "ParameterValue": "[{\"query_group\":[\"reports\"],\"query_group_wild_card\":0,
\"query_concurrency\":4,\"max_execution_time\":20000,\"memory_percent_to_use\":25},
{\"user_group\":[\"admin\",\"dba\"],\"user_group_wild_card\":1,\"query_concurrency\":5,
\"memory_percent_to_use\":40},{\"query_concurrency\":5,\"memory_percent_to_use\":35},
{\"short_query_queue\": true, \"max_execution_time\": 5000 }]",
        "ApplyType": "dynamic"
    }
]
```

Linux および macOS オペレーティングシステムのコマンドラインで AWS CLI を使用して WLM を設定する際のルール

1 行にパラメータを指定して AWS CLI コマンドを実行するには、以下のルールに従います。

- JSON 構造全体を単一引用符 (') と 1 組の角括弧 ([ ]) で囲む必要があります。
- すべてのパラメータ名とパラメータ値は、二重引用符 (") で囲む必要があります。
- ParameterValue 値内では、ネストされた構造全体を二重引用符 (") と角括弧 ([ ]) で囲む必要があります。
- ネストされた構造内では、各キューのプロパティと値をそれぞれ中括弧 ({ }) で囲む必要があります。
- ネストされた構造内では、各二重引用符 (") の前で円記号 (\) のエスケープ文字を使用する必要があります。
- 名前と値のペアでは、各プロパティとその値をコロン (:) で区切ります。
- 名前と値の各ペアは、カンマ (,) で区切ります。
- 複数のキューは、キューの中括弧の末尾 (}) とその次のキューの中括弧の先頭 ({) の間にカンマ (,) を置くことで区切ります。

Microsoft Windows オペレーティングシステムの Windows PowerShell で AWS CLI を使用して WLM を設定する際のルール

1 行にパラメータを指定して AWS CLI コマンドを実行するには、以下のルールに従います。

- JSON 構造全体を単一引用符 (') と 1 組の角括弧 ([ ]) で囲む必要があります。
- すべてのパラメータ名とパラメータ値は、二重引用符 (") で囲む必要があります。
- `ParameterValue` 値内では、ネストされた構造全体を二重引用符 (") と角括弧 ([ ]) で囲む必要があります。
- ネストされた構造内では、各キューのプロパティと値をそれぞれ中括弧 ({ }) で囲む必要があります。
- ネストされた構造内では、各二重引用符 (") とその円記号 (\) のエスケープ文字の前で円記号 (\) のエスケープ文字を使用する必要があります。この要件は、プロパティが正しく渡されるようにするため (\\\")、3 つの円記号と 1 つの二重引用符を使用することを意味します。
- 名前と値のペアでは、各プロパティとその値をコロン (:) で区切ります。
- 名前と値の各ペアは、カンマ (,) で区切ります。
- 複数のキューは、キューの中括弧の末尾 (}) とその次のキューの中括弧の先頭 ({) の間にカンマ (,) を置くことで区切ります。

Windows オペレーティングシステムでコマンドプロンプトを使用して WLM を設定する際のルール

1 行にパラメータを指定して AWS CLI コマンドを実行するには、以下のルールに従います。

- JSON 構造全体を二重引用符 (") と 1 組の角括弧 ([ ]) で囲む必要があります。
- すべてのパラメータ名とパラメータ値は、二重引用符 (") で囲む必要があります。
- `ParameterValue` 値内では、ネストされた構造全体を二重引用符 (") と角括弧 ([ ]) で囲む必要があります。
- ネストされた構造内では、各キューのプロパティと値をそれぞれ中括弧 ({ }) で囲む必要があります。
- ネストされた構造内では、各二重引用符 (") とその円記号 (\) のエスケープ文字の前で円記号 (\) のエスケープ文字を使用する必要があります。この要件は、プロパティが正しく渡されるようにするため (\\\")、3 つの円記号と 1 つの二重引用符を使用することを意味します。
- 名前と値のペアでは、各プロパティとその値をコロン (:) で区切ります。
- 名前と値の各ペアは、カンマ (,) で区切ります。

- 複数のキューは、キューの中括弧の末尾 (}) とその次のキューの中括弧の先頭 ({) の間にカンマ (,) を置くことで区切ります。

パラメータグループを作成する

デフォルトのパラメータグループとは異なるパラメータ値を設定する場合は、独自のパラメータグループを作成することができます。

パラメータグループを作成するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Configurations] (設定) を選択し、次に [Workload management] (ワークロード管理) を選択して [Workload management] (ワークロード管理) ページを表示します。
3. [作成] を選択して [パラメータグループの作成] ウィンドウを表示します。
4. [パラメータグループ名] と [説明] に値を入力します。
5. [作成] を選択して、パラメータグループを作成します。

パラメータグループを変更する

任意のパラメータグループを表示して、パラメータおよびワークロード管理 (WLM) の設定の値の概要を確認することができます。パラメータを変更してパラメータ設定や WLM 設定プロパティを変更できます。

> **Note**
>
> デフォルトのパラメータグループを変更することはできません。

AWS マネジメントコンソール

コンソールでは、グループパラメータは [パラメータ] タブに表示され、[ワークロードキュー] は [ワークロード管理] タブに表示されます。

パラメータグループを変更するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Configurations] (設定) を選択し、次に [Workload management] (ワークロード管理) を選択して [Workload management] (ワークロード管理) ページを表示します。
3. 変更するパラメータグループを選択して、[パラメータ] と [ワークロード管理] タブのある詳細ページを表示します。
4. [パラメータ] タブを選択して、現在のパラメータ設定を表示します。
5. [パラメータの編集] を選択して、次のパラメータの設定変更を有効化します。

   - auto_analyze
   - auto_mv
   - datestyle
   - enable_case_sensitive_identifier
   - enable_user_activity_logging
   - extra_float_digits
   - max_concurrency_scaling_clusters
   - max_cursor_result_set_size
   - query_group
   - require_ssl
   - search_path
   - statement_timeout
   - use_fips_ssl

   これらのパラメータの詳細については、Amazon Redshift パラメータグループを作成します。を参照してください。
6. 変更を入力して [保存] を選択し、パラメータグループを更新します。

パラメータグループの WLM 設定を変更するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Configurations] (設定) を選択し、次に [Workload management] (ワークロード管理) を選択して [Workload management] (ワークロード管理) ページを表示します。
3. 変更するパラメータグループを選択して、[パラメータ] と [ワークロード管理] タブのある詳細ページを表示します。
4. [ワークロード管理] タブを選択して現行の WLM 設定を表示します。
5. [ワークロードキューの編集] を選択して、WLM 設定を編集します。
6. (オプション) [Enable short query acceleration (ショートクエリアクセラレーションの有効化)] を選択して、ショートクエリアクセラレーション (SQA) を有効にします。

   SQA を有効にした場合、[ショートクエリの最大実行時間 (1 ~ 20 秒)] は、デフォルトで [動的] に設定されます。最大実行時間を固定値に設定するには、値 1 20 を選択します。
7. キュー設定を変更するには、次の 1 つ以上の操作を行います:

   - [Switch WLM mode (WLM モードの切り替え)] を選択し、[Automatic WLM (自動 WLM)] と [Manual WLM (手動 WLM)] の間で切り替えます。

     [Automatic WLM (自動 WLM)] では、[メモリ] と [Concurrency on main (メインでの同時実行数)] の値が [auto (自動)] に設定されます。
   - キューを作成するには、[Edit workload queues (ワークロードキューの編集)] を選択してから、[Add Queue (キューの追加)] を選択します。
   - キューを変更するには、テーブルのプロパティ値を変更します。キューのタイプに応じて、プロパティには以下が含まれます。
     - キュー名は変更できます。
     - メモリ
     - メインクラスターでの同時実行数
     - Concurrency scaling mode (同時実行スケーリングモード) (off または auto)
     - タイムアウト (ms)
     - ユーザーグループ
     - クエリグループ
     - ユーザーロール

これらのプロパティの詳細については、「WLM 設定パラメータのプロパティ」を参照してください。

> **Important**
>
> キュー名を変更すると、WLM キューメトリクスの QueueName ディメンションの値 (WLMQueueLength、WLMQueueWaitTime、WLMQueriesCompletedPerSecond、WLMQue など) も変更されます。そのため、キュー名を変更する場合は、設定した CloudWatch のアラームを変更する必要がある場合があります。

- キューの順序を変更するには、上 および下の矢印ボタンを選択します。
- キューを削除するには、テーブル内にあるキューの行の [削除] ボタンを選択します。

8. (オプション) [再起動するまで動的な変更を延期する] を選択して、次の再起動後に関連付けられたクラスターに変更を適用します。

> **Note**
>
> 一部の変更では、この設定に関係なくクラスターの再起動が必要となります。詳細については、WLM の動的プロパティと静的プロパティ を参照してください。

9. [保存] を選択します。

AWS CLI

AWS CLI を使用して Amazon Redshift パラメータを設定するには、特定のパラメータグループに対して modify-cluster-parameter-group コマンドを使用します。`parameter-group-name` で、変更するパラメータグループを指定します。`parameters` パラメータ (`modify-cluster-parameter-group` コマンドのパラメータ) を使用して、パラメータグループで変更する各パラメータの名前と値のペアを指定します。

> **Note**
>
> `wlm_json_configuration` を使用して AWS CLI パラメータを設定する際は、特別な考慮事項があります。このセクションの例は、`wlm_json_configuration` を除くすべ

てのパラメータに適用されます。wlm_json_configuration による AWS CLI の設定の詳細については、「ワークロード管理」を参照してください。

パラメータ値を変更した後は、変更したパラメータグループに関連付けられているクラスターを再起動する必要があります。値が適用中の間、applying のクラスター状態は ParameterApplyStatus と表示され、値の適用が完了すると pending-reboot になります。再起動後、クラスター内のデータベースは新しいパラメータ値の使用を開始します。クラスターの再起動の詳細については、「クラスターの再起動」を参照してください。

> **Note**
>
> wlm_json_configuration パラメータに含まれる一部のプロパティは動的プロパティであり、関連付けられたクラスターを再起動しなくても変更が適用されます。動的プロパティと静的プロパティの詳細については、「WLM の動的プロパティと静的プロパティ」を参照してください。

次の構文は、modify-cluster-parameter-group コマンドを使用してパラメータを設定する方法を示しています。*parameter_group_name* を指定し、*parameter_name* と *parameter_value* の両方を実際に設定するパラメータとパラメータ値に置き換えます。複数のパラメータを同時に変更する場合は、パラメータと値の各セットをスペースで区切ります。

```
aws redshift modify-cluster-parameter-group --parameter-group-
name parameter_group_name --parameters
 ParameterName=parameter_name,ParameterValue=parameter_value
```

次の例は、statement_timeout パラメータグループの enable_user_activity_logging パラメータと myclusterparametergroup パラメータを設定する方法を示しています。

> **Note**
>
> 読みやすくするため、例は複数行で表示されていますが、実際の AWS CLI では 1 行になります。

```
aws redshift modify-cluster-parameter-group
--parameter-group-name myclusterparametergroup
```

```
--parameters ParameterName=statement_timeout,ParameterValue=20000
 ParameterName=enable_user_activity_logging,ParameterValue=true
```

クエリモニタリングルールを作成する

Amazon Redshift コンソールを使用して、WLM クエリモニタリングルールを作成および変更できます。クエリモニタリングルールは、パラメータグループの WLM 構成パラメータの一部です。クエリモニタリングルール (QMR) を変更すると、クラスターを変更しなくても自動的に変更が行われます。詳細については、WLM クエリモニタリングルールを参照してください。

ルールを作成する際には、ルール名、述語 (複数可)、アクションを定義します。

ルールの含まれた WLM 構成を保存する際は、WLM 構成パラメータの JSON の一部として、ルール定義の JSON コードを表示できます。

クエリモニタリングルールを作成するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Configurations] (設定) を選択し、次に [Workload management] (ワークロード管理) を選択して [Workload management] (ワークロード管理) ページを表示します。
3. 変更するパラメータグループを選択して、[パラメータ] と [ワークロード管理] タブのある詳細ページを表示します。
4. [ワークロード管理] タブを選択し、[ワークロードキューの編集] を選択して WLM 設定を編集します。
5. 事前定義されたテンプレートを使って、あるいはゼロから新しいルールを追加します。

   事前定義されたテンプレートを使用するには、以下を実行します。

   1. [Query monitoring rules (クエリモニタリングルール)] グループの [Add rule from template (テンプレートからルールを追加)] を選択します。ルールテンプレートのリストが表示されます。
   2. 1 つ以上のルールテンプレートを選択します。[保存] を選択すると、WLM が選択したテンプレートごとにルールを 1 つ作成します。
   3. [ルール名]、[述語] および [アクション] などのルールの値を入力または確認します。

4. [Save] (保存) をクリックします。

ゼロから新しいルールを追加するには、以下を実行します。

1. 述語を追加するには、[Add predicate (述語を追加)] を選択します。述語は、ルールごとに最大で 3 つまで設定できます。述語がすべて満たされると、WLM は関連アクションをトリガーします。
2. [アクション] を選択します。各ルールにアクションを 1 つ設定します。
3. [Save] (保存) をクリックします。

Amazon Redshift は、JSON フォーマットで WLM 設定パラメータを生成し、[JSON] セクションに表示します。

パラメータグループを削除する

パラメータグループが必要なくなり、どのクラスターにも関連付けられていない場合は、削除できます。カスタムパラメータグループのみを削除できます。

パラメータグループを削除するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Configurations] (設定) を選択し、次に [Workload management] (ワークロード管理) を選択して [Workload management] (ワークロード管理) ページを表示します。
3. [パラメータグループ] で、変更するパラメータグループを選択します。

> **Note**
>
> デフォルトのパラメータグループは削除できません。

4. [削除] を選択し、パラメータグループを削除することを確認します。

Amazon Redshift クラスターを AWS パートナーと統合する

Amazon Redshift を使用すると、Amazon Redshift コンソールから AWS パートナーと統合できます。クラスターの詳細ページでは、AWS パートナーアプリケーションを使用して Amazon Redshift データウェアハウスへのデータのオンボーディングを高速化できます。クラスター内の既存のデータとともに、さまざまなソースからのデータを結合して分析することもできます。Informatica との統合を完了する前に、パートナーの IP アドレスをインバウンドトラフィックの許可リストに追加する必要があります。次の AWS パートナーは、Amazon Redshift と統合できます。

- Datacoral
- Etleap
- Fivetran
- SnapLogic
- Stitch
- Upsolver
- Matillion (プレビュー)
- Sisense (プレビュー)
- Thoughtspot

AWS パートナーは、AWS CLI または Amazon Redshift API オペレーションを使用して Amazon Redshift と統合できます。詳細については、Amazon Redshift API リファレンス、または AWS CLI コマンドリファレンスを参照してください。

クラスターを AWS パートナーと統合するには、次の手順を使用します。

Amazon Redshift クラスターを AWS パートナーと統合するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [クラスター] を選択します。
3. 使用する DB クラスターを選択します。
4. [Add partner integration (パートナ統合の追加)] を選択します。パートナーの選択 ページが開き、利用可能な AWS パートナーの詳細が表示されます。
5. AWS パートナーを選択してから、[次へ] を選択します。

統合するクラスターについての詳細とともに、選択した AWS パートナーのさらなる詳細が表示されます。[クラスターの詳細] セクションには、クラスター識別子、エンドポイント、データベース名、ユーザー名 (データベースユーザー名) など、AWS パートナーのウェブサイトで指定した情報が表示されます。この情報は、選択したパートナーに送信されます。

6. [パートナーの追加] を選択して、AWS パートナーのウェブサイトを開きます。
7. パートナーのウェブサイトで Amazon Redshift クラスターとの統合を設定します。パートナーのウェブサイトでは、Amazon Redshift クラスターにロードされるデータソースを選択して設定できます。また、追加の抽出、読み込み、変換 (ELT) 変換を定義して、ビジネスデータを処理したり、他のデータセットと結合したり、分析とレポート用の統合ビューを構築することもできます。

AWS パートナーの統合は、クラスターの詳細の [プロパティ] タブから表示および管理できます。[統合] セクションには、AWS パートナーのウェブサイトとのリンクに使用できるパートナー名、統合のステータス、データを受信するデータベース、およびクラスターを更新した可能性のある最新の成功した接続が一覧表示されます。

指定できる値は次のとおりです。

- アクティブ — AWS パートナーはクラスターに接続し、設定されたタスクを完了できます。
- 非アクティブ — AWS パートナーの統合は存在しません。
- ランタイム障害 — AWS パートナーはクラスターに接続できますが、設定されたタスクを完了できません。
- 接続エラー — AWS パートナーはクラスターに接続できません。

Amazon Redshift から AWS パートナーの統合を削除しても、データはクラスターに流入し続けます。パートナーのウェブサイトで削除を完了します。

AWS パートナーによるデータの読み込み

パートナーを Amazon Redshift クラスターと統合する以外に、パートナーのデータロードツールを使用して 30 を超えるソースから Amazon Redshift クラスターにデータを移動することもできます。それを行う前に、パートナーの IP アドレス (下記参照) をインバウンドルールの許可リストに追加する必要があります。Amazon EC2 セキュリティグループにルールを追加する方法の詳細については、「Amazon EC2 ユーザーガイド」の「インスタンスのインバウンドトラフィックの認可」を参

照してください。Informatica Data Loader ツールは無料ですが、選択したデータソースとターゲットによっては、データ入力料金が適用される場合があることに注意してください。

次のパートナーからデータをロードすることができます。

- Informatica

Amazon Redshift クラスターをパートナーと統合するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [AWS パートナー統合] を選択し、クラスターを統合するパートナーを選択します。
3. [Complete <partner-name> integration] (<partner-name> の完全な統合) を選択します。パートナーの統合サイトにリダイレクトされます。
4. パートナーのサイトで必要な詳細を入力し、統合を完了します。

リザーブドノード

AWS では、Amazon Redshift の使用のため発生する料金はコンピューティングノードに基づいています。各コンピューティングノードに対して、時間料金で課金されます。時間料金は、リージョン、ノードの種類、ノードに適用されるのがオンデマンドノード料金であるかリザーブドノード料金であるかなどの要因によって異なります。

オンデマンドノード料金は最も高価ですが、Amazon Redshift で最も柔軟性のあるオプションです。オンデマンドレートでは、実行中のクラスターにあるコンピューティングノードに対してのみ課金されます。クラスターをシャットダウンまたは削除すると、そのクラスターにあったコンピューティングノードに対して、それ以降は課金されません。使用するコンピューティングノードに対してのみ請求され、それ以外は課金されません。各コンピューティングノードに対して課金される時間料金は、リージョンやノードの種類などの要因によって異なります。

リザーブドノード料金はコンピューティングノードに対して、割引された時間単位の料金で課金されるため、オンデマンド料金よりも安価です。ただし、これらの割引料金の適用を受けるには、リザーブドノードのサービスを購入する必要があります。サービスを購入する際は、予約を行います。予約により、予約期間中に予約する各ノードについて割引料金が設定されます。サービスの割引料金は、リージョン、ノードの種類、期間、支払いオプションなどの要因によって異なります。

ノードをリザーブドノードとして指定するには、`PurchaseReservedNodeOffering` API オペレーションを呼び出すか、Amazon Redshift コンソールで [Purchase reserved nodes (リザーブドノードの購入)] を選択します。リザーブドノードを購入する場合、該当するリザーブドノードタイプの AWS リージョン、ノードタイプ、期間、ノード数、オファリングタイプを指定する必要があります。リザーブドノードは指定した AWS リージョンでのみ使用できます。

このトピックでは、リザーブドノードのサービスについてと、これらのサービスを購入して Amazon Redshift クラスターの実行コストを削減する方法について説明します。このトピックでは、一般的な用語としてオンデマンドレートまたは割引料金について説明し、料金の概念と、それによる課金への影響について理解できるようにします。特定のレートの詳細については、Amazon Redshift 料金表を参照してください。

リザーブドノードサービス

長期間にわたって Amazon Redshift クラスターを実行したままにする予定の場合は、リザーブドノードサービスの購入を検討してください。これらのサービスでは、オンデマンド料金に比べて大幅なコスト削減ができますが、コンピューティングノードを予約し、1 年または 3 年の期間にわたってこれらのノードの料金の支払いを確約する必要があります。

リザーブドノードは、ノードに対する課金レートを決定するために厳密に使用される課金の概念です。ノードを予約しても、実際にはノードは作成されません。使用量にかかわらず、リザーブドノードに対して課金されます。つまり、割引料金の適用対象となる実行中のクラスターにノードがあるかどうかにかかわらず、予約期間中に予約する各ノードについて支払いを行う必要があります。

プロジェクトの評価フェーズ、または概念実証を開発する場合、オンデマンド料金により、従量料金制の柔軟性を得ることができ、使用した分だけ支払い、クラスターをシャットダウンまたは削除することでいつでも支払いを停止することができます。本稼働環境のニーズが確立され、実装段階を開始したら、1 つ以上のサービスを購入して、コンピューティングノードの予約を検討してください。

サービスでは 1 つ以上のコンピューティングノードを適用できます。サービスを購入するときに、予約するコンピューティングノードの数を指定します。複数のコンピューティングノードの 1 つのサービスを購入するか、複数のサービスを購入し、各ノードのコンピューティングノードの特定の数を指定する選択ができます。

たとえば、次のいずれも、3 つのコンピューティングノードのサービスを購入するための有効な方法です。

- 1 つのサービスを購入し、3 つのコンピューティングノードを指定する。
- 2 つのサービスを購入し、最初のサービスに対して 1 つのコンピューティングノード、2 つ目のサービスに対して 2 つのコンピューティングノードを指定する。
- 3 つのサービスを購入し、サービスごとに 1 つのコンピューティングノードを指定する。

リザーブドノードサービス間の価格の比較

Amazon Redshift には、サービスについて複数の支払いオプションがあります。選択する支払いオプションは、支払いスケジュールと、予約に対して課金される割引料金に影響します。予約に対して支払う前払い料金が高いほど、全体的な削減額が大きくなります。

次の支払いオプションがサービスで利用できます。サービスは、オンデマンドレートに対する削減額が小さいものから大きいものの順に示されています。

> **Note**
>
> 予約に指定された期間中は、リザーブドノードを使用するかどうかにかかわらず、毎時間、該当する時間料金で課金されます。支払いオプションによって決定されるのは、支払いの頻度と、適用される割引だけです。詳細については、「リザーブドノードサービス」を参照してください。

お支払い方法	支払いスケジュール	比較の削減額	所要時間	前払い料金	繰り返しの月額料金
前払いなし	予約期間中の月次の支払い額。前払い料金はありません。	オンデマンド料金に対して約 20 パーセントの割引。	1 年間または 3 年間	なし	はい
一部前払い	予約の期間については部分的な前払い料金と月次の支払い額。	期間によって最大 41～73 パーセントの割引。	1 年間または 3 年間	はい	はい
全前払い	予約の完全な前払い料金。月額料金なし。	期間によって最大 42～76 パーセントの割引。	1 年間または 3 年間	はい	なし

特定のオプションと期間は空き状況により異なります。

> **Note**
>
> 以前に Amazon Redshift の [重度使用] サービスを購入した場合、同等のサービスには [一部前払い] サービスがあります。

リザーブドノードの動作

リザーブドノードサービスでは、前のセクションで説明したように、支払い条件に応じて支払いが発生します。すでに実行中のクラスターがあるか、予約後にクラスターを起動するかどうかにかかわらず、この方法で支払いを行います。

サービスを購入すると、予約が処理されるまで予約のステータスは [支払い保留中] になります。予約の処理に失敗した場合、ステータスは [支払い失敗] と表示され、プロセスを再試行することができます。予約が正常に処理されると、ステータスは [アクティブ] になります。予約の該当する割引料金は、ステータスが [アクティブ] になるまで適用されません。予約期間が経過すると、ステータスは [リタイア] になりますが、履歴の目的で予約に関する情報に引き続きアクセスすることができ

ます。予約が [リタイア] になると、クラスターは引き続き実行されますが、ノードに割引価格を適用する別の予約がない限り、オンデマンドレートで課金される場合があります。

リザーブドノードは、サービスを購入するリージョンに固有です。Amazon Redshift コンソールを使用してサービスを購入する場合は、サービスを購入する AWS リージョンを選択し、予約プロセスを完了します。サービスをプログラムで購入する場合、リージョンは接続先の Amazon Redshift エンドポイントによって決まります。Amazon Redshift のリージョンについては、Amazon Web Services 全般のリファレンスの「リージョンとエンドポイント」を参照してください。

クラスターを起動するときに割引料金がすべてのノードに適用されるようにするには、リージョン、ノードの種類、および選択するノードの数が、1 つ以上のアクティブな予約に一致することを確認します。それ以外の場合、アクティブな予約に一致しないノードについては、オンデマンドレートで課金されます。

実行中のクラスターで、予約したノードの数を超えた場合は、それらの追加のノードに対して、オンデマンドレートで課金が発生し続けます。この発生は、予約したノードの数に応じて、同じクラスターのノードに対してさまざまなレートで課金される場合があることを示します。別のサービスを購入して、これらの追加のノードに対応することができます。その場合、予約のステータスが [アクティブ] になると、残りの期間は割引料金がノードに適用されます。

別のノードの種類にクラスターのサイズを変更し、その種類のノードを予約していない場合は、オンデマンドレートで課金されます。サイズ変更したクラスターに対して割引料金の適用を受けたい場合は、新しいノードの種類の別のサービスを購入できます。ただし、期間が経過するまで、元の予約についても引き続き支払いが発生します。期間の終了前に予約を変更する必要がある場合は、AWS コンソールを使用してサポートケースを作成してください。

> **Note**
>
> コンソールには、使用中および未使用のリザーブドノードの数が表示されます。ただし、コンソールには、現在のユーザーアカウントが使用しているノードのみが使用中の数として表示されます。同じ支払いアカウントで別のユーザーアカウントが使用しているノードは、コンソールに未使用として表示されます。
>
> 例
>
> - 支払いアカウントが予約しているノード数が 20 個
> - 現在のユーザーアカウントが使用しているノード数が 6 個
> - 同じ支払いアカウントで別のユーザーアカウントが使用しているノード数も 6 個

この例では、コンソールに 6 個のノードのみが使用中として表示され、14 個のノードは未使用として表示されます。

リザーブドノードと一括請求 (コンソリデーティッドビリング)

購入アカウントが、1 つの一括請求の支払いアカウントに請求される一連のアカウントの一部である場合、リザーブドノードの料金面でのメリットを広範囲に利用できます。すべてのサブアカウントの時間単位の使用量が月次で支払いアカウントに集約されます。さまざまな役割を持つチームやグループがある企業にとっては特に便利です。したがって、請求書の計算には通常のリザーブドノードの口ジックが適用されます。詳細については、AWS Billing ユーザーガイドの「Consolidated Billing」を参照してください。

リザーブドインスタンスを使用するクラスターのノードタイプを変更する前に考慮すべき事項

サイズ変更機能 (伸縮自在なサイズ変更または従来のサイズ変更) を使用してノードファミリーまたはノードサイズを変更することで、クラスターのノードタイプを変更できます。ノードタイプを変更しても、クラスターに関連付けられているリザーブドノードの購入は自動的に変更されません。既存の予約は元のノードタイプに関連付けられたままになります。これらの予約については、期間が終了するまで引き続き請求されます。一方、新しいタイプのノードはオンデマンド料金で請求される場合があります。

このため、リザーブドインスタンスの変更については、リザーブドノードを使用するクラスターのノードタイプを変更する前に、AWS アカウントチームまたは AWS 請求サポート に問い合わせることをお勧めします。担当者が適切にご案内します。

リザーブドノードの例

このセクションのシナリオでは、次の予約の詳細を使用し、オンデマンドレートと割引料金に基づいて料金がどのように発生するかを示します。

- リージョン: 米国西部 (オレゴン)
- ノードタイプ: ra3.xlplus

- 支払い方法: 前払い料金なし
- 期間: 1 年
- リザーブドノード数: 16

例 1

20 個のノードがある米国西部 (オレゴン) リージョンに、1 つのクラスターがある。

このシナリオでは、16 個のノードで予約から割引料金が適用されますが、クラスターのその他の 4 つのノードはオンデマンドレートで課金されます。

例 2

12 個のノードがある米国西部 (オレゴン) リージョンに、1 つのクラスターがある。

このシナリオでは、クラスターの 12 個のノードにはすべて、予約から割引料金が適用されます。ただし、現在それらの適用対象となる実行中のクラスターがなくても、予約の残りのリザーブドノードに対して支払いが発生します。

例 3

12 個のノードがある米国西部 (オレゴン) リージョンに、1 つのクラスターがある。この設定ではクラスターを数か月実行し、その後でクラスターにノードを追加する必要があります。クラスターのサイズを変更し、同じノードの種類を選択して、合計 16 個のノードを指定します。

このシナリオでは、16 個のノードに対して割引料金が請求されます。クラスターにあるノードの数は予約したノード数と等しいため、料金は 1 年間は同じままです。

例 4

16 個のノードがある米国西部 (オレゴン) リージョンに、1 つのクラスターがある。この設定ではクラスターを数か月実行し、その後でノードを追加する必要があります。クラスターのサイズを変更し、同じノードの種類を選択して、合計 20 個のノードを指定します。

このシナリオでは、サイズ変更の前のすべてのノードに対して割引料金で課金されます。サイズ変更後は、1 年の残りの期間は 16 個のノードに対して割引料金で課金され、クラスターに追加した 4 つの追加のノードについてはオンデマンドレートで課金されます。

例 5

米国西部 (オレゴン) リージョンに 2 つクラスターがある。クラスターの 1 つは 6 個のノードを持ち、他のクラスターは 10 個のノードを持つ。

このシナリオでは、両方のクラスターのノードの合計数は予約したノードの数と等しいため、すべてのノードに対して割引料金で課金されます。

例 6

米国西部 (オレゴン) リージョンに 2 つクラスターがある。クラスターの 1 つは 4 個のノードを持ち、他のクラスターは 6 個のノードを持つ。

このシナリオでは、実行中のクラスターの 10 個のノードに対して割引料金で課金され、割引料金の適用対象となる実行中のクラスターが現在なくても、予約したその他の 6 個のノードに対して割引料金で支払いが発生します。

リザーブドノードの購入

AWS マネジメントコンソール または AWS CLI を使用してリザーブドノードサービスを購入し、現在および過去の予約を表示できます。

AWS マネジメントコンソール

リザーブドノードを購入するには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Clusters] (クラスター) を選択し、次に [Reserved nodes] (リザーブドノート) を選択してリザーブドノートのリストを表示します。
3. [リザーブドノートの購入] を選択して、購入するノードのプロパティを選択するためのページが表示されます。
4. ノードのプロパティを入力してから、[リザーブドノートの購入] を選択します。

サービスを購入した後、[リザーブドノード] リストには、各予約とその詳細 (ノードタイプ、ノード数、予約状況など) が表示されます。予約の詳細のさらなる情報については、「リザーブドノードの動作」を参照してください。

リザーブドノードをアップグレードするには、AWS CLI を使用します。

すべてのノードタイプをリザーブドノードに変換することはできません。また、既存のリザーブドノードを更新できない可能性もあります。これは、ノードタイプが廃止されたことが原因である可能性があります。廃止されたノードタイプを更新するには、カスタマーサポートにお問い合わせください。

AWS CLI

AWS CLI を使用してリザーブドノードの予約をアップグレードするには

1. 支払いタイプ、期間、および条件の要件を満たすサービスに関する ReservedNodeOfferingID のリストを取得します。このステップを、次の例に示します。

```
aws redshift get-reserved-node-exchange-offerings --reserved-node-id xxxxxxxx-
xxxx-xxxx-xxxx-xxxxxxxxxxxx
{
    "ReservedNodeOfferings": [
        {
            "Duration": 31536000,
            "ReservedNodeOfferingId": "yyyyyyyy-yyyy-yyyy-yyyy-yyyyyyyyyyyy",
            "UsagePrice": 0.0,
            "NodeType": "dc2.large",
            "RecurringCharges": [
                {
                    "RecurringChargeFrequency": "Hourly",
                    "RecurringChargeAmount": 0.2
                }
            ],
            "CurrencyCode": "USD",
            "OfferingType": "No Upfront",
            "ReservedNodeOfferingType": "Regular",
            "FixedPrice": 0.0
        }
    ]
}
```

2. `accept-reserved-node-exchange` を呼び出し、前のステップで取得した ReservedNodeOfferingID とともに交換する DC1 リザーブドノードの ID を指定します。

   このステップを、次の例に示します。

```
aws redshift accept-reserved-node-exchange --reserved-node-id xxxxxxxx-xxxx-
xxxx-xxxx-xxxxxxxxxxxx --target-reserved-node-offering-id yyyyyyyy-yyyy-yyyy-
yyyy-yyyyyyyyyyyy
{
    "ExchangedReservedNode": {
        "UsagePrice": 0.0,
        "OfferingType": "No Upfront",
        "State": "exchanging",
        "FixedPrice": 0.0,
        "CurrencyCode": "USD",
        "ReservedNodeId": "zzzzzzzz-zzzz-zzzz-zzzz-zzzzzzzzzzzz",
        "NodeType": "dc2.large",
        "NodeCount": 1,
        "RecurringCharges": [
            {
                "RecurringChargeFrequency": "Hourly",
                "RecurringChargeAmount": 0.2
            }
        ],
        "ReservedNodeOfferingType": "Regular",
        "StartTime": "2018-06-27T18:02:58Z",
        "ReservedNodeOfferingId": "yyyyyyyy-yyyy-yyyy-yyyy-yyyyyyyyyyyyy",
        "Duration": 31536000
    }
}
```

describe-reserved-nodes を呼び出し、Node type の値を確認して、交換が完了したことを確認できます。

Amazon Redshift のセキュリティ

AWS では、クラウドのセキュリティが最優先事項です。セキュリティを最も重視する組織の要件を満たすために構築された「AWS」のデータセンターとネットワークアーキテクチャは、お客様に大きく貢献します。

セキュリティは、「AWS」と顧客の間の責任共有です。責任共有モデルでは、この責任がクラウドのセキュリティおよびクラウド内のセキュリティとして説明されています。

- クラウドのセキュリティ - AWS は、AWS クラウドで AWS サービスを実行するインフラストラクチャを保護する責任を負います。また、AWS は、使用するサービスを安全に提供します。セキュリティの有効性は、AWS コンプライアンスプログラムの一環として、サードパーティーの審査機関によって定期的にテストおよび検証されています。Amazon Redshift に適用されるコンプライアンスプログラムについては、「コンプライアンスプログラムによる AWS 対象範囲内のサービス」を参照してください。
- クラウド内のセキュリティ – お客様の責任は使用する AWS のサービスによって決まります。また、お客様は、お客様のデータの機密性、組織の要件、および適用可能な法律および規制などの他の要因についても責任を担います。

Amazon Redshift リソースへのアクセスは 4 つのレベルで制御されます。

- クラスターの管理 – クラスターを作成、設定、削除できるかどうかは、AWS セキュリティ認証情報と関連するユーザーやアカウントに付与したアクセス許可で制御します。適切なアクセス許可が付与されたユーザーは、AWS マネジメントコンソール、AWS Command Line Interface (CLI)、または Amazon Redshift アプリケーションプログラミングインターフェイス (API) を使用して、クラスターを管理できます。このアクセスは、 ポリシーを使用して管理されます。

  > **Important**
  >
  > Amazon Redshift には、アクセス許可、ID、および安全なアクセスを管理するためのベストプラクティス集が用意されています。Amazon Redshift を使い始めるにあたり、これらに慣れておくことをお勧めします。詳細については、「Amazon Redshift での Identity and Access Management」を参照してください。

- クラスター接続性 – Amazon Redshift セキュリティグループにより、クラスレスドメイン間ルーティング (CIDR) 形式での Amazon Redshift クラスターへの接続が許可される AWS インスタンスが指定されます。Amazon Redshift、Amazon EC2、Amazon VPC セキュリティグループを作成

し、それらをクラスターに関連付ける方法については、Amazon Redshift セキュリティグループを参照してください。

- データベースアクセス – テーブルやビューなどのデータベースオブジェクトにアクセスできるかどうかは、Amazon Redshift データベースのデータベースユーザーアカウントで制御します。ユーザーは、ユーザーアカウントがアクセス権限を付与されたデータベースのリソースにのみアクセス可能です。これらの Amazon Redshift ユーザーアカウントを作成してアクセス許可を管理するには、CREATE USER、CREATE GROUP、GRANT、REVOKE SQL ステートメントを使用します。詳細については、「Amazon Redshift データベースデベロッパーガイド」の「データベースセキュリティの管理」を参照してください。
- 一時的データベース認証情報およびシングルサインオン – CREATE USER や ALTER USER などの SQL コマンドを使用したデータベースユーザーの作成と管理に加えて、カスタム Amazon Redshift JDBC または ODBC ドライバーで SQL クライアントを設定できます。これらのドライバーは、データベースのログオンプロセスの一部として、データベースユーザーや仮パスワードを作成するプロセスを管理します。

  ドライバーは AWS Identity and Access Management (IAM) 認証に基づいてデータベースユーザーを認証します。既に AWS の外部でユーザー ID を管理している場合、Security Assertion Markup Language (SAML) 2.0 に準拠した ID プロバイダー (IdP) を使用して、Amazon Redshift リソースへのアクセスを管理できます。IAM ロールを使用して、フェデレーティッドユーザーが一時データベース認証情報を生成して Amazon Redshift データベースにログオンすることを許可するよう IdP および AWS を設定できます。詳細については、「IAM 認証を使用したデータベースユーザー認証情報の生成」を参照してください。

このドキュメントは、Amazon Redshift を使用する際の責任共有モデルの適用方法を理解するのに役立ちます。以下のトピックでは、セキュリティおよびコンプライアンスの目的を達成するように Amazon Redshift を設定する方法について説明します。また、Amazon Redshift リソースのモニタリングや保護に役立つ他の AWS サービスの使用方法についても説明します。

トピック

- Amazon Redshift におけるデータ保護
- Amazon Redshift での Identity and Access Management
- AWS Secrets Manager を使用した Amazon Redshift 管理者パスワードの管理
- Amazon Redshift でのログ作成とモニタリング
- Amazon Redshift のコンプライアンス検証
- Amazon Redshift の復元力

- Amazon Redshift のインフラストラクチャセキュリティ
- Amazon Redshift での設定と脆弱性の分析

Amazon Redshift におけるデータ保護

AWS 責任共有モデル は Amazon Redshift のデータ保護に適用されます。このモデルで説明されているように、「AWS」は、「AWS クラウド」のすべてを実行するグローバルインフラストラクチャを保護する責任があります。ユーザーは、このインフラストラクチャでホストされるコンテンツに対する管理を維持する責任があります。また、使用する「AWS のサービス」のセキュリティ設定と管理タスクもユーザーの責任となります。データプライバシーに関する詳細については、 Data Privacy FAQ をご覧ください。欧州におけるデータ保護に関する情報については、General Data Protection Regulation (GDPR) Center を参照してください。

データを保護するため、「AWS アカウント」認証情報を保護し、AWS IAM アイデンティティセンター または AWS Identity and Access Management (IAM) を使用して個々のユーザーをセットアップすることをお勧めします。この方法により、それぞれのジョブを遂行するために必要な権限のみが各ユーザーに付与されます。また、次の方法でデータを保護することもお勧めします:

- 各アカウントで多要素認証 (MFA) を使用します。
- SSL/TLS を使用して AWS リソースと通信します。TLS 1.2 は必須ですが、TLS 1.3 を推奨します。
- AWS CloudTrail で API とユーザーアクティビティロギングを設定します。CloudTrail 証跡を使用して AWS アクティビティをキャプチャする方法については、「AWS CloudTrail ユーザーガイド」の「CloudTrail 証跡の使用」を参照してください。
- AWS のサービス内のすべてのデフォルトセキュリティコントロールに加え、AWS 暗号化ソリューションを使用します。
- Amazon Macie などの高度な管理されたセキュリティサービスを使用します。これらは、Amazon S3 に保存されている機密データの検出と保護を支援します。
- コマンドラインインターフェイスまたは API を使用して AWS にアクセスする際に FIPS 140-3 検証済みの暗号化モジュールが必要な場合は、FIPS エンドポイントを使用します。利用可能な FIPS エンドポイントの詳細については、「連邦情報処理規格 (FIPS) 140-3」を参照してください。

お客様の E メールアドレスなどの極秘または機密情報を、タグ、または [名前] フィールドなどの自由形式のテキストフィールドに含めないことを強くお勧めします。これは、コンソール、API、AWS CLI、または AWS SDK で Amazon Redshift またはその他の AWS のサービス を使用する場合も同様

です。タグ、または名前に使用される自由記述のテキストフィールドに入力したデータは、請求または診断ログに使用される場合があります。外部サーバーに URL を提供する場合、そのサーバーへのリクエストを検証できるように、認証情報を URL に含めないことを強くお勧めします。

データ暗号化

データ保護には、転送時 (Amazon Redshift とのデータの送受信) と、保管時 (Amazon Redshift データセンター内のディスクへの保存) のデータ保護があります。SSL を使用するか、クライアント側の暗号化を使用することによって、転送中のデータを保護できます。Amazon Redshift で保管中のデータを保護するには、次のようなオプションがあります。

- サーバー側の暗号化を使用する – データをデータセンター内のディスクに保存する前に暗号化し、オブジェクトをダウンロードするときに復号するように Amazon Redshift に要求します。
- クライアント側の暗号化を使用する — クライアント側でデータを暗号化し、暗号化したデータを Amazon Redshift にアップロードできます。この場合、暗号化プロセス、暗号化キー、関連ツールはお客様が管理してください。

保管中の暗号化

サーバー側の暗号化は、保管中のデータ暗号化に関するものです。つまり、Amazon Redshift は、データセンターの書き込み時にデータを暗号化し、お客様がデータにアクセスするときに復号します。リクエストが認証され、お客様がアクセス許可を持っている限りは、オブジェクトが暗号化されているかどうかに関係なく同じ方法でアクセスできます。

Amazon Redshift では暗号化を使用して、保管中のデータを保護します。必要に応じて、Advanced Encryption Standard AES-256 を使用して、クラスター内のディスクに保存されているデータと Amazon S3 のバックアップをすべて保護することができます。

Amazon Redshift リソースの暗号化と復号化のために使用するキーを管理するには、AWS Key Management Service (AWS KMS) を使用します。 AWS KMS では、安全で可用性の高いハードウェアとソフトウェアの組み合わせにより、クラウド向けに拡張されたキー管理システムが提供されます。AWS KMS を使用すると、キーの暗号化を作成し、このキーの使用方法を制御するポリシーを定義できます。 AWS KMS は AWS CloudTrail をサポートするため、キーの使用を監査して、キーが適切に使用されていることを確認できます。AWS KMS キーは、Amazon Redshift やサポートされている AWS サービスと組み合わせて使用できます。AWS KMSがサポートされているサービスの一覧については、 AWS Key Management Service デベロッパーガイドの How AWS Services Use AWS KMS を参照してください。

AWS Secrets Manager を使用して、プロビジョン済みのクラスターまたはサーバーレス名前空間の管理者パスワードで管理する場合、Amazon Redshift は AWS Secrets Manager が認証情報の暗号化に使用する追加の AWS KMS キーも受け入れます。この追加のキーは、AWS Secrets Manager から自動的に生成されたキーでも、ユーザーが提供するカスタムキーでも構いません。

Amazon Redshift クエリエディタ v2 は、次のようにクエリエディタに入力された情報を安全に保存します。

- クエリエディタ v2 のデータの暗号化に使用する KMS キーの Amazon リソースネーム (ARN)。
- データベース接続の情報
- ファイルとフォルダの名前と内容。

Amazon Redshift クエリエディタ v2 は、KMS キーまたはサービスアカウント KMS キーを使用して、ブロックレベルの暗号化を使用して情報を暗号化します。Amazon Redshift データの暗号化は、Amazon Redshift クラスタープロパティによって制御されます。

トピック

- Amazon Redshift データベース暗号化

Amazon Redshift データベース暗号化

Amazon Redshift では、保管中のデータを保護するために、データベースはデフォルトで暗号化されます。データベース暗号化は、クラスターとクラスターのスナップショットに適用されます。

暗号化されていないクラスターを AWS Key Management Service (AWS KMS) 暗号化を使用するように変更できます。これを実行するには、AWS が所有するキーか、カスタマーマネージドキーのいずれかを使用できます。クラスターを変更して AWS KMS 暗号化を有効にすると、Amazon Redshift は自動的にデータを新たな暗号化されたクラスターに移行します。暗号化されたクラスターから作成されたスナップショットも暗号化されます。また、クラスターを変更して データベースの暗号化 オプションを変更することで、暗号化されたクラスターを暗号化されていないクラスターに移行することもできます。詳細については、「クラスターの暗号化の変更」を参照してください。

クラスターの作成後も、デフォルトの暗号化されたクラスターを暗号化されていないクラスターに変換することはできるとはいえ、機密データを含むクラスターについては暗号化されたままにしておくことをお勧めします。また、データに適用されるガイドラインやルールに応じて暗号化の使用が必要になる場合があります。たとえば、Payment Card Industry Data Security Standard (PCI DSS)、Sarbanes-Oxley Act (SOX)、Health Insurance Portability and Accountability Act (HIPAA) といった規制では、特定の種類のデータの取り扱いに関するガイドラインが規定されています。

Amazon Redshift では、暗号化キーの階層を使用してデータベースを暗号化します。AWS Key Management Service (AWS KMS) またはハードウェアセキュリティモジュール (HSM) のいずれかを使用して、この階層の最上位の暗号化キーを管理できます。Amazon Redshift が暗号化に使用するプロセスは、キーの管理方法によって異なります。Amazon Redshift は AWS KMS と自動的に統合されますが、HSM とは統合されません。HSM を使用するときは、クライアントとサーバーの証明書を使用して、Amazon Redshift と HSM との間で信頼された接続を設定する必要があります。

Important

カスタマーマネージド KMS キーを無効にすると、Amazon Redshift はプロビジョニングされたクラスターまたはサーバーレス名前空間の KMS キーにアクセスできなくなる可能性があります。このような場合、Amazon Redshift は Amazon Redshift データウェアハウスのバックアップを取得し、14 日間 `inaccessible-kms-key` 状態になります。その期間内に KMS キーを復元すると、Amazon Redshift はアクセスを復元し、ウェアハウスは正常に機能します。KMS キーを復元せずに 14 日間が終了すると、Amazon Redshift はデータウェアハウスを削除します。ウェアハウスが `inaccessible-kms-key` 状態の間は、次の特性があります。

- データウェアハウスでクエリを実行することはできません。
- データウェアハウスがデータ共有のプロデューサーウェアハウスである場合、コンシューマーウェアハウスからデータ共有クエリを実行することはできません。
- クロスリージョンスナップショットコピーを作成することはできません。

無効な KMS キーの復元については、「AWS Key Management Service デベロッパーガイド」の「Enable and disable keys」を参照してください。ウェアハウスの KMS キーが削除された場合は、バックアップを使用して、`inaccessible-kms-key` 状態のウェアハウスが削除される前に新しいデータウェアハウスを作成できます。

パフォーマンスと可用性の向上のための暗号化プロセスの改善

RG および RA3 ノードによる暗号化

RG および RA3 ノードの暗号化プロセスの更新により、エクスペリエンスが大幅に向上しました。プロセス中に読み取りクエリと書き込みクエリの両方を実行でき、暗号化によるパフォーマンスへの影響が少なくなります。また、暗号化がはるかに迅速に終了します。更新されたプロセスステップには、復元オペレーションとクラスターメタデータのターゲットクラスターへの移行が含まれます。エ

クスペリエンスの向上は、例えば AWS KMS のような暗号化タイプでも実現します。ペタバイトスケールのデータ量であれば、運用にかかる時間が数週間から数日に短縮されます。

クラスターを暗号化する前に、データベースワークロードを引き続き実行する予定がある場合は、伸縮自在なサイズ変更が可能なノードを追加することで、パフォーマンスを向上させ、プロセスをスピードアップできます。暗号化中は伸縮自在なサイズ変更を使用できないため、暗号化する前に使用してください。通常は、ノードを追加するとコストが高くなることに注意してください。

他のノードタイプによる暗号化

DC2 ノードを使用してクラスターを暗号化する場合、RG または RA3 ノードを使用する場合のように、書き込みクエリを実行することはできません。読み取りクエリのみ実行できます。

RG および RA3 ノードでの暗号化の使用上の注意

以下のインサイトとリソースは、暗号化の準備とプロセスの監視に役立ちます。

- 暗号化開始後のクエリの実行 — 暗号化の開始後、約 15 分以内に読み取りと書き込みが可能になります。完全な暗号化プロセスが完了するまでにかかる時間は、クラスター上のデータ量とワークロードレベルによって異なります。
- 暗号化にはどれくらいの時間がかかりますか? — データを暗号化する時間は、実行中のワークロードの数、使用されているコンピューティングリソース、ノードの数、ノードのタイプなど、複数の要因によって異なります。最初は、テスト環境で暗号化を実行することをお勧めします。経験則として、ペタバイト単位のデータ量を扱う場合、暗号化が完了するまでに 1~3 日かかる可能性があります。
- 暗号化が完了したことを確認するにはどうすればよいですか? – 暗号化を有効にすると、最初のスナップショットの完了により、暗号化が完了したことが確認されます。
- 暗号化のロールバック — 暗号化操作をロールバックする必要がある場合は、暗号化が開始される前に作成された最新のバックアップから復元するのが最善の方法です。最後のバックアップの後で、新しい更新 (更新/削除/挿入) を再適用する必要があります。
- テーブルの復元の実行 — 暗号化されていないクラスターから暗号化されたクラスターにテーブルを復元することはできないことに注意してください。
- 単一ノードクラスターの暗号化 — 単一ノードクラスターの暗号化にはパフォーマンス上の制限があります。マルチノードクラスターでは、暗号化よりも時間がかかります。
- 暗号化後のバックアップの作成 — クラスター内のデータを暗号化する場合、クラスターが完全に暗号化されるまでバックアップは作成されません。この処理にかかる時間はさまざまです。バックアップにかかる時間は、クラスターのサイズによって、数時間から数日かかる場合があります。暗号化の完了後、バックアップを作成できるようになるまでに時間がかかることがあります。

バックアップおよび復元オペレーションは暗号化プロセス中に発生するため、BACKUP NO で作成したテーブルやマテリアライズドビューは保持されません。詳細については、「CREATE TABLE」または「CREATE MATERIALIZED VIEW」を参照してください。

トピック

- AWS KMS を使用した暗号化
- ハードウェアセキュリティモジュールを使用した暗号化
- 暗号化キーのローテーション
- クラスターの暗号化の変更
- HSM 暗号化されたクラスターへの移行
- 暗号化キーのローテーション

AWS KMS を使用した暗号化

Amazon Redshift によるキー管理用に AWS KMS を選択した場合、4 階層の暗号化キーがあります。これらのキーは、階層の最上位から順にルートキー、クラスターの暗号化キー (CEK)、データベースの暗号キー (DEK)、データ暗号化キーとなっています。

クラスターを起動すると、Amazon Redshift は、Amazon Redshift または AWS アカウントが作成したか、AWS KMS で使用するアクセス許可が付与されている AWS KMS keys キーのリストを返します。暗号化キーの階層でルートキーとして使用する KMS キーを選択します。

Amazon Redshift はデフォルトでは、Amazon Redshift で使用する AWS アカウントのルートキーとして自動的に生成された、AWS が所有するキーを選択します。

デフォルトのキーを使用しない場合は、Amazon Redshift でクラスターを起動する前に、AWS KMS でカスタマー管理 KMS キーを別途用意 (または作成) する必要があります。カスタマー管理キーを作成すると、データの保護に使用する暗号化キーを作成、ローテーション、無効化、アクセス制御定義、監査できるなど、より柔軟に取り扱うことができます。KMS キーの作成の詳細については、AWS Key Management Service デベロッパーガイドの キーの作成 を参照してください。

別の AWS アカウントの AWS KMS キーを使用する場合は、そのキーを使用するアクセス許可が必要であり、Amazon Redshift でその Amazon リソースネーム (ARN) を指定する必要があります。AWS KMS でのキーアクセスの詳細については、AWS Key Management Service デベロッパーガイドの キーのアクセスの制御 を参照してください。

ルートキーを選択すると、Amazon Redshift では、 AWS KMS によるデータキーの生成と、選択されたルートキーによる暗号化が要求されます。このデータキーは Amazon Redshift で CEK として使用されます。 AWS KMS では、暗号化された CEK が Amazon Redshift にエクスポートされます。CEK は、クラスターから別のネットワークのディスクの内部で、KMS キーと CEK の暗号化コンテキストに対する権限とともに保存されます。暗号化された CEK だけが Amazon Redshift にエクスポートされ、KMS キーは AWS KMS に残ります。また、Amazon Redshift は暗号化された CEK を安全なチャンネル経由でクラスターに渡し、メモリにロードします。その後、Amazon Redshift は、 AWS KMS を呼び出して CEK を復号化し、復号化した CEK をメモリにロードします。権限、暗号化コンテキスト、およびその他の AWS KMS関連概念の詳細については、 AWS Key Management Service デベロッパーガイドの 概念 を参照してください。

次に、Amazon Redshift は DEK として使用するキーを生成し、クラスターのメモリにランダムにロードします。復号化された CEK は DEK の暗号化に使用されます。その後、暗号化された DEK はクラスターから安全なチャンネル経由で Amazon Redshift に渡され、クラスターとは別のネットワーク内のディスクに内部で保存されます。CEK と同様に、DEK の暗号化と復号の両方のバージョンはクラスターのメモリにロードされます。その後、DEK の復号バージョンは、データベース内の各データブロックのランダムに生成された暗号化キーを暗号化するために使用されます。

クラスターを再起動すると、Amazon Redshift は、内部に保存した CEK と DEK の暗号化バージョンをまずメモリに再ロードしてから、AWS KMS を呼び出して KMS キーで CEK を再び復号化します。これで、CEK がメモリにロードできるようになります。復号された CEK で DEK は再び復号され、復号された DEK はメモリにロードされて、必要に応じてデータブロックキーの暗号化および復号に使用されます。

AWS KMS キーで暗号化された Amazon Redshift クラスターの作成の詳細については、「クラスターの作成」を参照してください。

別の AWS リージョンに AWS KMS 暗号化スナップショットをコピーする

AWS KMS キーは、 AWS リージョンに固有です。暗号化されたソースクラスターから別の AWS リージョンへの Amazon Redshift スナップショットのコピーを有効にして、コピー先のスナップショットに独自の AWS KMS キーを使用する場合は、コピー先の AWS リージョンのアカウントでルートキーを使用するように Amazon Redshift の権限を設定する必要があります。この権限により、Amazon Redshift はコピー先の AWS リージョンでスナップショットを暗号化できるようになります。コピー先のスナップショットを AWS リージョンが所有するキーで暗号化する場合は、コピー先の AWS リージョンで権限を設定する必要はありません。リージョン間スナップショットコピーに関する詳細については、 スナップショットを別の AWS リージョンにコピーする を参照してください。

> **Note**
>
> 暗号化されたクラスターからのスナップショットのコピーを有効にし、ルートキーとして AWS KMS を使用する場合は、クラスター名が暗号化コンテキストの一部になるため、そのクラスター名を変更することはできません。クラスター名を変更する必要がある場合、ソースの AWS リージョンでスナップショットのコピーを無効にし、クラスター名を変更してから、再びスナップショットのコピーを設定して有効にします。

スナップショットのコピー権限を設定するプロセスは、以下のとおりです。

1. ターゲットの AWS リージョンで、次の手順を実行してスナップショットのコピー権限を作成します。
   - 使用する AWS KMS キーがまだなければ作成します。AWS KMS キーの作成の詳細については、 AWS Key Management Service デベロッパーガイドの キーの作成 を参照してください。
   - スナップショットのコピー権限の名前を指定します。この名前は、 AWS アカウントの AWS リージョンで一意である必要があります。
   - 作成する権限が適用される AWS KMS キー ID を指定します。キー ID を指定しない場合は、この権限はデフォルトのキーに適用されます。
2. ソースの AWS リージョンでスナップショットのコピーを有効にし、ターゲットの AWS リージョンで作成したスナップショットのコピー権限の名前を指定します。

この事前のプロセスは、 AWS CLI、Amazon Redshift API、または SDK を使用してスナップショットのコピーを有効にする場合にのみ必要です。コンソールを使用する場合、Amazon Redshift では、リージョン間のスナップショットのコピーを有効にするときに、権限の設定ワークフローがあります。コンソールを使用して、AWS KMS暗号化されたクラスター用にリージョン間のスナップショットのコピーを設定する方法の詳細については、 AWS KMS 暗号化クラスターのクロスリージョンスナップショットのコピーを設定する を参照してください。

スナップショットがターゲットの AWS リージョンにコピーされる前に、Amazon Redshift はソースの AWS リージョンでルートキーを使用してスナップショットを復号化し、ランダムに生成された (内部管理の) RSA キーを使用して一時的な再暗号化を行います。その後、Amazon Redshift は安全なチャンネル経由でターゲットの AWS リージョンにスナップショットをコピーし、それを (内部管理の) RSA キーを使用して復号化した上で、ターゲットの AWS リージョンのルートキーを使用して再暗号化します。

ハードウェアセキュリティモジュールを使用した暗号化

キー管理に AWS KMS を使用しない場合、Amazon Redshift ではキー管理にハードウェアセキュリティモジュール (HSM) を使用できます。

> **Important**
>
> HSM 暗号化は DC2、RG、および RA3 ノードタイプではサポートされません。

HSM は、キー生成と管理を直接に制御するデバイスです。キーの管理をアプリケーションとデータベースのレイヤーから切り離すことで、より優れたセキュリティを実現します。Amazon Redshift では、キー管理のために AWS CloudHSM Classic がサポートされています。AWS KMS の代わりに HSM を使用して暗号化キーを管理するときは、暗号化プロセスが異なります。

> **Important**
>
> Amazon Redshift では、 AWS CloudHSM Classic のみがサポートされています。新しい AWS CloudHSM サービスはサポートされていません。
> AWS CloudHSM新規のお客様は、 クラシックをご利用いただけません。詳細については、CloudHSM Classic の料金を参照してください。 AWS CloudHSMClassic はすべての AWS リージョンで使用できるわけではありません。使用できる AWS リージョンについては、AWS リージョン表を参照してください。

HSM を使用するようにクラスターを設定すると、Amazon Redshift は HSM に、CEK として使用されるキーの生成と保存を要求します。ただし、 AWS KMS とは異なり、HSM では Amazon Redshift に CEK をエクスポートすることができません。代わりに、Amazon Redshift はクラスターで DEK をランダムに生成し、HSM に DEK を渡して、DEK は CEK で暗号化されます。HSM は暗号化された DEK を Amazon Redshift に返します。その DEK は、内部で保存されているランダムに生成されたルートキーでさらに暗号化され、クラスターとは別のネットワークにあるディスク内に保存されます。また、Amazon Redshift は、DEK の復号バージョンをクラスターのメモリにロードします。それにより、DEK を使用して個々のデータブロックキーを暗号化および復号できるようになります。

クラスターを再起動した場合、Amazon Redshift は、内部に保存したルートキーを使用して、内部で保存され二重に暗号化された DEK を復号化することで、その DEK を CEK で暗号化した状態に戻します。その後、CEK で暗号化された DEK は HSM に渡され、復号化されて、Amazon Redshift に戻され、再びメモリにロードされて、個々のデータブロックキーに使用できるようになります。

Amazon Redshift と HSM との間の信頼された接続の設定

クラスターキーの管理に HSM を使用するように選択した場合、Amazon Redshift と HSM の間に信頼されたネットワークリンクを構成する必要があります。これを行うには、クライアント証明書とサーバー証明書の設定が必要です。信頼された接続は、暗号化と復号化のオペレーション中に、HSM と Amazon Redshift との間で暗号化キーを渡すために使用されます。

Amazon Redshift は、ランダムに生成されたプライベートキーと公開キーの組み合わせから、公開クライアント証明書を作成します。これらは内部的に暗号化されて保存されます。ユーザーは、公開クライアント証明書を HSM にダウンロードして登録し、該当する HSM パーティションに割り当てます。

ユーザーは、HSM の IP アドレス、HSM パーティション名、HSM パーティションのパスワード、および内部ルートキーで暗号化された (パブリックな) HSM サーバー証明書を Amazon Redshift に対し指定します。Amazon Redshift は設定プロセスを完了し、HSM に接続できることを確認します。接続できない場合、クラスターは INCOMPATIBLE_HSM 状態に設定され、クラスターは作成されません。この場合、不完全なクラスターを削除してから、もう一度やり直す必要があります。

> **Important**
>
> 別の HSM パーティションを使用するクラスターを変更する場合、Amazon Redshift は新しいパーティションに接続できるか確認しますが、有効な暗号化キーがあるかは確認しません。新しいパーティションを使用する前に、新しいパーティションのキーをレプリケーションする必要があります。クラスターを再起動し、Amazon Redshift が有効なキーを見つけることができない場合、再起動は失敗します。詳細については、 HSM 間のキーのレプリケーションを参照してください。

初期設定の後で Amazon Redshift が HSM に接続できない場合、イベントが記録されます。これらのイベントの詳細については、 Amazon Redshift イベント通知を参照してください。

暗号化キーのローテーション

Amazon Redshift では、暗号化したクラスターの暗号化キーをローテーションすることができます。キーのローテーションプロセスを開始すると、Amazon Redshift は指定されたクラスターとそのクラスターの自動または手動スナップショット用の CEK をローテーションします。また、Amazon Redshift は指定されたクラスターの DEK をローテーションしますが、スナップショットの DEK をローテーションすることはできません。スナップショットの DEK は Amazon Simple Storage Service (Amazon S3) に内部で保存されている間、既存の DEK で暗号化されているためです。

ローテーションの進行中、クラスターはその完了まで ROTATING_KEYS 状態に設定され、完了すると AVAILABLE 状態に戻ります。 は、キーのローテーションプロセスの間に、復号と再暗号化の処理を行います。Amazon Redshift は、キーローテーションプロセス中に復号と再暗号化を処理します。

> **Note**
>
> ソースクラスターなしでスナップショットのキーをローテーションすることはできません。クラスターを削除する際には、スナップショットのキーローテーションが必要になるかどうかを検討してください。

キーローテーションプロセス中はクラスターが一時的に使用不能になるので、キーローテーションは、データのニーズによって必要となるか、キーの漏洩の疑いがある場合のみ行ってください。ベストプラクティスとして、保存するデータの種類を考慮してデータの暗号化キーのローテーション頻度を計画してください。キーローテーションの頻度は、データセキュリティに対する企業のポリシーと、機密データと規制コンプライアンスに関する業界標準によって異なります。セキュリティのニーズとクラスターの可用性のバランスを考慮して計画を立ててください。

キーローテーションの詳細については、「暗号化キーのローテーション」を参照してください。

クラスターの暗号化の変更

暗号化されていないクラスターは、AWS が所有するキーまたはカスタマーマネージドキーで AWS Key Management Service (AWS KMS) 暗号化を使用するように変更できます。クラスターを変更して AWS KMS 暗号化を有効にすると、Amazon Redshift は自動的にデータを新たな暗号化されたクラスターに移行します。また、AWS CLI を使用してクラスターを変更することで、暗号化されたクラスターを暗号化されていないクラスターに移行することもできますが、AWS マネジメントコンソール ではできません。

移行オペレーション中、クラスターは読み取り専用モードで閲覧可能になり、クラスターのステータスは サイズ変更と表示されます。

クラスターがAWS リージョン間のスナップショットコピーを有効にするように設定されている場合は、暗号化を変更する前に無効にする必要があります。詳細については、 スナップショットを別の AWS リージョンにコピーする と AWS KMS 暗号化クラスターのクロスリージョンスナップショットのコピーを設定するを参照してください。クラスターを変更してハードウェアセキュリティモジュール (HSM) 暗号化を有効にすることはできません。代わりに、HSM で暗号化された新しいク

ラスターを作成し、データを新しいクラスターに移行します。詳細については、「HSM 暗号化されたクラスターへの移行」を参照してください。

Amazon Redshift console

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [Clusters] (クラスター) を選択し、暗号化を変更するクラスターを選択します。
3. プロパティを選択します。
4. データベース設定 セクションで、 編集、 暗号化の編集の順に選択します。
5. いずれかの暗号化オプションを選択し、 変更の保存を選択します。

AWS CLI

暗号化を解除したクラスターを変更して AWS KMSを使用するには、次に示すように `modify-cluster` CLI コマンドを実行して `--encrypted`を指定します。デフォルトでは、デフォルトの KMS キーが使用されています。カスタマー管理キーを指定するには、 `--kms-key-id` オプションを含めます。

```
aws redshift modify-cluster --cluster-identifier <value> --encrypted --kms-key-id
 <value>
```

クラスターから暗号化を削除するには、次の CLI コマンドを実行します。

```
aws redshift modify-cluster --cluster-identifier <value> --no-encrypted
```

HSM 暗号化されたクラスターへの移行

ハードウェアセキュリティモジュール (HSM) を使用して、暗号化されていないクラスターを暗号化されたクラスターに移行するには、新しい暗号化クラスターを作成し、データを新しいクラスターに移動します。クラスターを変更して HSM 暗号化クラスターに移行することはできません。

暗号化されていないクラスターから HSM 暗号化されたクラスターに移行するには、まず既存のソースクラスターからデータをアンロードします。次に、選択した暗号化設定を使用して、新しいターゲットクラスター内のデータを再ロードします。暗号化されたクラスターを起動する方法の詳細については、 Amazon Redshift データベース暗号化を参照してください。

移行プロセスの間、ソースクラスターは最後の手順まで読み取り専用クエリで使用できます。最後のステップは、エンドポイントを切り替えるターゲットクラスターとソースクラスターの名前を変更して、すべてのトラフィックが新しいターゲットクラスターにルーティングされるようにすることです。名前を変更して再起動するまで、ターゲットクラスターは使用できません。データの転送中に、ソースクラスター上のすべてのデータロードおよびその他の書き込み操作を中断します。

移行の準備をするには

1. ビジネスインテリジェンス (BI) ツールや抽出、変換、ロード (ETL) システムなど、Amazon Redshift と対話するすべての依存システムを特定します。
2. 検証クエリを特定して移行をテストします。

   たとえば、次のクエリを使用してユーザー定義テーブルの数を検索できます。

   ```
   select count(*)
   from pg_table_def
   where schemaname != 'pg_catalog';
   ```

   次のクエリは、すべてのユーザー定義テーブルの一覧と各テーブルの行数を返します。

   ```
   select "table", tbl_rows
   from svv_table_info;
   ```
3. 移行に適した時間を選択します。クラスター使用率が最も低い時間を見つけるには、CPU 使用率やデータベース接続数などのクラスターメトリクスをモニタリングします。詳細については、クラスターのパフォーマンスデータを表示するを参照してください。
4. 未使用のテーブルを削除します。

   テーブルのリストを作成し、各テーブルがクエリされた回数を知るには、次のクエリを実行します。

   ```
   select database,
   schema,
   table_id,
   "table",
   round(size::float/(1024*1024)::float,2) as size,
   sortkey1,
   nvl(s.num_qs,0) num_qs
   from svv_table_info t
   left join (select tbl,
   ```

```
perm_table_name,
count(distinct query) num_qs
from stl_scan s
where s.userid > 1
and   s.perm_table_name not in ('Internal worktable','S3')
group by tbl,
perm_table_name) s on s.tbl = t.table_id
where t."schema" not in ('pg_internal');
```

5. 暗号化された新しいクラスターを起動します。

   ソースクラスターと同じポート番号をターゲットクラスターに使用します。暗号化されたクラスターを起動する方法の詳細については、 Amazon Redshift データベース暗号化を参照してください。

6. アンロードとロードのプロセスを設定します。

   Amazon Redshift アンロード/コピーユーティリティ を使用すると、クラスター間でデータを移行するのに役立ちます。このユーティリティは、ソースクラスターから Amazon S3 上の場所にデータをエクスポートします。データは AWS KMSで暗号化されます。ユーティリティは、データをターゲットに自動的にインポートします。必要に応じて、移行が完了した後でこのユーティリティを使用して Amazon S3 をクリーンアップすることができます。

7. テストを実行してプロセスを検証し、書き込みオペレーションを中断する必要のある期間を見積もります。

   アンロードとロードオペレーションに、データのロードとその他の書き込みオペレーションを中断して、データの整合性を維持します。最も大きなテーブルの 1 つを使用して、アンロードとロードのプロセスを実行すると、タイミングを推定するのに役立ちます。

8. スキーマ、ビュー、テーブルなどのデータベースオブジェクトを作成します。必要なデータ定義言語 (DDL) ステートメントを生成するには、 AWS GitHub リポジトリの AdminViews にあるスクリプトを使用できます。

クラスターを移行するには

1. ソースクラスターですべての ETL プロセスを停止します。

   処理中に書き込みオペレーションがないことを確認するには、Amazon Redshift マネジメントコンソールを使用して書き込み IOPS をモニタリングします。詳細については、 クラスターのパフォーマンスデータを表示するを参照してください。

2. 以前に特定した検証クエリを実行して、移行前に暗号化されていないソースクラスターに関する情報を収集します。
3. （任意）1 つのワークロード管理 (WLM) キューを作成して、ソースクラスターとターゲットクラスターの両方で使用可能な最大限のリソースを使用します。たとえば、 `data_migrate` という名前のキューを作成し、メモリーを 95 パーセント、同時実行レベル 4 でキューを構成します。詳細については、 Amazon Redshift データベースデベロッパーガイドから ユーザーグループとクエリグループに基づいてクエリをキューにルーティング を参照してください。
4. `data_migrate` キューを使用して UnloadCopyUtility を実行します。

   Amazon Redshift コンソールを使用して UNLOAD と COPY プロセスをモニタリングします。
5. 検証クエリを再度実行し、結果がソースクラスターの結果と一致することを確認します。
6. ソースクラスターとターゲットクラスターの名前を変更して、エンドポイントをスワップします。混乱を避けるために、このオペレーションは営業時間外に実行してください。
7. ETL やレポートツールなどのすべての SQL クライアントを使用して、ターゲットクラスターに接続できることを確認します。
8. 暗号化されたソースクラスターをシャットダウンします。

暗号化キーのローテーション

Amazon Redshift コンソールを使用して暗号化キーをローテーションするには、次の手順を使用します。

クラスターの暗号化キーをローテーションするには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションペインで、[Clusters] (クラスター) を選択して、暗号化キーを更新するクラスターを選択します。
3. アクション に、暗号化キーのローテーション を選択して 暗号化キーのローテーション ページを表示します。
4. 暗号化キーのローテーション ページで、 暗号化キーのローテーションを選択します。

送信中の暗号化

送信中の機密データと整合性データを保護するように環境を設定できます。

以下の詳細は、Amazon Redshift クラスターと SQL クライアントの間で JDBC/ODBC 経由で送信されるデータの暗号化に適用されます。

- Java Database Connectivity (JDBC) と Open Database Connectivity (ODBC) 接続を介して SQL クライアントツールから Amazon Redshift クラスターに接続できます。
- Amazon Redshift では Secure Sockets Layer (SSL) 接続がサポートされているため、データとサーバー証明書を暗号化でき、クライアントが接続するサーバー証明書を検証できます。このクライアントは、Amazon Redshift クラスターのリーダーノードに接続されます。詳細については、接続のセキュリティオプションを設定するを参照してください。
- SSL 接続をサポートするため、Amazon Redshift では AWS Certificate Manager (ACM) によって発行された証明書が作成され、各クラスターにインストールされます。詳細については、 SSL 接続用 ACM 証明書への移行を参照してください。
- AWS クラウド内の送信中のデータを保護するため、Amazon Redshift では COPY、UNLOAD、バックアップ、および復元オペレーションを実行する際、ハードウェアによる SSL を使用して Amazon S3 または Amazon DynamoDB と通信が行われます。

以下の詳細は、Amazon Redshift クラスターと Amazon S3 または DynamoDB の間の送信中のデータの暗号化に適用されます。

- Amazon Redshift では、COPY、UNLOAD、バックアップ、および復元オペレーションを実行する際、ハードウェアによる SSL を使用して、Amazon S3 または DynamoDB と通信します。
- Redshift Spectrum では、 AWS Key Management Service (KMS) によって管理されるアカウントのデフォルトのキーを使用した Amazon S3 サーバー側の暗号化 (SSE) がサポートされています。
- Amazon S3 と AWS KMS を使用して、Amazon Redshift のロードを暗号化できます。詳細については、「Encrypt Your Amazon Redshift Loads with Amazon S3 and AWS KMS」を参照してください。

以下の詳細は、AWS CLI、SDK、または API クライアントと Amazon Redshift エンドポイントの間で送信中のデータの暗号化と署名に適用されます。

- Amazon Redshift には、送信中のデータを暗号化するための HTTP エンドポイントが用意されています。
- Amazon Redshift への API 要求の整合性を保護するには、API 呼び出しは呼び出し元によって署名されている必要があります。呼び出しは、署名バージョン 4 の署名プロセス (Sigv4) に従って、X.509 証明書またはお客様の AWS シークレットアクセスキーを使用して署名される必要があります。詳細については、https://docs.aws.amazon.com/general/latest/gr/signature-

version-4.htmlの「AWS 全般のリファレンス署名バージョン 4 の署名プロセス」を参照してください。

- AWS CLI または AWS SDK の 1 つを使用して、 AWSに要求を送信します。これらのツールで、設定時に指定されたアクセスキーを使用すると、自動的に要求に署名されます。

以下の詳細は、Amazon Redshift クラスターと Amazon Redshift クエリエディタ v2 の間の送信時のデータの暗号化に適用されます。

- データは、TLS で暗号化されたチャンネルを介してクエリエディタ v2 と Amazon Redshift クラスターとの間で送信されます。

Amazon Redshift による VPC 暗号化コントロール

Amazon Redshift は、VPC 暗号化コントロールをサポートしています。VPC 暗号化コントロールは、リージョン内の VPC 内および VPC 間で転送中のすべてのトラフィックに対して暗号化を適用するのに役立つセキュリティ機能です。このドキュメントでは、Amazon Redshift クラスターとサーバーレスワークグループで VPC 暗号化コントロールを使用する方法について説明します。

VPC 暗号化コントロールは、VPC 内の転送中の暗号化を監視および適用するための一元的なコントロールを提供します。強制モードで有効にすると、すべてのネットワークトラフィックがハードウェアレイヤー (AWS Nitro System を使用) またはアプリケーションレイヤー (TLS/SSL を使用) で暗号化されます。

Amazon Redshift は VPC 暗号化コントロールと統合され、医療 (HIPAA)、政府 (FedRAMP)、財務 (PCI DSS) などの業界のコンプライアンス要件を満たすのに役立ちます。

Amazon Redshift での VPC 暗号化コントロールの仕組み

VPC 暗号化コントロールは 2 つのモードで動作します。

- モニタリングモード: トラフィックフローの暗号化ステータスを可視化し、暗号化されていないトラフィックを許可するリソースを特定するのに役立ちます。
- 強制モード: VPC 内の暗号化されていないトラフィックを許可するリソースの作成または使用を防止します。すべてのトラフィックは、ハードウェアレイヤー (Nitro ベースのインスタンス) またはアプリケーションレイヤー (TLS/SSL) で暗号化する必要があります。

VPC 暗号化コントロールを使用するための要件

インスタンスタイプの要件

Amazon Redshift では、VPC 暗号化コントロールをサポートするために Nitro ベースのインスタンスが必要です。最新の Redshift インスタンスタイプはすべて、必要な暗号化機能をサポートしています。

SSL/TLS の要件

VPC 暗号化コントロールが強制モードで有効になっている場合、require_ssl パラメータを true に設定する必要があり、無効にすることはできません。これにより、すべてのクライアント接続で、暗号化された TLS 接続が使用されます。

VPC 暗号化コントロールへの移行

既存のクラスターとワークグループの場合

既存の Redshift クラスターまたはサーバーレスワークグループを含む VPC では、強制モードで VPC 暗号化コントロールを有効にすることはできません。既存のクラスターまたはワークグループがある場合は、次の手順を参照して暗号化コントロールを使用します。

1. 既存のクラスターまたは名前空間のスナップショットを作成します。
2. 強制モードで VPC 暗号化コントロールを有効にして新しい VPC を作成する
3. 次のいずれかの操作を使用して、スナップショットから新しい VPC に復元します。

    - プロビジョニングされたクラスターの場合: `restore-from-cluster-snapshot` オペレーションを使用します。
    - サーバーレスの場合: ワークグループで `restore-from-snapshot` オペレーションを使用します。

暗号化コントロールを有効にして VPC に新しいクラスターまたはワークグループを作成する場合、require_ssl パラメータを true に設定する必要があります。

Amazon Redshift では、VPC 暗号化コントロールをサポートするために Nitro ベースのインスタンスが必要です。最新の Redshift インスタンスタイプはすべて、必要な暗号化機能をサポートしています。

SSL/TLS の要件

VPC 暗号化コントロールが強制モードで有効になっている場合、require_ssl パラメータを true に設定する必要があり、無効にすることはできません。これにより、すべてのクライアント接続で、暗号化された TLS 接続が使用されます。

考慮事項と制限事項

Amazon Redshift で VPC 暗号化コントロールを使用する場合は、次の点を考慮してください。

VPC の状態の制限

- VPC 暗号化コントロールが `enforce-in-progress` 状態になると、クラスターとワークグループの作成がブロックされます
- 新しいリソースを作成する前に、VPC が `enforce` モードになるまで待つ必要があります

SSL 設定

- require_ssl パラメータ: 暗号化が強制された VPC で作成されたクラスターとワークグループの場合は常に `true` である必要があります
- クラスターまたはワークグループが暗号化が強制された VPC に作成されると、`require_ssl` はその有効期間にわたって無効にすることはできません。

リージョンの可用性

この機能は、次のリージョンの Amazon Redshift Serverless では、強制モードで使用できません。

- 南米 (サンパウロ)
- 欧州 (チューリッヒ)

キーの管理

キーでデータを保護するように環境を設定することができます。

- Amazon Redshift はキー管理のために AWS Key Management Service (AWS KMS) と自動的に統合されます。 AWS KMS では、エンベロープ暗号化が使用されます。詳細については、 エンベロープ暗号化を参照してください。
- 暗号化キーが AWS KMSで管理されている場合、Amazon Redshift では 4 層のキーベースの暗号化アーキテクチャが使用されます。このアーキテクチャは、ランダムに生成された AES-256 データ暗号化キー、データベースキー、クラスターキー、およびルートキーにより構成されます。詳細については、 Amazon Redshift の使用方法 AWS KMSを参照してください。

- 独自のカスタマーマスターキー (CMK) を AWS KMSで作成できます。詳細については、「KMS キーの作成」および「Amazon Redshift (プロビジョニング済み) および Amazon Redshift Serverless での AWS KMS カスタマーマネージドキーの使用の承認」を参照してください。
- また、新しい AWS KMS keys の独自のキーマテリアルをインポートすることもできます。詳細については、 AWS Key Management Service (AWS KMS) のキーマテリアルのインポートを参照してください。
- Amazon Redshift では、外部のハードウェアセキュリティモジュール (HSM) を使用した暗号化キーの管理がサポートされています。HSM は、オンプレミスまたは AWS CloudHSMです。HSM を使用するときは、クライアントとサーバーの証明書を使用して、Amazon Redshift と HSM との間で信頼された接続を設定する必要があります。Amazon Redshift では、キー管理のために AWS CloudHSM Classic のみがサポートされています。詳細については、「ハードウェアセキュリティモジュールを使用した暗号化」を参照してください。AWS CloudHSMの詳細については、 AWS CloudHSMとは？を参照
- 暗号化したクラスターの暗号化キーをローテーションすることができます。詳細については、「暗号化キーのローテーション」を参照してください。

Amazon Redshift (プロビジョニング済み) および Amazon Redshift Serverless での AWS KMS カスタマーマネージドキーの使用の承認

Amazon Redshift は AWS Key Management Service (AWS KMS) と統合して、保管中のデータを暗号化します。カスタマーマネージドキーを使用すると、キーポリシー、キーローテーション、キーの有効化または無効化を完全に制御できます。カスタマーマネージドキーの詳細については、AWS Key Management Service デベロッパーガイドの「カスタマーマネージドキー」を参照してください。

Amazon Redshift が暗号化オペレーションでカスタマーマネージドキーを使用する場合、キーは Amazon Redshift でプロビジョニングされたクラスターまたは Amazon Redshift Serverless リソースを作成または変更するユーザーに代わって動作します。

必要なアクセス許可

カスタマーマネージドキーで暗号化された Amazon Redshift でプロビジョニングされたクラスターまたは Amazon Redshift Serverless リソースを作成して使用するには、ユーザーはカスタマーマネージドキーで次のオペレーションを呼び出すアクセス許可を持っている必要があります。

- `kms:CreateGrant`
- `kms:GenerateDataKey`

- kms:Encrypt
- kms:Decrypt
- kms:DescribeKey

カスタマーマネージドキーの所有者または管理者は、カスタマーマネージドキーのキーポリシー、またはキーポリシーで許可されている場合は IAM ポリシーで、特定のユーザーに対してこれらのアクセス許可を追加できます。

Amazon Redshift は、AWS KMS の許可を通じて、カスタマーマネージドキーにアクセスします。キーポリシーによって Amazon Redshift によるキーの許可の作成または使用が妨げられることがないようにしてください。例えば、明示的な Deny ステートメントや、Allow ステートメントの制限条件は、許可の作成をブロックし、オペレーションが失敗する可能性があります。

セキュリティのベストプラクティスとして、ユースケースに必要なアクセス許可のみを指定することをお勧めします。例えば、kms:GrantIsForAWSResource などの条件キーを使用して、kms:CreateGrant アクセス許可の範囲を絞り込むことができます。

AWS KMS キーへのアクセスの詳細については、「AWS Key Management Service デベロッパーガイド」の「AWS KMS のキーポリシー」および「他のアカウントのユーザーに KMS キーの使用を許可する」を参照してください。

データのトークナイゼーション

トークナイゼーション は、データセキュリティの目的で実際の値を不透明な値に置き換えるプロセスです。セキュリティに敏感なアプリケーションは、トークナイゼーションを使用して、個人を特定できる情報 (PII) や保護すべき医療情報 (PHI) などの機密データをトークンに置き換え、セキュリティリスクを軽減します。デトークナイゼーション は、適切なセキュリティポリシーを持つ許可されたユーザーの実際の値でトークンをリバースすることです。

サードパーティーのトークナイゼーションサービスと統合するために、 AWS Lambdaを使用して作成した Amazon Redshift ユーザー定義関数 (UDF) を使用できます。詳細については、Amazon Redshift データベースデベロッパーガイドから Lambda ユーザー定義関数を参照してください。たとえば、 Protegrityを参照してください。

Amazon Redshift は、REST API または定義済みのエンドポイントを介してアクセスされるトークナイゼーションサーバーにトークナイゼーション要求を送信します。2 つ以上の余の Lambda 関数が、トークナイゼーションとデトークナイゼーション要求を処理します。この処理では、サードパー

ティのトークナイゼーションプロバイダーが提供する Lambda 関数を使用できます。また、Amazon Redshift で Lambda UDF として登録する Lambda 関数を使用することもできます。

たとえば、列に対してトークナイゼーションまたはデトークナイゼーション UDF を呼び出すクエリが送信されたとします。Amazon Redshift クラスターは、該当する引数の行をスプールし、それらの行をバッチで Lambda 関数に並行して送信します。Amazon Redshift コンピューティングノードと Lambda 間のデータ転送は、クライアントからアクセスできない独立したネットワーク接続で行われます。Lambda 関数は、データをトークナイゼーションサーバーのエンドポイントに渡します。トークナイゼーションサーバーは、必要に応じてデータをトークナイゼーションまたはデトークナイゼーションし、それを返します。次に、Lambda 関数は、必要に応じて、結果を Amazon Redshift クラスターに送信してさらに処理し、クエリ結果を返します。

Amazon Redshift でのインターネットワークトラフィックのルーティング

Amazon Redshift で既知のネットワークルートまたはプライベートネットワークルートを介してトラフィックをルーティングできます。このページでは、企業ネットワーク上および同じ AWS リージョン内のリソース間でトラフィックをルーティングする方法について説明します。

Amazon Redshift と、企業ネットワークのクライアントおよびアプリケーションの間でトラフィックをルーティングするには:

- Virtual Private Cloud (VPC) と企業ネットワークの間にプライベート接続を設定します。Direct Connect 接続を使用して、インターネット経由またはプライベートの物理接続で IPsec VPN 接続を設定します。Direct Connect を使用すると、オンプレミスネットワークから Amazon VPC に直接接続するためのプライベート仮想インターフェイスを確立できます。これにより、お客様のネットワークと VPC をプライベートの高帯域幅ネットワークで接続することが可能になります。複数の仮想インターフェイスを使用するため、ネットワーク分離が維持しながら、複数の VPC へのプライベート接続も確立できます。詳細については、「AWS Site-to-Site VPN とは」および「What is Direct Connect?」を参照してください。

VPC の Amazon Redshift クラスターと、同じ AWS リージョン内の Amazon S3 バケットの間でトラフィックをルーティングするには、次の手順を実行します。

- ETL ロードまたはアンロードから Amazon S3 データにプライベートにアクセスするように Amazon S3 プライベート VPC エンドポイントをセットアップします。詳細については、 Amazon S3 のエンドポイントを参照してください。
- ターゲットの Amazon S3 VPC エンドポイントを指定して、Amazon Redshift クラスターの [Enhanced VPC routing (拡張された VPC のルーティング)] を有効にします。Amazon Redshift

COPY、UNLOAD、または CREATE LIBRARY コマンドによって生成されたトラフィックは、プライベートエンドポイントを介してルーティングされます。詳細については、拡張 VPC ルーティングを有効にする を参照してください。

Amazon Redshift での Identity and Access Management

Amazon Redshift へのアクセスには、AWS が要求の認証に使用する認証情報が必要です。これらの認証情報には、Amazon Redshift クラスターなどの AWS リソースへのアクセス許可が必要です。次のセクションでは、AWS Identity and Access Management(IAM) と Amazon Redshift を使用してリソースにアクセスできるユーザーを制御することで、リソースを保護する方法について詳しく説明します。

- アイデンティティを使用した認証
- アクセスコントロール

> **Important**
>
> このトピックには、アクセス許可、ID、および安全なアクセスを管理するためのベストプラクティス集が用意されています。Amazon Redshift で IAM を使用するためのベストプラクティスに精通することをお勧めします。これには、アクセス許可を適用するために IAM ロールを使用することが含まれています。これらのセクションをよく理解すると、Amazon Redshift データウェアハウスの安全性を向上させるために役立ちます。

アイデンティティを使用した認証

認証とは、アイデンティティ認証情報を使用して AWS にサインインする方法です。ユーザーは、IAM ユーザー の AWS アカウントのルートユーザー として、または IAM ロールを引き受けることによって、認証される 必要があります。

AWS IAM アイデンティティセンター (IAM アイデンティティセンター)、シングルサインオン認証、Google/Facebook 認証情報などの ID ソースからの認証情報を使用して、フェデレーテッドアイデンティティとしてサインインできます。サインインの詳細については、「AWS サインイン ユーザーガイド」の「AWS アカウントにサインインする方法」を参照してください。

プログラムによるアクセスの場合、AWS はリクエストに暗号で署名するための SDK と CLI を提供します。詳細については、「IAM ユーザーガイド」の「API リクエストに対する AWS 署名バージョン 4」を参照してください。

AWS アカウント のルートユーザー

AWS アカウントを作成すると、すべての AWS のサービスとリソースに対する完全なアクセス権を持つ AWS アカウント ルートユーザーと呼ばれる 1 つのサインイン ID を使用して開始します。日常的なタスクには、ルートユーザーを使用しないことを強くお勧めします。ルートユーザー認証情報を必要とするタスクについては、「IAM ユーザーガイド」の「ルートユーザー認証情報が必要なタスク」を参照してください。

IAM ユーザーとグループ

IAM ユーザーは、特定の個人やアプリケーションに対する特定のアクセス許可を持つアイデンティティです。長期認証情報を持つ IAM ユーザーの代わりに一時的な認証情報を使用することをお勧めします。詳細は「IAM ユーザーガイド」の「人間のユーザーが一時的な認証情報を使用して AWS にアクセスするには ID プロバイダーとのフェデレーションの使用が必要です」を参照してください。

IAM グループは、IAM ユーザーの集合を指定し、大量のユーザーに対するアクセス許可の管理を容易にします。詳細については、「IAM ユーザーガイド」の「IAM ユーザーに関するユースケース」を参照してください。

IAM ロール

IAM ロールは、特定のアクセス許可を持つアイデンティであり、一時的な認証情報を提供します。ユーザーから IAM ロール (コンソール) に切り替える、または AWS CLI や AWS API オペレーションを呼び出すことで、ロールを引き受けることができます。詳細については、「IAM ユーザーガイド」の「ロールを引き受けるための各種方法」を参照してください。

IAM ロールは、フェデレーションユーザーアクセス、一時的な IAM ユーザーのアクセス許可、クロスアカウントアクセス、クロスサービスアクセス、および Amazon EC2 で実行するアプリケーションに役立ちます。詳細については、IAM ユーザーガイド の IAM でのクロスアカウントリソースアクセス を参照してください。

> **Note**
>
> Redshift の転送アクセスセッション (FAS) は 12 時間のみ有効です。この期間を過ぎると、FAS を使用して他の サービスと統合する接続セッションを再確立する必要があります。

サービス間の混乱した代理の防止

混乱した代理問題とは、アクションを実行する許可を持たないエンティティが、より高い特権を持つエンティティにそのアクションの実行を強制できるというセキュリティ問題です。AWS では、サービス間でのなりすましによって、混乱した代理問題が発生する場合があります。サービス間でのなりすましは、1 つのサービス (呼び出し元サービス) が、別のサービス (呼び出し対象サービス) を呼び出すときに発生する可能性があります。呼び出し元サービスは、本来ならアクセスすることが許可されるべきではない方法でその許可を使用して、別のお客様のリソースに対する処理を実行するように操作される場合があります。これを防ぐため、AWS では、アカウントのリソースへのアクセス権が付与されたサービスプリンシパルで、すべてのサービスのデータを保護するために役立つツールを提供しています。詳細については、「IAM ユーザーガイド」の「サービス間の混乱した代理の防止」を参照してください。

リソースベースのポリシー内では `aws:SourceArn` および `aws:SourceAccount` のグローバル条件コンテキストキーを使用して、Amazon Redshift が別のサービスに付与する、リソースへのアクセス許可を制限することをお勧めします。Amazon S3 バケットの ARN などのアカウント ID が、`aws:SourceArn` 値に含まれていない場合、アクセス許可を制限するためには、これら両方のグローバル条件コンテキストキーを使用する必要があります。グローバル条件コンテキストキーの両方を使用し、`aws:SourceArn` の値にアカウント ID が含まれている場合、同じポリシーステートメントで使用する場合は、`aws:SourceArn` の値と `aws:SourceAccount` の値のアカウントでは同じアカウント ID を使用する必要があります。

次の例は、Amazon Redshift の混乱した代理問題を制限するために適用できるポリシーを示しています。

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "ConfusedDeputyPreventionForRedshift",
    "Effect": "Allow",
    "Principal": {
      "Service": "redshift.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:redshift:us-east-1:123456789012:cluster:my-cluster"
      },
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
```

```
      }
    }
  }
}
```

次の例は、Amazon Redshift Serverless のポリシーを示しています。

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "ConfusedDeputyPreventionForRedshiftServerless",
    "Effect": "Allow",
    "Principal": {
      "Service": "redshift-serverless.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:redshift-serverless:us-east-1:123456789012:workgroup/
my-workgroup"
      },
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      }
    }
  }
}
```

アクセスコントロール

リクエストを認証するために有効な認証情報を持つことができますが、アクセス許可を持っていなければ Amazon Redshift リソースを作成またはアクセスすることはできません。たとえば、Amazon Redshift クラスターの作成、スナップショットの作成、イベントサブスクリプションの追加などにはアクセス権限が必要です。

以下のセクションでは、Amazon Redshift のアクセス許可を管理する方法について説明します。最初に概要のセクションを読むことをお勧めします。

- Amazon Redshift リソースに対するアクセス許可の管理の概要
- Amazon Redshift でのアイデンティティベースのポリシー (IAM ポリシー) の使用

Amazon Redshift リソースに対するアクセス許可の管理の概要

すべての AWS リソースは AWS アカウントによって所有され、リソースの作成またはアクセスは、アクセス許可のポリシーによって管理されます。アカウント管理者は、アクセス許可ポリシーを IAM ID (ユーザー、グループ、ロール) にアタッチできます。一部のサービス (AWS Lambda など) では、アクセス許可ポリシーをリソースにアタッチすることもできます。

> **Note**
>
> アカウント管理者 (または管理者ユーザー) は、管理者権限を持つユーザーです。詳細については、「IAM ユーザーガイド」の「IAM ベストプラクティス」を参照してください。

アクセス権限を付与する場合、アクセス権限を取得するユーザー、取得するアクセス権限の対象となるリソース、およびそれらのリソースに対して許可される特定のアクションを決定します。

Amazon Redshift のリソースとオペレーション

Amazon Redshift には、IAM アクセス許可ポリシーで使用するサービス固有のリソース、アクション、条件コンテキストキーが用意されています。

Amazon Redshift、Amazon Redshift Serverless、Amazon Redshift Data API、Amazon Redshift クエリエディタ v2 のアクセス許可

アクセスコントロール をセットアップするときに、IAM ID にアタッチできるアクセス権限ポリシー (ID ベースのポリシー) を記述します。詳細なリファレンスについては「サービス認可リファレンスガイド」で以下のトピックを参照してください。

- Amazon Redshift の場合、`redshift:` プレフィックスを使用する「Amazon Redshift のアクション、リソース、および条件キー」を参照してください。
- Amazon Redshift Serverless の場合、`redshift-serverless:` プレフィックスを使用する「Amazon Redshift Serverless のアクション、リソース、および条件キー」を参照してください。
- Amazon Redshift Data API の場合、`redshift-data:` プレフィックスを使用する「Amazon Redshift Data API のアクション、リソース、および条件キー」を参照してください。
- Amazon Redshift クエリエディタ v2 の場合、`sqlworkbench:` プレフィックスを使用する「AWS SQL Workbench (Amazon Redshift クエリエディタ v2) のアクション、リソース、および条件キー」を参照してください。

クエリエディタ v2 には、API オペレーションに直接対応しない権限のみのアクションが含まれています。こうしたアクションには、「サービス認可リファレンス」で [permission only] と表示されています。

この「サービス認可リファレンス」には、IAM ポリシーで使用できる API オペレーションに関する情報が記載されています。また、アクセス許可を付与できる AWS リソースや、きめ細かなアクセスコントロールに含めることができる条件キーも含まれています。条件の詳細については、「詳細に設定されたアクセスコントロールのための IAM ポリシー条件の使用」を参照してください。

ポリシーの Action フィールドにアクションを、ポリシーの Resource フィールドにリソース値を、ポリシーの Condition フィールドに条件を指定します。Amazon Redshift のアクションを指定するには、API オペレーション名 (例: redshift:CreateCluster など) の前に redshift: プレフィックスを使用します。

リソース所有権についての理解

リソース所有者 は、リソースを作成した AWS アカウントです。つまり、リソース所有者とは、リソースの作成リクエストを認証する プリンシパルエンティティ (ルートアカウント、IAM ユーザー、または IAM ロール) の AWS アカウントです。以下の例は、このしくみを示しています。

- AWS アカウントのルートアカウント認証情報を使用して DB クラスターを作成する場合、AWSアカウントは Amazon Redshift リソースの所有者です。
- Amazon Redshift リソースを作成するためのアクセス許可を持つ AWS アカウントで IAM ロールを作成すると、ロールを引き受けることのできるユーザーなら誰でも Amazon Redshift リソースを作成できます。Amazon Redshift リソースは、ロールが属するお客様の AWS アカウントで所有されます。
- AWS アカウントに IAM ユーザーを作成し、Amazon Redshift リソースを作成するアクセス許可をそのユーザーに付与すると、そのユーザーは Amazon Redshift リソースを作成できます。ただし Amazon Redshift リソースは、ユーザーが属するお客様の AWS アカウントで所有されます。ほとんどの場合、この方法はお勧めしません。代わりに IAM ロールを作成し、このロールにアクセス許可をアタッチしてユーザーに割り当てることをお勧めします。

リソースへのアクセスの管理

アクセス権限ポリシー では、誰が何にアクセスできるかを記述します。以下のセクションで、アクセス許可ポリシーを作成するために使用可能なオプションについて説明します。

> **Note**
>
> このセクションでは、Amazon Redshift のコンテキストでの IAM の使用について説明します。これは、IAM サービスに関する詳細情報を取得できません。IAM に関する詳細なドキュメントについては、「IAM ユーザーガイド」の「What is IAM?」(IAM とは？) を参照してください。IAM ポリシー構文の詳細と説明については、「IAM ユーザーガイド」の「AWS IAM ポリシーリファレンス」を参照してください。

IAM アイデンティティにアタッチされたポリシーはアイデンティティベースのポリシー (IAM ポリシー) と呼ばれ、リソースにアタッチされたポリシーはリソースベースのポリシーと呼ばれます。Amazon Redshift では、アイデンティティベースのポリシー (IAM ポリシー) のみをサポートしています。

アイデンティティベースのポリシー (IAM ポリシー)

アクセス許可を割り当てるには、ポリシーを IAM ロールにアタッチし、このロールをユーザーまたはグループに割り当てます。次に示す例は、AWS アカウントで Amazon Redshift クラスターを作成、削除、変更、再起動するためのアクセス許可を含むポリシーを示しています。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid":"AllowManageClusters",
      "Effect":"Allow",
      "Action": [
        "redshift:CreateCluster",
        "redshift:DeleteCluster",
        "redshift:ModifyCluster",
        "redshift:RebootCluster"
      ],
      "Resource":"*"
    }
  ]
}
```

Amazon Redshift で ID ベースのポリシーを使用する場合の詳細については、「Amazon Redshift でのアイデンティティベースのポリシー (IAM ポリシー) の使用」を参照してください。ユーザー、グループ、ロール、アクセス許可の詳細については、IAM ユーザーガイドの「アイデンティティ (ユーザー、グループ、ロール)」を参照してください。

リソースベースのポリシー

Simple Storage Service (Amazon S3) などの他のサービスでは、リソースベースの許可ポリシーもサポートされています。たとえば、ポリシーを S3 バケットにアタッチして、そのバケットに対するアクセス許可を管理できます。Amazon Redshift では、リソースベースのポリシーはサポートされていません。

ポリシー要素の指定 : アクション、効果、リソース、プリンシパル

サービスは、Amazon Redshift リソースごとに (「Amazon Redshift のリソースとオペレーション」を参照)、一連の API オペレーションを定義します ([アクション]を参照) 。これらの API オペレーションを実行するためのアクセス許可を付与するために、Amazon Redshift ではポリシーに一連のアクションを定義できます。1 つの API オペレーションの実行で、複数のアクションのアクセス権限が必要になる場合があります。

以下は、基本的なポリシーの要素です。

- リソース – ポリシーで Amazon リソースネーム (ARN) を使用して、ポリシーを適用するリソースを識別します。詳細については、「Amazon Redshift のリソースとオペレーション」を参照してください。
- アクション - アクションのキーワードを使用して、許可または拒否するリソースオペレーションを識別します。たとえば、`redshift:DescribeClusters` 許可は、Amazon Redshift `DescribeClusters` オペレーションの実行をユーザーに許可します。
- 効果 – ユーザーが特定のアクションを要求する際の効果を指定します。許可または拒否のいずれかになります。リソースへのアクセスを明示的に付与 (許可) していない場合、アクセスは暗黙的に拒否されます。また、明示的にリソースへのアクセスを拒否すると、別のポリシーによってアクセスが許可されている場合でも、ユーザーはそのリソースにアクセスできなくなります。
- プリンシパル - ID ベースのポリシー (IAM ポリシー) で、ポリシーがアタッチされているユーザーが黙示的なプリンシパルとなります。リソースベースのポリシーでは、権限 (リソースベースのポリシーにのみ適用) を受け取りたいユーザー、アカウント、サービス、またはその他のエンティティを指定します。Amazon Redshift では、リソースベースのポリシーはサポートされていません。

IAM ポリシーの構文と説明の詳細はこちら IAM ユーザーガイドの AWS IAM ポリシーリファレンス。

すべての Amazon Redshift API アクションとそれらが適用されるリソースの表については、「Amazon Redshift、Amazon Redshift Serverless、Amazon Redshift Data API、Amazon Redshift クエリエディタ v2 のアクセス許可」を参照してください。

ポリシーでの条件を指定する

アクセス許可を付与するとき、アクセスポリシー言語を使用して、ポリシーが有効になる条件を指定できます。たとえば、特定の日付の後にのみ適用されるポリシーが必要になる場合があります。ポリシー言語での条件の指定の詳細については、「IAM ユーザーガイド」の「IAM JSON ポリシー要素: 条件」を参照してください。

アクセス権限ポリシーが適用される条件を指定するには、IAM アクセス権限ポリシーに Condition 要素を含めます。たとえば、redshift:CreateCluster アクションを使用してクラスターを作成することをユーザーに許可するポリシーを作成します。その後、特定のリージョンでのクラスターの作成のみにそのアクションを制限する Condition 要素を追加できます。詳細については、「詳細に設定されたアクセスコントロールのための IAM ポリシー条件の使用」を参照してください。すべての条件キー値とそれらが適用される Amazon Redshift アクションとリソースを示すリストについては、「Amazon Redshift、Amazon Redshift Serverless、Amazon Redshift Data API、Amazon Redshift クエリエディタ v2 のアクセス許可」を参照してください。

詳細に設定されたアクセスコントロールのための IAM ポリシー条件の使用

Amazon Redshift では、条件キーを使用し、リソースのタグに基づいてリソースへのアクセスを制限できます。Amazon Redshift の一般的な条件キーは以下のとおりです。

条件キー	説明
aws:RequestTag	リソースの作成時にはタグのキー (名前) と値を含める必要があります。詳細については、「IAM ユーザーガイド」の「aws:RequestTag」を参照してください。
aws:ResourceTag	特定のタグのキーと値に基づいてリソースへのユーザーアクセスを制限します。詳細については、「IAM ユーザーガイド」の「aws:ResourceTag」を参照してください。

条件キー	説明
aws:TagKeys	このキーを使用して、リクエスト内のタグキーとポリシーで指定したキーを比較します。詳細については、「IAM ユーザーガイド」の「aws:TagKeys」を参照してください。

タグの詳細については、「Amazon Redshift のリソースのタグ付け」を参照してください。

redshift:RequestTag と redshift:ResourceTag 条件キーをサポートする API アクションのリストについては、「Amazon Redshift、Amazon Redshift Serverless、Amazon Redshift Data API、Amazon Redshift クエリエディタ v2 のアクセス許可」を参照してください。

次の条件キーは Amazon Redshift の GetClusterCredentials アクションで使用できます。

条件キー	説明
redshift:DurationSeconds	期間に対して指定できる秒数を制限します。
redshift:DbName	指定できるデータベース名を制限します。
redshift:DbUser	指定できるデータベースユーザー名を制限します。

例 1: aws:ResourceTag 条件キーを使用してアクセスを制限する

次の IAM ポリシーを使用すると、us-west-2 リージョンで特定の AWS アカウントの Amazon Redshift クラスターにタグ (名前が environment、値が test) が含まれている場合に限り、クラスターを変更することをユーザーに許可できます。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Sid":"AllowModifyTestCluster",
        "Effect": "Allow",
        "Action": "redshift:ModifyCluster",
```

```
        "Resource": "arn:aws:redshift:us-west-2:123456789012:cluster:*",
        "Condition": {
            "StringEquals": {
                "aws:ResourceTag/environment": "test"
            }
        }
    }
}
```

例 2: aws:RequestTag 条件キーを使用してアクセスを制限する

次の IAM ポリシーを使用すると、クラスターを作成するコマンドにタグ (名前が usage 、値が production) が含まれている場合に限り、Amazon Redshift クラスターを作成することが許可されます。aws:TagKeys と ForAllValues 修飾子との条件は、costcenter キーと usage キーのみを要求で指定できることを指定します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Sid":"AllowCreateProductionCluster",
        "Effect": "Allow",
        "Action": [
            "redshift:CreateCluster",
            "redshift:CreateTags"
        ],
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:RequestTag/usage": "production"
            },
            "ForAllValues:StringEquals": {
                "aws:TagKeys": [
                    "costcenter",
                    "usage"
                ]
            }
        }
    }
}
```

Amazon Redshift でのアイデンティティベースのポリシー (IAM ポリシー) の使用

このトピックでは、アカウント管理者が IAM アイデンティティ (ユーザー、グループ、ロール) へのアクセス権限ポリシーをアタッチする、アイデンティティベースのポリシーの例を示します。

> **Important**
>
> 初めに、Amazon Redshift のリソースへのアクセスを管理するための基本概念と使用できるオプションについて説明する概要トピックをお読みになることをお勧めします。詳細については、「Amazon Redshift リソースに対するアクセス許可の管理の概要」を参照してください。

以下に示しているのは、アクセス権限ポリシーの例です。このポリシーでは、ユーザーはすべてのクラスターを作成、削除、変更、再起動できます。さらに、クラスター識別子が AWS リージョン `us-west-2` および AWS アカウント `123456789012` の `production` で始まるクラスターを削除または変更するアクセス権限が拒否されます。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid":"AllowClusterManagement",
      "Action": [
        "redshift:CreateCluster",
        "redshift:DeleteCluster",
        "redshift:ModifyCluster",
        "redshift:RebootCluster"
      ],
      "Resource": [
        "*"
      ],
      "Effect": "Allow"
    },
    {
      "Sid":"DenyDeleteModifyProtected",
      "Action": [
```

```
        "redshift:DeleteCluster",
        "redshift:ModifyCluster"
      ],
      "Resource": [
        "arn:aws:redshift:us-west-2:123456789012:cluster:production*"
      ],
      "Effect": "Deny"
    }
  ]
}
```

このポリシーには以下の 2 つのステートメントがあります。

- 最初のステートメントでは、ユーザーがクラスターを作成、削除、変更、再起動するユーザーのアクセス権限を付与します。ステートメントでは、ワイルドカード文字 (*) が `Resource` 値として指定されます。そうすることで、ルート AWS アカウントが所有するすべての Amazon Redshift リソースにポリシーを適用できます。
- 2 番目のステートメントでは、クラスターの削除または変更するアクセス権限を拒否します。ステートメントではワイルドカード文字 (*) が含まれる `Resource` 値のためにクラスター Amazon リソースネーム (ARN) を指定します。その結果、クラスター識別子が `production`で始まり、ルート AWS アカウントが所有するすべての Amazon Redshift クラスターにこのステートメントが適用されます。

AWSAmazon Redshift の 管理ポリシー

AWS は、によって作成され管理されるスタンドアロンの IAM ポリシーを提供することで、多くの一般的ユースケースに対応します。AWSマネージドポリシーは、一般的ユースケースに必要な許可を付与することで、どの許可が必要なのかをユーザーが調査する必要をなくすることができます。詳細については、「IAM ユーザーガイド」の「AWS 管理ポリシー」を参照してください。

独自のカスタム IAM ポリシーを作成して、Amazon Redshift API オペレーションとリソースのためのアクセス権限を許可することもできます。これらのカスタムポリシーは、それらのアクセス許可が必要な IAM ロールまたはグループにアタッチできます。

次のセクションで、アカウント内のユーザーにアタッチでき、Amazon Redshift に固有の AWS 管理ポリシーについて説明します。

Amazon Redshift での AWS 管理ポリシーに関する更新

Amazon Redshift で管理ポリシーの変更の追跡が開始されてからの分について、AWS 管理ポリシーの更新に関する詳細を表示します。このページの変更に関する自動アラートを入手するには、Amazon Redshift ドキュメントの履歴ページから、RSS フィードをサブスクライブしてください。

変更	説明	日付
AmazonRedshiftFederatedAuthorization - 新しいポリシー	Amazon Redshift は、Amazon Redshift フェデレーティッド認可でクエリを実行するための新しく使いやすいポリシーを追加しました。	2025 年 11 月 21 日
AmazonRedshiftServiceLinkedRolePolicy – 既存ポリシーへの更新	管理ポリシーにアクション `lakeformation:GetDataAccess` に対するアクセス許可が追加されました。これを追加すると、AWS Lake Formation からカタログ情報を取得するアクセス許可が付与されます。 アクション `glue:GetCatalog` および `glue:GetCatalogs` の追加の条件が管理ポリシーに追加されました。	2025 年 3 月 13 日
AmazonRedshiftServiceLinkedRolePolicy – 既存ポリシーへの更新	アクション `glue:GetCatalog` および `glue:GetCatalogs` に対するアクセス許可が管理ポリシーに追加されました。これらを追加すると、AWS Glue からカタログ情報を取得するアクセス許可が付与されます。	2024 年 12 月 3 日

変更	説明	日付
AmazonRedshiftServiceLinkedRolePolicy – 既存ポリシーへの更新	管理ポリシーにアクション servicequotas:GetServiceQuota に対するアクセス許可が追加されました。これにより、クォータまたは制限にアクセスするためのアクセス許可が付与されます。	2024 年 3 月 8 日
AmazonRedshiftQueryEditorV2FullAccess – 既存ポリシーへの更新	アクション redshift-serverless:ListNamespaces および redshift-serverless:ListWorkgroups に対するアクセス許可が管理ポリシーに追加されました。これらの追加により、Amazon Redshift データウェアハウスのサーバーレス名前空間とサーバーレスワークグループを一覧表示する権限が付与されます。	2024 年 2 月 21 日

変更	説明	日付
AmazonRedshiftQueryEditorV2NoSharing – 既存ポリシーへの更新	アクション `redshift-serverless:ListNamespaces` および `redshift-serverless:ListWorkgroups` に対するアクセス許可が管理ポリシーに追加されました。これらの追加により、Amazon Redshift データウェアハウスのサーバーレス名前空間とサーバーレスワークグループを一覧表示する権限が付与されます。	2024 年 2 月 21 日
AmazonRedshiftQueryEditorV2ReadSharing – 既存ポリシーへの更新	アクション `redshift-serverless:ListNamespaces` および `redshift-serverless:ListWorkgroups` に対するアクセス許可が管理ポリシーに追加されました。これらの追加により、Amazon Redshift データウェアハウスのサーバーレス名前空間とサーバーレスワークグループを一覧表示する権限が付与されます。	2024 年 2 月 21 日

変更	説明	日付
AmazonRedshiftQueryEditorV2ReadWriteSharing – 既存ポリシーへの更新	アクション `redshift-serverless:ListNamespaces` および `redshift-serverless:ListWorkgroups` に対するアクセス許可が管理ポリシーに追加されました。これらの追加により、Amazon Redshift データウェアハウスのサーバーレス名前空間とサーバーレスワークグループを一覧表示する権限が付与されます。	2024 年 2 月 21 日
AmazonRedshiftReadOnlyAccess – 既存ポリシーへの更新	管理ポリシーにアクション `redshift:ListRecommendations` に対するアクセス許可が追加されました。これにより、Amazon Redshift Advisor のレコメンデーションを一覧表示するアクセス許可が付与されます。	2024 年 2 月 7 日
AmazonRedshiftServiceLinkedRolePolicy – 既存ポリシーへの更新	アクション `ec2:AssignIpv6Addresses` および `ec2:UnassignIpv6Addresses` に対するアクセス許可が管理ポリシーに追加されました。これらを追加すると、IP アドレスの割り当てと割り当て解除を行うアクセス許可が付与されます。	2023 年 10 月 31 日

変更	説明	日付
AmazonRedshiftQueryEditorV2NoSharing – 既存ポリシーへの更新	アクション sqlworkbench:GetAutocompletionMetadata および sqlworkbench:GetAutocompletionResource に対するアクセス許可が管理ポリシーに追加されました。これらを追加すると、クエリの編集中に SQL をオートコンプリートするためのデータベース情報を生成および取得する権限が付与されます。	2023 年 8 月 16 日
AmazonRedshiftQueryEditorV2ReadSharing – 既存ポリシーへの更新	アクション sqlworkbench:GetAutocompletionMetadata および sqlworkbench:GetAutocompletionResource に対するアクセス許可が管理ポリシーに追加されました。これらを追加すると、クエリの編集中に SQL をオートコンプリートするためのデータベース情報を生成および取得する権限が付与されます。	2023 年 8 月 16 日

変更	説明	日付
AmazonRedshiftQueryEditorV2ReadWriteSharing – 既存ポリシーへの更新	アクション sqlworkbench:GetAutocompletionMetadata および sqlworkbench:GetAutocompletionResource に対するアクセス許可が管理ポリシーに追加されました。これらを追加すると、クエリの編集中に SQL をオートコンプリートするためのデータベース情報を生成および取得する権限が付与されます。	2023 年 8 月 16 日

変更	説明	日付
AmazonRedshiftServiceLinkedRolePolicy – 既存ポリシーへの更新	シークレットを作成して管理するための AWS Secrets Manager のアクションのアクセス許可が管理ポリシーに追加されました。追加されたアクセス許可は、次のとおりです。 • secretsmanager:GetRandomPassword • secretsmanager:DescribeSecret • secretsmanager:PutSecretValue • secretsmanager:UpdateSecret • secretsmanager:UpdateSecretVersionStage • secretsmanager:RotateSecret • secretsmanager:DeleteSecret	2023 年 8 月 14 日

変更	説明	日付
AmazonRedshiftServiceLinkedRolePolicy – 既存ポリシーへの更新	セキュリティグループとルーティングルールの作成と管理のための Amazon EC2 でのアクションのアクセス許可がマネージドポリシーから削除されました。これらのアクセス許可は、サブネットと VPC の作成に関するものです。削除されたアクセス許可は以下の通りです。 • ec2:AuthorizeSecurityGroupEgress • ec2:AuthorizeSecurityGroupIngress • ec2:UpdateSecurityGroupRuleDescriptionsEgress • ec2:ReplaceRouteTableAssociation • ec2:CreateRouteTable • ec2:AttachInternetGateway • ec2:UpdateSecurityGroupRuleDescriptionsIngress • ec2:AssociateRouteTable • ec2:RevokeSecurityGroupIngress • ec2:CreateRoute	2023 年 5 月 8 日

変更	説明	日付
	• ec2:CreateSecurityGroup • ec2:RevokeSecurityGroupEgress • ec2:ModifyVpcAttribute • ec2:CreateSubnet • ec2:CreateInternetGateway • ec2:CreateVpc これらは、Purpose:RedshiftMigrateToVpc リソースタグと関連付けられていました。このタグは、Amazon EC2 Classic から Amazon EC2 VPC に移行するためのタスクに対するアクセス許可の範囲を制限しました。リソースタグの詳細については、「タグを使用した AWS リソースへのアクセスの制御」を参照してください。	

変更	説明	日付
AmazonRedshiftDataFullAccess – 既存ポリシーへの更新	管理ポリシーにアクション redshift:GetClusterCredentialsWithIAM に対するアクセス許可が追加されました。指定された AWS アカウント によって Amazon Redshift データベースにアクセスするための詳細な一時的認証情報を取得する許可を付与します。	2023 年 4 月 7 日
AmazonRedshiftServiceLinkedRolePolicy – 既存ポリシーへの更新	セキュリティグループルールの作成と管理のための Amazon EC2 でのアクションの許可が管理ポリシーに追加されました。これらのセキュリティグループとルールは、明確に Amazon Redshift aws:RequestTag/Redshift リソースタグに関連付けられます。これにより、アクセス許可の範囲は特定の Amazon Redshift のリソースに制限されます。	2023 年 4 月 6 日
AmazonRedshiftQueryEditorV2NoSharing – 既存ポリシーへの更新	管理ポリシーにアクション sqlworkbench:GetSchemaInference に対するアクセス許可が追加されました。ファイルから推測される列とデータ型を取得する許可を付与します。	2023 年 3 月 21 日

変更	説明	日付
AmazonRedshiftQueryEditorV2ReadSharing – 既存ポリシーへの更新	管理ポリシーにアクション sqlworkbench:GetSchemaInference に対するアクセス許可が追加されました。ファイルから推測される列とデータ型を取得する許可を付与します。	2023 年 3 月 21 日
AmazonRedshiftQueryEditorV2ReadWriteSharing – 既存ポリシーへの更新	管理ポリシーにアクション sqlworkbench:GetSchemaInference に対するアクセス許可が追加されました。ファイルから推測される列とデータ型を取得する許可を付与します。	2023 年 3 月 21 日
AmazonRedshiftQueryEditorV2NoSharing – 既存ポリシーへの更新	管理ポリシーにアクション sqlworkbench:AssociateNotebookWithTab に対するアクセス許可が追加されました。これを追加すると、ユーザー自身のノートブックにリンクされたタブを作成および更新する権限が付与されます。	2023 年 2 月 2 日

変更	説明	日付
AmazonRedshiftQueryEditorV2ReadSharing – 既存ポリシーへの更新	管理ポリシーにアクション sqlworkbench:AssociateNotebookWithTab に対するアクセス許可が追加されました。これを追加すると、ユーザー自身のノートブックやユーザーと共有されているノートブックにリンクさごれたタブの作成と更新の権限が付与されます。	2023 年 2 月 2 日
AmazonRedshiftQueryEditorV2ReadWriteSharing – 既存ポリシーへの更新	管理ポリシーにアクション sqlworkbench:AssociateNotebookWithTab に対するアクセス許可が追加されました。これを追加すると、ユーザー自身のノートブックやユーザーと共有されているノートブックにリンクさごれたタブの作成と更新の権限が付与されます。	2023 年 2 月 2 日

変更	説明	日付
AmazonRedshiftQueryEditorV2NoSharing – 既存ポリシーへの更新	ノートブックを使用するアクセス許可を付与するために、Amazon Redshift は次のアクションのアクセス許可を追加しました。 • sqlworkbench:ListNotebooks • sqlworkbench:CreateNotebook • sqlworkbench:DuplicateNotebook • sqlworkbench:CreateNotebookFromVersion • sqlworkbench:ImportNotebook • sqlworkbench:GetNotebook • sqlworkbench:UpdateNotebook • sqlworkbench:DeleteNotebook • sqlworkbench:CreateNotebookCell • sqlworkbench:DeleteNotebookCell • sqlworkbench:UpdateNotebookCellContent	2022 年 10 月 17 日

変更	説明	日付
	• sqlworkbench:UpdateNotebookCellLayout • sqlworkbench:BatchGetNotebookCell • sqlworkbench:ListNotebookVersions • sqlworkbench:CreateNotebookVersion • sqlworkbench:GetNotebookVersion • sqlworkbench:DeleteNotebookVersion • sqlworkbench:RestoreNotebookVersion • sqlworkbench:ExportNotebook	

変更	説明	日付
AmazonRedshiftQueryEditorV2ReadSharing – 既存ポリシーへの更新	ノートブックを使用するアクセス許可を付与するために、Amazon Redshift は次のアクションのアクセス許可を追加しました。 • sqlworkbench:ListNotebooks • sqlworkbench:CreateNotebook • sqlworkbench:DuplicateNotebook • sqlworkbench:CreateNotebookFromVersion • sqlworkbench:ImportNotebook • sqlworkbench:GetNotebook • sqlworkbench:UpdateNotebook • sqlworkbench:DeleteNotebook • sqlworkbench:CreateNotebookCell • sqlworkbench:DeleteNotebookCell • sqlworkbench:UpdateNotebookCellContent	2022 年 10 月 17 日

変更	説明	日付
	• sqlworkbench:UpdateNotebookCellLayout • sqlworkbench:BatchGetNotebookCell • sqlworkbench:ListNotebookVersions • sqlworkbench:CreateNotebookVersion • sqlworkbench:GetNotebookVersion • sqlworkbench:DeleteNotebookVersion • sqlworkbench:RestoreNotebookVersion • sqlworkbench:ExportNotebook	

変更	説明	日付
AmazonRedshiftQueryEditorV2ReadWriteSharing – 既存ポリシーへの更新	ノートブックを使用するアクセス許可を付与するために、Amazon Redshift は次のアクションのアクセス許可を追加しました。 • sqlworkbench:ListNotebooks • sqlworkbench:CreateNotebook • sqlworkbench:DuplicateNotebook • sqlworkbench:CreateNotebookFromVersion • sqlworkbench:ImportNotebook • sqlworkbench:GetNotebook • sqlworkbench:UpdateNotebook • sqlworkbench:DeleteNotebook • sqlworkbench:CreateNotebookCell • sqlworkbench:DeleteNotebookCell • sqlworkbench:UpdateNotebookCellContent	2022 年 10 月 17 日

変更	説明	日付
	• sqlworkbench:UpdateNotebookCellLayout • sqlworkbench:BatchGetNotebookCell • sqlworkbench:ListNotebookVersions • sqlworkbench:CreateNotebookVersion • sqlworkbench:GetNotebookVersion • sqlworkbench:DeleteNotebookVersion • sqlworkbench:RestoreNotebookVersion • sqlworkbench:ExportNotebook	
AmazonRedshiftServiceLinkedRolePolicy – 既存ポリシーへの更新	Amazon Redshift は、CloudWatch にメトリクスを公開できるように名前空間 AWS/Redshift を追加しました。	2022 年 9 月 7 日
AmazonRedshiftQueryEditorV2NoSharing – 既存ポリシーへの更新	Amazon Redshift がアクション sqlworkbench:ListQueryExecutionHistory と sqlworkbench:GetQueryExecutionHistory にアクセス許可を追加しました。これにより、クエリ履歴を表示するアクセス許可が付与されます。	2022 年 8 月 30 日

変更	説明	日付
AmazonRedshiftQueryEditorV2ReadSharing – 既存ポリシーへの更新	Amazon Redshift がアクション `sqlworkbench:ListQueryExecutionHistory` と `sqlworkbench:GetQueryExecutionHistory` にアクセス許可を追加しました。これにより、クエリ履歴を表示するアクセス許可が付与されます。	2022 年 8 月 30 日
AmazonRedshiftQueryEditorV2ReadWriteSharing – 既存ポリシーへの更新	Amazon Redshift がアクション `sqlworkbench:ListQueryExecutionHistory` と `sqlworkbench:GetQueryExecutionHistory` にアクセス許可を追加しました。これにより、クエリ履歴を表示するアクセス許可が付与されます。	2022 年 8 月 30 日
AmazonRedshiftFullAccess – 既存ポリシーへの更新	Amazon Redshift サーバーレスのアクセス権限は、既存の AmazonRedshiftFullAccess マネージドポリシーに追加されます。	2022 年 7 月 22 日

変更	説明	日付
AmazonRedshiftDataFullAccess – 既存ポリシーへの更新	Amazon Redshift は、タグキー RedshiftDataFullAccess や任意のタグ値でタグ付けされたリソースへのアクセスを許可するために、aws:ResourceTag/RedshiftDataFullAccess タグのアクセス権限で redshift-serverless:GetCredentials のデフォルトの適用条件を StringEquals から StringLike に更新しました。	2022 年 7 月 11 日
AmazonRedshiftDataFullAccess – 既存ポリシーへの更新	Amazon Redshift が新しいアクセス許可を追加し、Amazon Redshift サーバーレスへの一時的な認証情報に redshift-serverless:GetCredentials が許可されました。	2022 年 7 月 8 日
AmazonRedshiftQueryEditorV2NoSharing – 既存ポリシーへの更新	Amazon Redshift がアクション sqlworkbench:GetAccountSettings にアクセス許可を追加しました。これによりアカウント設定を取得するアクセス許可が付与されます。	2022 年 6 月 15 日

変更	説明	日付
AmazonRedshiftQueryEditorV2ReadSharing – 既存ポリシーへの更新	Amazon Redshift がアクション sqlworkbench:GetAccountSettings にアクセス許可を追加しました。これによりアカウント設定を取得するアクセス許可が付与されます。	2022 年 6 月 15 日
AmazonRedshiftQueryEditorV2ReadWriteSharing – 既存ポリシーへの更新	Amazon Redshift がアクション sqlworkbench:GetAccountSettings にアクセス許可を追加しました。これによりアカウント設定を取得するアクセス許可が付与されます。	2022 年 6 月 15 日
AmazonRedshiftServiceLinkedRolePolicy – 既存ポリシーへの更新	新しい Amazon Redshift サーバーレスエンドポイントへのパブリックアクセスを有効にするため、Amazon Redshift ではカスタマーアカウントの VPC エンドポイントの Elastic ネットワークインターフェイスに Elastic IP アドレスを割り当てて、関連付けます。これは、サービスにリンクされたロールを介して提供される権限によって行われます。このユースケースを有効にするため、Elastic IP アドレスを割り当てて解放するアクションが Amazon Redshift サーバーレスのサービスにリンクされたロールに追加されます。	2022 年 5 月 26 日

変更	説明	日付
AmazonRedshiftQueryEditorV2FullAccess – 既存ポリシーへの更新	アクション sqlworkbench:ListTaggedResources に対するアクセス許可。これは、Amazon Redshift クエリエディタ v2 のリソース専用に設定されています。このポリシーの更新により、クエリエディタ v2 からのみ tag:GetResources の呼び出しを行う権利が与えられます。	2022 年 2 月 22 日
AmazonRedshiftQueryEditorV2NoSharing – 既存ポリシーへの更新	アクション sqlworkbench:ListTaggedResources に対するアクセス許可。これは、Amazon Redshift クエリエディタ v2 のリソース専用に設定されています。このポリシーの更新により、クエリエディタ v2 からのみ tag:GetResources の呼び出しを行う権利が与えられます。	2022 年 2 月 22 日
AmazonRedshiftQueryEditorV2ReadSharing – 既存ポリシーへの更新	アクション sqlworkbench:ListTaggedResources に対するアクセス許可。これは、Amazon Redshift クエリエディタ v2 のリソース専用に設定されています。このポリシーの更新により、クエリエディタ v2 からのみ tag:GetResources の呼び出しを行う権利が与えられます。	2022 年 2 月 22 日

変更	説明	日付
AmazonRedshiftQueryEditorV2ReadWriteSharing – 既存ポリシーへの更新	アクション `sqlworkbench:ListTaggedResources` に対するアクセス許可。これは、Amazon Redshift クエリエディタ v2 のリソース専用に設定されています。このポリシーの更新により、クエリエディタ v2 からのみ `tag:GetResources` の呼び出しを行う権利が与えられます。	2022 年 2 月 22 日
AmazonRedshiftQueryEditorV2ReadSharing – 既存ポリシーへの更新	管理ポリシーにアクション `sqlworkbench:AssociateQueryWithTab` に対するアクセス許可が追加されました。これにより、お客様は共有するクエリにリンクされたエディタタブを作成できます。	2022 年 2 月 22 日
AmazonRedshiftServiceLinkedRolePolicy – 既存ポリシーへの更新	Amazon Redshift に、Amazon Redshift ネットワークおよび VPC リソースの管理を有効化するための、新しいアクションに対するアクセス許可を追加しました。	2021 年 11 月 22 日

変更	説明	日付
AmazonRedshiftAllCommandsFullAccess - 新しいポリシー	Amazon Redshift に、Amazon Redshift コンソールから作成された IAM ロールの使用を許可する新しいポリシーが追加されました。同時にこのポリシーを、Amazon S3 からの COPY、UNLOAD、CREATE EXTERNAL SCHEMA、CREATE EXTERNAL FUNCTION、CREATE MODEL、および CREATE LIBRARY の各コマンドを、クラスターが実行するためのデフォルトとして設定しました。	2021 年 11 月 18 日
AmazonRedshiftServiceLinkedRolePolicy – 既存ポリシーへの更新	Amazon Redshift に、監査ログのエクスポートを含む Amazon Redshift CloudWatch ロググループとログストリームの管理を有効化するための、新しいアクションに対するアクセス許可を追加しました。	2021 年 11 月 15 日
AmazonRedshiftFullAccess – 既存ポリシーへの更新	Amazon Redshift に、モデルの説明可能性、DynamoDB、Redshift Spectrum、および Amazon RDS フェデレーションを有効化するための、新しいアクセス許可を追加しました。	2021 年 10 月 7 日

変更	説明	日付
AmazonRedshiftQueryEditorV2FullAccess - 新しいポリシー	Amazon Redshift では、Amazon Redshift クエリエディタ v2 へのフルアクセスを許可する新しいポリシーが追加されました。	2021 年 9 月 24 日
AmazonRedshiftQueryEditorV2NoSharing - 新しいポリシー	Amazon Redshift は、リソースを共有せずに Amazon Redshift クエリエディタ v2 を使用できるようにする新しいポリシーを追加しました。	2021 年 9 月 24 日
AmazonRedshiftQueryEditorV2ReadSharing - 新しいポリシー	Amazon Redshift は、Amazon Redshift クエリエディタ v2 での読み取り共有を許可する新しいポリシーを追加しました。	2021 年 9 月 24 日
AmazonRedshiftQueryEditorV2ReadWriteSharing - 新しいポリシー	Amazon Redshift は、Amazon Redshift クエリエディタ v2 での読み取りと更新の共有を許可する新しいポリシーを追加しました。	2021 年 9 月 24 日
AmazonRedshiftFullAccess – 既存ポリシーへの更新	Amazon Redshift が新しいアクセス許可を追加し、`sagemaker:*Job*` を許可しました。	2021 年 8 月 18 日
AmazonRedshiftDataFullAccess – 既存ポリシーへの更新	Amazon Redshift が新しいアクセス許可を追加し、`AuthorizeDataShare` を許可しました。	2021 年 8 月 12 日

変更	説明	日付
AmazonRedshiftDataFullAccess – 既存ポリシーへの更新	Amazon Redshift が新しいアクセス許可を追加し、`BatchExecuteStatement` を許可しました。	2021 年 7 月 27 日
Amazon Redshift が変更の追跡を開始	Amazon Redshift が AWS 管理ポリシーの変更の追跡を開始しました。	2021 年 7 月 27 日

AmazonRedshiftReadOnlyAccess

AWS アカウントにあるすべての Amazon Redshift リソースに対する、読み取り専用のアクセス権を付与します。

AmazonRedshiftReadOnlyAccess ポリシーは IAM コンソールに、AmazonRedshiftReadOnlyAccess はAWS管理ポリシーリファレンスガイドに記載されています。

AmazonRedshiftFullAccess

AWS アカウントの内のすべての Amazon Redshift リソースに対する、完全なアクセス権を付与します。さらに、このポリシーは全ての Amazon Redshift サーバーレスリソースへの完全なアクセスを許可します。

AmazonRedshiftFullAccess ポリシーは IAM コンソールに、AmazonRedshiftFullAccess はAWS管理ポリシーリファレンスガイドに記載されています。

AmazonRedshiftQueryEditor

Amazon Redshift コンソールのクエリエディタへの、完全なアクセス権を付与します。

AmazonRedshiftQueryEditor ポリシーは IAM コンソールに、AmazonRedshiftQueryEditor はAWS管理ポリシーリファレンスガイドに記載されています。

AmazonRedshiftDataFullAccess

AWS アカウントの Amazon Redshift Data API オペレーション、およびリソースに対する完全なアクセス権を付与します。

AmazonRedshiftDataFullAccess ポリシーは IAM コンソールに、AmazonRedshiftDataFullAccess はAWS管理ポリシーリファレンスガイドに記載されています。

AmazonRedshiftQueryEditorV2FullAccess

Amazon Redshift クエリエディタ v2 のオペレーションとリソースに対する、完全なアクセス権を付与します。このポリシーは、その他の必要なサービスへのアクセス権限も付与します。

AmazonRedshiftQueryEditorV2FullAccess ポリシーは IAM コンソールにあり、AmazonRedshiftQueryEditorV2FullAccess はAWSマネージドポリシーリファレンスガイドに記載されています。

AmazonRedshiftQueryEditorV2NoSharing

リソースを共有せずに Amazon Redshift クエリエディタ v2 を操作するための権限を付与します。このポリシーは、その他の必要なサービスへのアクセス権限も付与します。このポリシーを使用しているプリンシパルは、リソース (クエリなど) にタグ付けして、同じ AWS アカウントにあるその他のプリンシパルと共有することができません。

AmazonRedshiftQueryEditorV2NoSharing 禁止ポリシーは IAM コンソールに、AmazonRedshiftQueryEditorV2NoSharing はAWSマネージドポリシーリファレンスガイドに記載されています。

AmazonRedshiftQueryEditorV2ReadSharing

限定されたリソースを共有しながら、Amazon Redshift クエリエディタ v2 を操作できる権限を付与します。このポリシーは、その他の必要なサービスへのアクセス権限も付与します。このポリシーを使用しているプリンシパルは、リソース (クエリなど) にタグ付けして、同じ AWS アカウントにあるその他のプリンシパルと共有することができます。付与されたプリンシパルは、チームと共有されているリソースの読み取りはできますが、更新はできません。

AmazonRedshiftQueryEditorV2ReadSharing ポリシーは IAM コンソールに、AmazonRedshiftQueryEditorV2ReadSharing はAWS管理ポリシーリファレンスガイドに記載されています。

AmazonRedshiftQueryEditorV2ReadWriteSharing

リソースを共有して、Amazon Redshift クエリエディタ v2 を操作できる権限を付与します。このポリシーは、その他の必要なサービスへのアクセス権限も付与します。このポリシーを使用しているプリンシパルは、リソース (クエリなど) にタグ付けして、同じ AWS アカウントにあるその他のプ

リンシパルと共有することができます。付与されたプリンシパルは、そのチームと共有されているリソースを読み取り、更新することができます。

AmazonRedshiftQueryEditorV2ReadWriteSharing ポリシーは IAM コンソールに、AmazonRedshiftQueryEditorV2ReadWriteSharing はAWS管理ポリシーリファレンスガイドに記載されています。

AmazonRedshiftServiceLinkedRolePolicy

IAM エンティティに AmazonRedshiftServiceLinkedRolePolicy をアタッチすることはできません。このポリシーは、Amazon Redshift がアカウントリソースにアクセスすることを許可する、サービスにリンクされたロールにアタッチされます。詳細については、「Amazon Redshift でのサービスにリンクされたロールの使用」を参照してください。

AmazonRedshiftServiceLinkedRolePolicy ポリシーは IAM コンソールに、AmazonRedshiftServiceLinkedRolePolicy はAWS管理ポリシーリファレンスガイドに記載されています。

AmazonRedshiftAllCommandsFullAccess

Amazon Redshift コンソールから作成された IAM ロールを使用するための機能を付与します。このロールは、Amazon S3 からの COPY に加え、UNLOAD、CREATE EXTERNAL SCHEMA、CREATE EXTERNAL FUNCTION、および CREATE MODEL の各コマンドをクラスターが実行するためのデフォルトとして設定されす。同時にこのポリシーは、Amazon S3、CloudWatch Logs、Amazon SageMaker AI、AWS Glue などの関連サービスで、SELECT ステートメントを実行することも許可します。

AmazonRedshiftAllCommandsFullAccess ポリシーは IAM コンソールに、AmazonRedshiftAllCommandsFullAccess はAWS管理ポリシーリファレンスガイドに記載されています。

AmazonRedshiftFederatedAuthorization

このポリシーは、Amazon Redshift フェデレーティッドアクセス許可を持つ Glue Data Catalog データベースに対してクエリを実行するために必要な IAM アクションを統合します。このようなクエリは AWS Glue を通過するため、オブジェクトを検出するためにカタログオブジェクトに対する Get アクションが必要であり、オブジェクトを変更するために Create、Update、Rename、Delete アクションが必要です。リソースは Amazon Redshift によって管理されるため、プリンシパルにはクエリを完了するための Redshift アクセス許可も必要です。`glue:FederateAuthorization` アクションにより AWS Glue はカタログオブジェクトの承認決定を Amazon Redshift に委任できます。

このポリシーでは、プリンシパルは Amazon Redshift フェデレーティッドアクセス許可を持つカタログに対してクエリを実行できますが、Amazon Redshift 名前空間を AWS Glue に登録および登録解除することはできません。Amazon Redshift フェデレーティッドアクセス許可設定については、IAM ポリシー要件に関するドキュメントを参照してください。

AmazonRedshiftFederatedAuthorization ポリシーは IAM コンソールに、AmazonRedshiftFederatedAuthorization は「AWS マネージドポリシーリファレンスガイド」に記載されています。

独自のカスタム IAM ポリシーを作成して、Amazon Redshift API オペレーションとリソースのためのアクセス権限を許可することもできます。これらのカスタムポリシーは、それらのアクセス許可が必要な IAM ロールまたはグループにアタッチできます。

Redshift Spectrum を使用するために必要なアクセス許可

Amazon Redshift Spectrum には、リソースにアクセスするためのその他の AWS サービスへのアクセス許可が必要です。Redshift Spectrum での IAM ポリシーのアクセス許可の詳細については、「Amazon Redshift データベースデベロッパーガイド」から「Amazon Redshift Spectrum 用 IAM ポリシー」を参照してください。

Amazon Redshift コンソールの使用に必要なアクセス許可

Amazon Redshift コンソールを使用するユーザーは、AWS アカウントの Amazon Redshift リソースを記述できる最小限のアクセス許可のセットを持つ必要があります。また、それらのアクセス許可では、Amazon EC2 セキュリティ、Amazon CloudWatch、Amazon SNS、ネットワーク情報など、その他の関連情報の記述も許可される必要があります。

これらの最小限必要なアクセス権限よりも制限された IAM ポリシーを作成している場合、その IAM ポリシーを使用するユーザーに対してコンソールは意図したとおりには機能しません。これらのユーザーが引き続き Amazon Redshift コンソールを使用できるようにするには、`AmazonRedshiftReadOnlyAccess` 管理ポリシーもユーザーにアタッチしてください これを行う方法については、AWSAmazon Redshift の 管理ポリシーに記載されています。

Amazon Redshift コンソールのクエリエディタへのアクセス権をユーザーに付与するには、「Amazon Redshift コンソールのクエリエディタを使用するために必要なアクセス許可」を参照してください。

AWS CLI または Amazon Redshift API のみを呼び出すユーザーには、最小限のコンソールのアクセス許可を付与する必要はありません。

Amazon Redshift コンソールのクエリエディタを使用するために必要なアクセス許可

ユーザーが Amazon Redshift クエリエディタを操作するには、そのユーザーが Amazon Redshift および Amazon Redshift データ API オペレーションに対する最低限のアクセス権限を持っている必要があります。秘密を使用してデータベースに接続するには、Secrets Manager の権限も必要です。

ユーザーに Amazon Redshift コンソールのクエリエディタへのアクセス許可を付与するには、AmazonRedshiftQueryEditor と AmazonRedshiftReadOnlyAccess AWS 管理ポリシーをアタッチします。AmazonRedshiftQueryEditor ポリシーでは、ユーザーが SQL ステートメントの結果のみを取得することを許可しています。つまり、AmazonRedshiftQueryEditor AWS 管理ポリシーのこのセクションに示されているのと同じ aws:userid によって送信されるステートメントです。

```
{
    "Sid":"DataAPIIAMStatementPermissionsRestriction",
    "Action": [
        "redshift-data:GetStatementResult",
        "redshift-data:CancelStatement",
        "redshift-data:DescribeStatement",
        "redshift-data:ListStatements"
    ],
    "Effect": "Allow",
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "redshift-data:statement-owner-iam-userid": "${aws:userid}"
        }
    }
}
```

ユーザーが同じ IAM ロール内の他のユーザーの SQL ステートメントの結果を取得できるようにするには、現在のユーザーへのアクセスを制限する条件を指定せずに、独自のポリシーを作成します。また、ポリシーを変更するためのアクセスを制限します。

クエリエディタv2 を使用するために必要なアクセス許可

ユーザーが Amazon Redshift クエリエディタ v2 を操作するには、そのユーザーが Amazon Redshift、クエリエディタ v2 オペレーションに対する最低限のアクセス権限を持っている必要があります。 AWS などのサービス AWS Key Management Service、AWS Secrets Manager、タグ付けサービスなどがあります。

ユーザーにクエリエディタ v2 へのフルアクセス権限を付与するには、AmazonRedshiftQueryEditorV2FullAccess AWS 管理ポリシーをアタッチします。AmazonRedshiftQueryEditorV2FullAccess ポリシーでは、クエリエディタ v2 リソース (クエリなど) を同じチーム内のその他のユーザーと共有する権限をユーザーに許可します。クエリエディタ v2 リソースへのアクセスを制御する方法の詳細については、IAM コンソールでクエリエディタ v2 の特定の管理ポリシーの定義を参照してください。

Amazon Redshift クエリエディタの v2 AWS 管理ポリシーでは、条件内で AWS タグを使用して、リソースへのアクセスをスコープします。クエリエディタ v2 の場合、クエリの共有は、プリンシパル (IAM ロール) にアタッチした IAM ポリシーのタグキーおよび値 "aws:ResourceTag/sqlworkbench-team": "${aws:PrincipalTag/sqlworkbench-team}" に基づいて行われます。クエリエディタ v2 では、同じ AWS アカウント内で同じタグ値 (accounting-team など) を持つプリンシパルは、同じチームに属しています。一度に 1 つのチームにのみ関連付けることができます。管理者権限を持つユーザーは、IAM コンソールで、すべてのチームメンバーに同じ値の sqlworkbench-team タグを付与してチームを設定することができます。sqlworkbench-team のタグ値が IAM ユーザーまたは IAM ロールに対して変更された場合、変更が共有リソースに反映されるまで遅延が発生する可能性があります。リソース (クエリなど) のタグ値が変更された場合は、変更が反映されるまで遅延が発生する可能性があります。また、チームメンバーは tag:GetResources のアクセス許可を共有します。

例 : IAM ロールの **accounting-team** タグを追加する

1. AWS マネジメントコンソール にサインインして、IAM コンソール https://console.aws.amazon.com/iam/ を開きます。
2. コンソールのナビゲーションペインで、ロール を選択し、編集するロールの名前を選択します。
3. タグ タブを選択し、タグ管理を選択します。
4. タグキー sqlworkbench-team と値 accounting-teamを追加。
5. 変更を保存を選択します。

   これで、IAM プリンシパル (この IAM ロールがアタッチされた) がチームとクエリを共有すると、その他の accounting-team タグ値は、クエリを表示することができます。

IAM ロールや IAM ユーザーなど、プリンシパルにタグをアタッチする方法については、「IAM ユーザーガイド」の「IAM リソースのタグ付け」を参照してください。

ID プロバイダ (IdP) を使用して、セッションレベルでチームを設定することもできます。これにより、同じ IAM ロールを使用する複数のユーザーが異なるチームを持つことができます。IAM ロール信頼ポリシーが `sts:TagSession` オペレーションを許可している必要があります。詳しくは、「IAM ユーザーガイド」の「セッションタグの追加に必要な権限」を参照してください。IdP によって提供される SAML アサーションにプリンシパルタグ属性を追加します。

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:sqlworkbench-
team">
    <AttributeValue>accounting-team</AttributeValue>
</Attribute>
```

ID プロバイダ (IdP) の指示に従って、ディレクトリから取得したコンテンツを SAML 属性に設定します。アイデンティティプロバイダー (IdPs) と Amazon Redshift の詳細については、 IAM ユーザーガイド の IAM 認証を使用したデータベースユーザー認証情報の生成 と アイデンティティプロバイダーとフェデレーションを参照してください。

`sqlworkbench:CreateNotebookVersion` は、ノートブックセルの現在のコンテンツを取得し、アカウントにノートブックバージョンを作成するアクセス許可を付与します。つまり、バージョン作成時には、ノートブックの現在のコンテンツはバージョンのコンテンツと同じです。その後、バージョンのセルのコンテンツは、現在のノートブックが更新されたときと同じままになります。`sqlworkbench:GetNotebookVersion` は、ノートブックのバージョンを取得するアクセス許可を付与します。`sqlworkbench:BatchGetNotebookCell` アクセス許可は持っていないものの、ノートブックに対する `sqlworkbench:CreateNotebookVersion` および `sqlworkbench:GetNotebookVersion` アクセス許可を持っているユーザーは、そのバージョンのノートブックセルにアクセスできます。`sqlworkbench:BatchGetNotebookCell` アクセス許可を持っていないこのユーザーは、最初にバージョンを作成してから取得することで、ノートブックのセルのコンテンツを取得できます。

Amazon Redshift スケジューラを使用するために必要なアクセス許可

Amazon Redshift スケジューラを使用する際、スケジューラがユーザーに代わってアクセス許可を引き受けることができるよう、Amazon Redshift スケジューラ (**`scheduler.redshift.amazonaws.com`**) に対して信頼関係を持つ IAM ロールをセットアップします。また、スケジュールする Amazon Redshift API オペレーションのロールに対してポリシー (アクセス許可) をアタッチすることもできます。

次の例は、Amazon Redshift スケジューラと Amazon Redshift との信頼関係をセットアップする JSON フォーマットでのポリシードキュメントを示しています。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "scheduler.redshift.amazonaws.com",
                    "redshift.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

信頼エンティティの詳細については、「IAM ユーザーガイド」の「AWS サービスに権限を委譲するロールの作成」を参照してください。

また、スケジュールする Amazon Redshift オペレーションに対してもアクセス許可を追加する必要があります。

スケジューラで `ResizeCluster` オペレーションを使用するには、次に類似したアクセス許可を IAM ポリシーに追加します。環境によって、ポリシーをより限定的にする必要がある場合もあります。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "redshift:ResizeCluster",
            "Resource": "*"
        }
    ]
```

```
}
```

Amazon Redshift のスケジューラ用のロールを作成する手順については、「IAM ユーザーガイド」の「AWS サービスのロールの作成 (コンソール)」を参照してください。IAM コンソールでロールを作成する際は、次の選択を行います。

- このロールを使用するサービスを選択で、Redshiftを選択します。
- ユースケースの選択で Redshift - スケジューラを選択します。
- Amazon Redshift オペレーションがスケジュールされるのを許可するロールにポリシーを作成またはアタッチします。ポリシーの作成 を選択するか、ポリシーをアタッチするロールを変更します。スケジュールするオペレーションに対して JSON ポリシーを入力します。
- ロールを作成したら、`redshift.amazonaws.com`サービスを含むIAM ロールの 信頼関係 を編集します。

作成する IAM ロールには、`scheduler.redshift.amazonaws.com`と`redshift.amazonaws.com` という信頼されているエンティティがあります。また、`"redshift:ResizeCluster"`などのサポートされた Amazon Redshift API アクションを許可するポリシーがアタッチされています。

Amazon EventBridge スケジューラを使用するために必要なアクセス許可

Amazon EventBridge スケジューラを使用する際、スケジューラがユーザーに代わってアクセス許可を引き受けることができるよう、EventBridge スケジューラ (**`events.amazonaws.com`**) に対して信頼関係を持つ IAM ロールをセットアップします。また、スケジュールする Amazon Redshift Data API オペレーションのロールと Amazon EventBridge オペレーションのポリシーに、ポリシー (アクセス許可) をアタッチすることもできます。

EventBridge スケジューラは、コンソールの Amazon Redshift クエリエディタを使用してスケジュールされたクエリを作成するときに使用します。

IAM ロールを作成して、IAM コンソールでスケジュールされたクエリを実行できます。この IAM ロールで、`AmazonEventBridgeFullAccess` と `AmazonRedshiftDataFullAccess`をアタッチします。

次の例は、EventBridge スケジューラとの信頼関係をセットアップする JSON フォーマットでのポリシードキュメントを示しています。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "events.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

信頼エンティティの詳細については、「IAM ユーザーガイド」の「AWS のサービスにアクセス許可を委任するロールの作成」を参照してください。

EventBridge スケジューラのロールを作成する手順については、「IAM ユーザーガイド」から「AWS のサービス用ロールの作成 (コンソール)」を参照してください。IAM コンソールでロールを作成する際は、次の選択を行います。

- このロールを使用するサービスを選択で、CloudWatch Eventsを選択します。
- ユースケースの選択で CloudWatch Eventsを選択します。
- 次のアクセス許可ポリシー : `AmazonEventBridgeFullAccess` と `AmazonRedshiftDataFullAccess`をアタッチします。

作成する IAM ロールには、`events.amazonaws.com`という信頼されたエンティティがあります。また、`"redshift-data:*"`などのサポートされた Amazon Redshift API アクションを許可するポリシーがアタッチされています。

Amazon Redshift 機械学習 (ML) を使用するために必要なアクセス許可

Amazon Redshift 機械学習 (ML) を使用するために必要なアクセス権限の記述を以下に示します。

Amazon SageMaker AI で Amazon Redshift ML を使用するには、デフォルトよりも制限の厳しいポリシーをアタッチした IAM ロールを作成します。以下のフォーマットを使用できます。必要に応じてこのポリシーを変更することもできます。

次のポリシーは、Amazon Redshift のモデルの説明可能性機能を使用して SageMaker AI Autopilot を実行するために必要なアクセス許可を示しています。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sagemaker:CreateTrainingJob",
                "sagemaker:CreateAutoMLJob",
                "sagemaker:CreateCompilationJob",
                "sagemaker:CreateEndpoint",
                "sagemaker:DescribeAutoMLJob",
                "sagemaker:DescribeTrainingJob",
                "sagemaker:DescribeCompilationJob",
                "sagemaker:DescribeProcessingJob",
                "sagemaker:DescribeTransformJob",
                "sagemaker:ListCandidatesForAutoMLJob",
                "sagemaker:StopAutoMLJob",
                "sagemaker:StopCompilationJob",
                "sagemaker:StopTrainingJob",
                "sagemaker:DescribeEndpoint",
                "sagemaker:InvokeEndpoint",
                "sagemaker:StopProcessingJob",
                "sagemaker:CreateModel",
                "sagemaker:CreateProcessingJob"
            ],
            "Resource": [
                "arn:aws:sagemaker:*:*:model/*redshift*",
                "arn:aws:sagemaker:*:*:training-job/*redshift*",
                "arn:aws:sagemaker:*:*:automl-job/*redshift*",
                "arn:aws:sagemaker:*:*:compilation-job/*redshift*",
                "arn:aws:sagemaker:*:*:processing-job/*redshift*",
                "arn:aws:sagemaker:*:*:transform-job/*redshift*",
                "arn:aws:sagemaker:*:*:endpoint/*redshift*"
```

```
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:DescribeLogStreams",
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:*:*:log-group:/aws/sagemaker/Endpoints/*redshift*",
                "arn:aws:logs:*:*:log-group:/aws/sagemaker/ProcessingJobs/
*redshift*",
                "arn:aws:logs:*:*:log-group:/aws/sagemaker/TrainingJobs/
*redshift*",
                "arn:aws:logs:*:*:log-group:/aws/sagemaker/TransformJobs/
*redshift*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:PutMetricData"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "cloudwatch:namespace": [
                        "SageMaker",
                        "/aws/sagemaker/Endpoints",
                        "/aws/sagemaker/ProcessingJobs",
                        "/aws/sagemaker/TrainingJobs",
                        "/aws/sagemaker/TransformJobs"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ecr:BatchCheckLayerAvailability",
                "ecr:BatchGetImage",
                "ecr:GetAuthorizationToken",
```

```
                "ecr:GetDownloadUrlForLayer"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetBucketAcl",
                "s3:GetBucketCors",
                "s3:GetEncryptionConfiguration",
                "s3:GetBucketLocation",
                "s3:ListBucket",
                "s3:ListAllMyBuckets",
                "s3:ListMultipartUploadParts",
                "s3:ListBucketMultipartUploads",
                "s3:PutObject",
                "s3:PutBucketAcl",
                "s3:PutBucketCors",
                "s3:DeleteObject",
                "s3:AbortMultipartUpload",
                "s3:CreateBucket"
            ],
            "Resource": [
                "arn:aws:s3:::redshift-downloads",
                "arn:aws:s3:::redshift-downloads/*",
                "arn:aws:s3:::*redshift*",
                "arn:aws:s3:::*redshift*/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetBucketAcl",
                "s3:GetBucketCors",
                "s3:GetEncryptionConfiguration",
                "s3:GetBucketLocation",
                "s3:ListBucket",
                "s3:ListAllMyBuckets",
                "s3:ListMultipartUploadParts",
                "s3:ListBucketMultipartUploads",
                "s3:PutObject",
                "s3:PutBucketAcl",
```

```
                "s3:PutBucketCors",
                "s3:DeleteObject",
                "s3:AbortMultipartUpload",
                "s3:CreateBucket"
            ],
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {
                    "s3:ExistingObjectTag/Redshift": "true"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::*:role/*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "redshift.amazonaws.com",
                        "sagemaker.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

次のポリシーに、Amazon DynamoDB、Redshift Spectrum、および Amazon RDS フェデレーションへのアクセスを可能にするため必要な、完全かつ最小限のアクセス許可を示します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "sagemaker:CreateTrainingJob",
                "sagemaker:CreateAutoMLJob",
                "sagemaker:CreateCompilationJob",
                "sagemaker:CreateEndpoint",
                "sagemaker:DescribeAutoMLJob",
                "sagemaker:DescribeTrainingJob",
                "sagemaker:DescribeCompilationJob",
                "sagemaker:DescribeProcessingJob",
                "sagemaker:DescribeTransformJob",
                "sagemaker:ListCandidatesForAutoMLJob",
                "sagemaker:StopAutoMLJob",
                "sagemaker:StopCompilationJob",
                "sagemaker:StopTrainingJob",
                "sagemaker:DescribeEndpoint",
                "sagemaker:InvokeEndpoint",
                "sagemaker:StopProcessingJob",
                "sagemaker:CreateModel",
                "sagemaker:CreateProcessingJob"
            ],
            "Resource": [
                "arn:aws:sagemaker:*:*:model/*redshift*",
                "arn:aws:sagemaker:*:*:training-job/*redshift*",
                "arn:aws:sagemaker:*:*:automl-job/*redshift*",
                "arn:aws:sagemaker:*:*:compilation-job/*redshift*",
                "arn:aws:sagemaker:*:*:processing-job/*redshift*",
                "arn:aws:sagemaker:*:*:transform-job/*redshift*",
                "arn:aws:sagemaker:*:*:endpoint/*redshift*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:DescribeLogStreams",
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:*:*:log-group:/aws/sagemaker/Endpoints/*redshift*",
                "arn:aws:logs:*:*:log-group:/aws/sagemaker/ProcessingJobs/
*redshift*",
                "arn:aws:logs:*:*:log-group:/aws/sagemaker/TrainingJobs/
*redshift*",
```

```
                "arn:aws:logs:*:*:log-group:/aws/sagemaker/TransformJobs/
*redshift*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:PutMetricData"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "cloudwatch:namespace": [
                        "SageMaker",
                        "/aws/sagemaker/Endpoints",
                        "/aws/sagemaker/ProcessingJobs",
                        "/aws/sagemaker/TrainingJobs",
                        "/aws/sagemaker/TransformJobs"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ecr:BatchCheckLayerAvailability",
                "ecr:BatchGetImage",
                "ecr:GetAuthorizationToken",
                "ecr:GetDownloadUrlForLayer"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetBucketAcl",
                "s3:GetBucketCors",
                "s3:GetEncryptionConfiguration",
                "s3:GetBucketLocation",
                "s3:ListBucket",
                "s3:ListAllMyBuckets",
                "s3:ListMultipartUploadParts",
                "s3:ListBucketMultipartUploads",
```

```
                "s3:PutObject",
                "s3:PutBucketAcl",
                "s3:PutBucketCors",
                "s3:DeleteObject",
                "s3:AbortMultipartUpload",
                "s3:CreateBucket"
            ],
            "Resource": [
                "arn:aws:s3:::redshift-downloads",
                "arn:aws:s3:::redshift-downloads/*",
                "arn:aws:s3:::*redshift*",
                "arn:aws:s3:::*redshift*/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetBucketAcl",
                "s3:GetBucketCors",
                "s3:GetEncryptionConfiguration",
                "s3:GetBucketLocation",
                "s3:ListBucket",
                "s3:ListAllMyBuckets",
                "s3:ListMultipartUploadParts",
                "s3:ListBucketMultipartUploads",
                "s3:PutObject",
                "s3:PutBucketAcl",
                "s3:PutBucketCors",
                "s3:DeleteObject",
                "s3:AbortMultipartUpload",
                "s3:CreateBucket"
            ],
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {
                    "s3:ExistingObjectTag/Redshift": "true"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:Scan",
```

```
                "dynamodb:DescribeTable",
                "dynamodb:Getitem"
            ],
            "Resource": [
                "arn:aws:dynamodb:*:*:table/*redshift*",
                "arn:aws:dynamodb:*:*:table/*redshift*/index/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "elasticmapreduce:ListInstances"
            ],
            "Resource": [
                "arn:aws:elasticmapreduce:*:*:cluster/*redshift*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "elasticmapreduce:ListInstances"
            ],
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {
                    "elasticmapreduce:ResourceTag/Redshift": "true"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "lambda:InvokeFunction"
            ],
            "Resource": "arn:aws:lambda:*:*:function:*redshift*"
        },
        {
            "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": [
                "arn:aws:glue:*:*:table/*redshift*/*",
                "arn:aws:glue:*:*:catalog",
                "arn:aws:glue:*:*:database/*redshift*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetResourcePolicy",
                "secretsmanager:GetSecretValue",
                "secretsmanager:DescribeSecret",
                "secretsmanager:ListSecretVersionIds"
            ],
            "Resource": [
                "arn:aws:secretsmanager:*:*:secret:*redshift*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword",
                "secretsmanager:ListSecrets"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "secretsmanager:ResourceTag/Redshift": "true"
                }
```

```
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::*:role/*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "redshift.amazonaws.com",
                        "glue.amazonaws.com",
                        "sagemaker.amazonaws.com",
                        "athena.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

オプションとして、暗号化用に AWS KMS キーを使用するために、以下のアクセス許可をポリシーに追加します。

```
{
    "Effect": "Allow",
    "Action": [
        "kms:CreateGrant",
        "kms:Decrypt",
        "kms:DescribeKey",
        "kms:Encrypt",
        "kms:GenerateDataKey*"
    ],
    "Resource": [
        "arn:aws:kms:<your-region>:<your-account-id>:key/<your-kms-key>"
    ]
}
```

Amazon Redshift と SageMaker AI が前述のロールを引き受けて他のサービスとやり取りできるようにするには、IAM ロールに以下の信頼ポリシーを追加します。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "redshift.amazonaws.com",
          "sagemaker.amazonaws.com",
          "forecast.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Amazon S3 バケット `redshift-downloads/redshift-ml/` は、他のステップや例で使用されるサンプルデータが保存される場所です。Amazon S3 からデータをロードする必要がない場合は、削除できます。または、Amazon Redshift にデータをロードするために使用する他の Amazon S3 バケットに置き換えます。

`your-account-id`、**`your-role`**、および **`your-s3-bucket`** の値は、CREATE MODEL コマンドで指定したアカウント ID、ロール、およびバケットです。

(任意) Amazon Redshift ML で使用する AWS KMS キーを指定する場合は、サンプルポリシーの AWS KMS キーセクションを使用できます。**`your-kms-key`** 値は、CREATE MODEL コマンドの一部として使用するキーです。

ハイパーパラメータ調整ジョブにプライベート仮想プライベートクラウド (VPC) を指定する場合は、次のアクセス許可を追加します。

```
{
            "Effect": "Allow",
            "Action": [
            "ec2:CreateNetworkInterface",
            "ec2:CreateNetworkInterfacePermission",
            "ec2:DeleteNetworkInterface",
```

```
            "ec2:DeleteNetworkInterfacePermission",
            "ec2:DescribeNetworkInterfaces",
            "ec2:DescribeVpcs",
            "ec2:DescribeDhcpOptions",
            "ec2:DescribeSubnets",
            "ec2:DescribeSecurityGroups"
            ]
}
```

モデルの説明を使用するには、SageMaker AI API オペレーションを呼び出すためのアクセス許可があることを確認します。AmazonSageMakerFullAccess 管理ポリシーを使用することをお勧めします。より制限の厳しいポリシーを持つ IAM ロールを作成する場合、以下のポリシーを使用します。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sagemaker:CreateEndpoint",
        "sagemaker:CreateEndpointConfig",
        "sagemaker:DeleteEndpoint",
        "sagemaker:DeleteEndpointConfig",
        "sagemaker:DescribeEndpoint",
        "sagemaker:DescribeEndpointConfig",
        "sagemaker:DescribeModel",
        "sagemaker:InvokeEndpoint",
        "sagemaker:ListTags"
      ],
      "Resource": "*"
    }
  ]
}
```

AmazonSageMakerFullAccess マネージドポリシーの詳細については、「Amazon SageMaker 開発者ガイド」の「AmazonSageMakerFullAccess」を参照してください。

予測モデルを作成する場合は、AmazonForecastFullAccess マネージドポリシーを使用することをお勧めします。より制限の厳しいポリシーを使用する場合、IAM ロールに以下のポリシーを追加します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "forecast:CreateAutoPredictor",
                "forecast:CreateDataset",
                "forecast:CreateDatasetGroup",
                "forecast:CreateDatasetImportJob",
                "forecast:CreateForecast",
                "forecast:CreateForecastExportJob",
                "forecast:DeleteResourceTree",
                "forecast:DescribeAutoPredictor",
                "forecast:DescribeDataset",
                "forecast:DescribeDatasetGroup",
                "forecast:DescribeDatasetImportJob",
                "forecast:DescribeForecast",
                "forecast:DescribeForecastExportJob",
                "forecast:StopResource",
                "forecast:TagResource",
                "forecast:UpdateDatasetGroup"
             ],
             "Resource": "*"
         }
    ]
}
```

Amazon Bedrock モデルを作成する場合は、AmazonBedrockFullAccess マネージドポリシーを使用することをお勧めします。より制限の厳しいポリシーを使用する場合、IAM ロールに以下のポリシーを追加します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "bedrock:InvokeModel",
            "Resource": [
                "*",
                "arn:aws:bedrock:us-east-1::foundation-model/*"
            ]
        }
    ]
}
```

Amazon Redshift ML の詳細については、「Amazon Redshift での機械学習の使用」、「CREATE MODEL」、または「CREATE EXTERNAL MODEL」を参照してください。

ストリーミング取り込みのアクセス許可

ストリーミング取り込みは 2 つのサービスで機能します。Kinesis Data Streams と Amazon MSK です。

Kinesis Data Streams でストリーミング取り込みを使用するために必要なアクセス許可

マネージド ポリシーの例を含む手順については、「Amazon Kinesis Data Streams からストリーミング取り込みを開始する方法」を参照してください。

Amazon MSK でストリーミング取り込みを使用するために必要なアクセス許可

マネージド ポリシーの例を含む手順については、「Amazon Managed Streaming for Apache Kafka からのストリーミング取り込みを開始する」を参照してください。

データ共有 API オペレーションを使用するために必要な許可

データ共有 API オペレーションへのアクセスを制御するには、IAM アクションベースのポリシーを使用します。IAM ポリシーの管理方法については、「IAM ユーザーガイド」の「IAM ポリシーの管理」を参照してください。

特に、プロデューサークラスターの管理者が、AWS アカウントアカウント外のデータ共有のエグレスを承認するために AuthorizeDataShare 呼び出しを使用する必要があるとします。この場合、IAM アクションベースのポリシーを設定して、必要なアクセス許可を付与します。エグレスの取り消しには、DeauthorizeDataShare 呼び出しを使用します。

IAM アクションベースのポリシーを使用するときは、ポリシーで DataShareARNなどの IAM リソースを指定することもできます。以下は、DataShareARNの形式と例を示しています。

```
arn:aws:redshift:region:account-id:datashare:namespace-guid/datashare-name
arn:aws:redshift:us-east-1:555555555555:datashare:86b5169f-01dc-4a6f-9fbb-e2e24359e9a8/
SalesShare
```

IAM ポリシーでデータ共有名を指定することで、特定のデータ共有への AuthorizeDataShare アクセスを制限することができます。

```
{
  "Statement": [
    {
      "Action": [
        "redshift:AuthorizeDataShare",
      ],
      "Resource": [
        "arn:aws:redshift:us-east-1:555555555555:datashare:86b5169f-01dc-4a6f-9fbb-
e2e24359e9a8/SalesShare"
      ],
      "Effect": "Deny"
    }
  ]
}
```

また、IAM ポリシーを、特定のプロデューサクラスターが所有するすべてのデータ共有に制限して使用することも可能です。これを行うには、そのポリシー内の **datashare-name** 値を、ワイルドカードまたはアスタリスクに置き換えます。クラスターの namespace-guid 値はそのままにしておきます。

```
arn:aws:redshift:us-east-1:555555555555:datashare:86b5169f-01dc-4a6f-9fbb-e2e24359e9a8/
*
```

以下は、エンティティが、特定のプロデューサークラスターが所有するデータ共有上の AuthorizeDataShare を呼び出せないようにする IAM ポリシーの例です。

```
{
  "Statement": [
    {
      "Action": [
        "redshift:AuthorizeDataShare",
      ],
      "Resource": [
        "arn:aws:redshift:us-east-1:555555555555:datashare:86b5169f-01dc-4a6f-9fbb-
e2e24359e9a8/*"
      ],
      "Effect": "Deny"
    }
  ]
}
```

DataShareARN は、データ共有名と、所有クラスターの名前空間のグローバル一意 ID (GUID) の両方に基づいてアクセスを制限します。この制限は、名前をアスタリスクにより指定することによって実施されています。

GetClusterCredentials のリソースポリシー

JDBC または ODBC 接続と IAM データベース認証情報を使用してクラスターデータベースに接続するか、GetClusterCredentials アクションをプログラムで呼び出すには、redshift:GetClusterCredentials アクションを呼び出すアクセス許可と dbuser リソースへのアクセス権が必要です。

JDBC または ODBC 接続を使用する場合、serverと port の代わりに cluster_id と regionを指定できますが、そのためには、ポリシーで redshift:DescribeClusters アクションと cluster リソースへのアクセス権を許可する必要があります。

オプションパラメータ Autocreate、DbGroups、および DbNameを使用して GetClusterCredentials を呼び出す場合は、アクションも許可し、以下の表に一覧表示されているリソースへのアクセスを許可してください。

GetClusterCredentials パラメータ	アクション	リソース
Autocreate	redshift:CreateClusterUser	dbuser
DbGroups	redshift:JoinGroup	dbgroup
DbName	NA	dbname

リソースの詳細については、「Amazon Redshift のリソースとオペレーション」を参照してください。

また、次の条件をポリシーに含めることもできます。

- redshift:DurationSeconds
- redshift:DbName
- redshift:DbUser

> **Important**
>
> SAML SSO 統合では、${redshift:DbUser} 変数を使用して IAM ポリシーを指定する必要がある場合があります。このような場合は、発信者が AWS ユーザー ID と一致しないユーザーの認証情報を取得できないようにする条件ステートメントを使用することを強くお勧めします。例: "StringEquals": {"aws:userid":"AIDIODR4TAW7CSEXAMPLE:${redshift:DbUser}"}" 「例 8: GetClusterCredentials を使用する IAM ポリシー」を参照してください。条件の詳細については、「ポリシーでの条件を指定する」を参照してください

お客様が管理するポリシーの例

このセクションでは、さまざまな Amazon Redshift アクションのアクセス権限を付与するユーザーポリシー例を示しています。これらのポリシーは、Amazon Redshift API、AWS SDK、または AWS CLIを使用しているときに機能します。

> **Note**
>
> すべての例で、米国西部 (オレゴン) リージョン (us-west-2) を使用し、架空のアカウント ID を使用しています。

例 1: ユーザーにすべての Amazon Redshift アクションとリソースへの完全アクセスを許可する

次のポリシーでは、すべてのリソースですべての Amazon Redshift アクションにアクセスできます。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid":"AllowRedshift",
      "Action": [
        "redshift:*"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

Action 要素の値 redshift:* は、Amazon Redshift のすべてのアクションを示します。

例 2: 一連の Amazon Redshift アクションへのユーザーアクセスを拒否する

デフォルトでは、すべてのアクセス権が拒否されます。ただし、特定のアクションまたは一連のアクションへのアクセスを明示的に拒否しなければならない場合もあります。次のポリシーでは、すべての Amazon Redshift アクションへのアクセスが許可され、名前が Deleteで始まる Amazon

Redshift アクションへのアクセスが明示的に拒否されます。このポリシーは、us-west-2のすべての Amazon Redshift リソースに適用されます。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid":"AllowUSWest2Region",
      "Action": [
        "redshift:*"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:redshift:us-east-1:111122223333:*"
    },
   {
     "Sid":"DenyDeleteUSWest2Region",
     "Action": [
        "redshift:Delete*"
      ],
      "Effect": "Deny",
      "Resource": "arn:aws:redshift:us-east-1:111122223333:*"
   }
  ]
}
```

例 3: ユーザーがクラスターを管理することを許可する

次のポリシーでは、ユーザーはすべてのクラスターを作成、削除、変更、再起動できます。さらに、名前が protectedで始まるクラスターを削除するアクセス権限が拒否されます。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid":"AllowClusterManagement",
      "Action": [
```

```
        "redshift:CreateCluster",
        "redshift:DeleteCluster",
        "redshift:ModifyCluster",
        "redshift:RebootCluster"
      ],
      "Resource": [
        "*"
      ],
      "Effect": "Allow"
    },
    {
      "Sid":"DenyDeleteProtected",
      "Action": [
        "redshift:DeleteCluster"
      ],
      "Resource": [
        "arn:aws:redshift:us-west-2:123456789012:cluster:protected*"
      ],
      "Effect": "Deny"
    }
  ]
}
```

例 4: ユーザーにスナップショットアクセスの許可と取り消し

次のポリシーでは、ユーザー (例: ユーザー A) に次の操作を許可します。

- sharedというクラスターから作成されたスナップショットへのアクセスを許可します。
- shared クラスターから作成されたスナップショット (スナップショット名の先頭が revokable) のスナップショットアクセスを取り消します。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid":"AllowSharedSnapshots",
      "Action": [
        "redshift:AuthorizeSnapshotAccess"
```

```
      ],
      "Resource": [
        "arn:aws:redshift:us-west-2:123456789012:shared/*"
      ],
      "Effect": "Allow"
    },
    {
      "Sid":"AllowRevokableSnapshot",
      "Action": [
        "redshift:RevokeSnapshotAccess"
      ],
      "Resource": [
        "arn:aws:redshift:us-west-2:123456789012:snapshot:*/revokable*"
      ],
      "Effect": "Allow"
    }
  ]
}
```

ユーザー A がスナップショットへのアクセスをユーザー B に許可した場合、スナップショットからクラスターを復元できるように、ユーザー B には次のようなポリシーを与える必要があります。次のポリシーでは、ユーザー B はスナップショットを表現したり、スナップショットから復元したり、クラスターを作成したりできます。これらのクラスター名の先頭は `from-other-account` にする必要があります。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid":"AllowDescribeSnapshots",
      "Action": [
        "redshift:DescribeClusterSnapshots"
      ],
      "Resource": [
        "*"
      ],
      "Effect": "Allow"
    },
    {
```

```
      "Sid":"AllowUserRestoreFromSnapshot",
      "Action": [
        "redshift:RestoreFromClusterSnapshot"
      ],
      "Resource": [
        "arn:aws:redshift:us-west-2:123456789012:snapshot:*/*",
        "arn:aws:redshift:us-west-2:444455556666:cluster:from-other-account*"
      ],
      "Effect": "Allow"
    }
  ]
}
```

例 5: クラススナップショットのコピーとスナップショットからのクラスターの復元をユーザーに許可する

次のポリシーでは、ユーザーは big-cluster-1という名前のクラスターから作成したスナップショットをコピーし、名前が snapshot-for-restoreで始まるスナップショットを復元できます。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid":"AllowCopyClusterSnapshot",
      "Action": [
        "redshift:CopyClusterSnapshot"
      ],
      "Resource": [
        "arn:aws:redshift:us-west-2:123456789012:snapshot:big-cluster-1/*"
      ],
      "Effect": "Allow"
    },
    {
      "Sid":"AllowRestoreFromClusterSnapshot",
      "Action": [
        "redshift:RestoreFromClusterSnapshot"
      ],
      "Resource": [
```

```
        "arn:aws:redshift:us-west-2:123456789012:snapshot:*/snapshot-for-
restore*",
        "arn:aws:redshift:us-west-2:123456789012:cluster:*"
      ],
      "Effect": "Allow"
    }
  ]
}
```

例 6: ユーザーの Amazon Redshift と関連する AWS のサービスの一般的なアクションとリソースへのアクセスを許可する

次のポリシー例では、Amazon Redshift、Amazon Simple Notification Service (Amazon SNS)、および Amazon CloudWatch のすべてのアクションとリソースへのアクセスを許可します。アカウントにおける関連する Amazon EC2 リソースでの指定されたアクションも許可します。

> **Note**
>
> リソースレベルのアクセス権限は、このサンプルポリシーで指定された Amazon EC2 アクションではサポートされません。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid":"AllowRedshift",
      "Effect": "Allow",
      "Action": [
        "redshift:*"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Sid":"AllowSNS",
      "Effect": "Allow",
```

```
        "Action": [
          "sns:*"
        ],
        "Resource": [
          "*"
        ]
      },
    {
      "Sid":"AllowCloudWatch",
      "Effect": "Allow",
      "Action": [
        "cloudwatch:*"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Sid":"AllowEC2Actions",
      "Effect": "Allow",
      "Action": [
        "ec2:AllocateAddress",
        "ec2:AssociateAddress",
        "ec2:AttachNetworkInterface",
        "ec2:DescribeAccountAttributes",
        "ec2:DescribeAddresses",
        "ec2:DescribeAvailabilityZones",
        "ec2:DescribeInternetGateways",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeSubnets",
        "ec2:DescribeVpcs"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

例 7: ユーザーが Amazon Redshift コンソールを使用してリソースにタグを付けることを許可する

次のポリシー例では、ユーザーが AWS Resource Groupsを使用して Amazon Redshift コンソールでリソースにタグを付けることを許可しています。このポリシーは、新しい Amazon Redshift コンソールを呼び出すユーザーロールにアタッチできます。タグ付けの詳細については、「Amazon Redshift のリソースのタグ付け」を参照してください。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid":"TaggingPermissions",
            "Effect": "Allow",
            "Action": [
                "redshift:DeleteTags",
                "redshift:CreateTags",
                "redshift:DescribeTags",
                "tag:UntagResources",
                "tag:TagResources"
            ],
            "Resource": "*"
        }
    ]
}
```

例 8: GetClusterCredentials を使用する IAM ポリシー

以下のポリシーでは、これらのサンプルパラメータの値を使用します。

- リージョン : us-west-2
- AWS アカウント 123456789012
- クラスター名 : examplecluster

次のポリシーでは GetCredentials、CreateClusterUser、および JoinGroup アクションが有効になります。このポリシーでは、AWS ユーザー ID が "AIDIODR4TAW7CSEXAMPLE:${redshift:DbUser}@yourdomain.com"に一致する場合のみ条件キーが使用され、GetClusterCredentials と CreateClusterUser アクションが許可されます。IAM アク

セスは、"testdb" データベース対してのみ要求されます。また、このポリシーは、ユーザーが "common_group"という名前のグループに参加することを許可します。

JSON

```
{
"Version":"2012-10-17",
  "Statement": [
    {
     "Sid":"GetClusterCredsStatement",
      "Effect": "Allow",
      "Action": [
        "redshift:GetClusterCredentials"
      ],
      "Resource": [
        "arn:aws:redshift:us-west-2:123456789012:dbuser:examplecluster/
${redshift:DbUser}",
        "arn:aws:redshift:us-west-2:123456789012:dbname:examplecluster/testdb",
        "arn:aws:redshift:us-west-2:123456789012:dbgroup:examplecluster/
common_group"
      ],
        "Condition": {
           "StringEquals": {
           "aws:userid":"AIDIODR4TAW7CSEXAMPLE:${redshift:DbUser}@yourdomain.com"
           }
        }
    },
    {
      "Sid":"CreateClusterUserStatement",
      "Effect": "Allow",
      "Action": [
        "redshift:CreateClusterUser"
      ],
      "Resource": [
        "arn:aws:redshift:us-west-2:123456789012:dbuser:examplecluster/
${redshift:DbUser}"
      ],
      "Condition": {
        "StringEquals": {
          "aws:userid":"AIDIODR4TAW7CSEXAMPLE:${redshift:DbUser}@yourdomain.com"
        }
      }
    },
```

```
    {
      "Sid":"RedshiftJoinGroupStatement",
      "Effect": "Allow",
      "Action": [
        "redshift:JoinGroup"
      ],
      "Resource": [
        "arn:aws:redshift:us-west-2:123456789012:dbgroup:examplecluster/
common_group"
      ]
    }
  ]
}
```

次の例では、IAM ロールで GetClusterCredentials オペレーションの呼び出しを許可するポリシーを示します。Amazon Redshift dbuser リソースを指定すると、 examplecluster というクラスターで、データベースユーザー名 temp_creds_user にロールのアクセスが付与されます。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "redshift:GetClusterCredentials",
    "Resource": "arn:aws:redshift:us-west-2:123456789012:dbuser:examplecluster/
temp_creds_user"
  }
}
```

ワイルドカード (*) を使用して、クラスター名、ユーザー名、およびデータベースグループ名のすべてまたは一部を置き換えることができます。次の例では、指定したアカウントのすべてのクラスターで、temp_ で始まるすべてのユーザー名が許可されます。

> **Important**
>
> 次の例のステートメントでは、指定された文字から始まるすべてのリソースがポリシーで許可されるように、リソースの値の一部として、ワイルドカード文字 (*) を指定します。IAM ポリシーでワイルドカード文字を使用すると、過度に許可される場合があります。ベストプ

ラクティスとして、ビジネスアプリケーションを最も制限する現実的なポリシーを使用することをお勧めします。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "redshift:GetClusterCredentials",
    "Resource": "arn:aws:redshift:us-west-2:123456789012:dbuser:*/temp_*"
  }
}
```

次の例では、IAM ロールが GetClusterCredentials オペレーションを呼び出すことを許可するポリシーを示します。オプションで新しいユーザーを自動的に作成し、ログイン時にユーザーが参加するグループを指定することができます。"Resource": "*" 句により、クラスター、データベースユーザー、またはユーザーグループを含めて、任意のリソースへのロールアクセスが許可されます。

Important

次の例の ステートメントでは、ワイルドカード文字 (*) を特定のアクションのリソースとして指定します。これにより、ポリシーは任意のクラスターおよびデータベースユーザーへのアクセスを許可し、任意のユーザーの作成を許可します。IAM ポリシーでワイルドカード文字を使用すると、過度に許可される場合があります。ベストプラクティスとして、ビジネスアプリケーションを最も制限する現実的なポリシーを使用することをお勧めします。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
```

```
            "redshift:GetClusterCredentials",
            "redshift:CreateClusterUser",
  "redshift:JoinGroup"
          ],
    "Resource": "*"
  }
}
```

詳細については、Amazon Redshift ARN 構文を参照してください。

Amazon Redshift 用のネイティブ ID プロバイダー (IdP) フェデレーション

Amazon Redshift の ID とアクセス許可の管理は、既存の ID プロバイダーを活用して認証とアクセス許可の管理を簡素化するため、ネイティブ ID プロバイダーフェデレーションを使用することで簡単になります。これは、ID プロバイダーから Redshift に ID メタデータを共有できるようにすることで実現します。この機能の最初のイテレーションでは、サポートされる ID プロバイダーは Microsoft Azure Active Directory (Azure AD) です。

サードパーティーの ID プロバイダーからの ID を認証できるように Amazon Redshift を設定するには、ID プロバイダーを Amazon Redshift に登録します。これにより、Redshift は ID プロバイダーによって定義されたユーザーとロールを認証できます。したがって、アイデンティティ情報が共有されるため、サードパーティ ID プロバイダーと Amazon Redshift の両方できめ細かなアイデンティティ管理を実行する必要がなくなります。

ID プロバイダー (IdP) グループから転送されるセッションロールの使用の詳細については、「Amazon Redshift データベースデベロッパーガイド」の「PG_GET_SESSION_ROLES」を参照してください。

ネイティブ ID プロバイダー (IdP) フェデレーション

ID プロバイダーと Amazon Redshift の間の事前設定を完了するには、いくつかの手順を実行します。まず、Amazon Redshift をサードパーティーアプリケーションとして ID プロバイダーに登録し、必要な API アクセス許可を要求します。次に、ID プロバイダーでユーザーとグループを作成します。最後に、ID プロバイダーに固有の認証パラメータを設定する SQL ステートメントを使用して、ID プロバイダーを Amazon Redshift に登録します。Redshift に ID プロバイダーを登録する一環として、ユーザーおよびロールが正しくグループ化されていることを確認するために名前空間を割り当てます。

Amazon Redshift に登録された ID プロバイダーでは、Redshift と ID プロバイダー間の通信が設定されます。その後、クライアントはトークンを渡し、ID プロバイダーエンティティとして Redshift

に認証できます。Amazon Redshift は、IdP グループのメンバーシップ情報を使用して Redshift ロールにマッピングします。ユーザーが Redshift にまだ存在しない場合、ユーザーが作成されます。ID プロバイダーグループが存在しない場合は、それらにマップするロールが作成されます。Amazon Redshift 管理者はロールに対するアクセス許可を付与し、ユーザーはクエリを実行したり、他のデータベースタスクを実行したりできます。

次の手順では、ユーザーがログインしたときにネイティブ ID プロバイダーフェデレーションがどのように機能するかを概説します。

1. ユーザーがネイティブ IdP オプションを使用してログインすると、ID プロバイダーのトークンがクライアントからドライバーに送信されます。
2. ユーザーは認証されます。ユーザーが Amazon Redshift に存在していない場合は、新しいユーザーが作成されます。Redshift は、ユーザーの ID プロバイダーグループを Redshift ロールにマップします。
3. 権限は、ユーザーの Redshift ロールに基づいて割り当てられます。これらは、管理者によってユーザーとロールに付与されます。
4. ユーザーは Redshift をクエリできます。

デスクトップクライアントツール

ネイティブ ID プロバイダーフェデレーションを使用して Power BI で Amazon Redshift に接続する方法については、ブログ記事「Integrate Amazon Redshift native IdP federation with Microsoft Azure Active Directory (AD) and Power BI」(Amazon Redshift ネイティブ IdP フェデレーションと Microsoft Azure Active Directory (AD) および Power BI を統合する) を参照してください。Azure AD を使用した Amazon Redshift ネイティブ IdP セットアップの段階的な実装について説明します。Power BI デスクトップまたは Power BI サービスのクライアント接続を設定する手順についても説明します。この手順には、アプリケーションの登録、アクセス許可の設定、認証情報の設定が含まれます。

Power BI Desktop と JDBC クライアント-SQL Workbench/J を使用して Amazon Redshift ネイティブ IdP フェデレーションを Azure AD と統合する方法については、次の動画をご覧ください。

ネイティブ ID プロバイダーフェデレーションを使用して SQL クライアント (特に DBeaver または SQL Workbench/J) で Amazon Redshift に接続する方法については、ブログ記事「Integrate Amazon Redshift native IdP federation with Microsoft Azure AD using a SQL client」(SQL クライアントを使用して Amazon Redshift ネイティブ IdP フェデレーションを Microsoft Azure AD と統合する) を参照してください。

制限事項

次のとおりの制限が適用されます。

- Amazon Redshift ドライバーは、次のバージョンから `BrowserIdcAuthPlugin` をサポートしています。
  - Amazon Redshift JDBC ドライバー v2.1.0.30
  - Amazon Redshift ODBC ドライバー v2.1.3
  - Amazon Redshift Python ドライバー v2.1.3
- Amazon Redshift ドライバーは、次のバージョンから `IdpTokenAuthPlugin` をサポートしています。
  - Amazon Redshift JDBC ドライバー v2.1.0.19
  - Amazon Redshift JDBC ドライバー v2.0.0.9
  - Amazon Redshift Python ドライバー v2.0.914
- 拡張 VPC ルーティングのサポートなし – ネイティブ ID プロバイダーフェデレーションを使用する場合、拡張 VPC ルーティングはサポートされません。拡張 VPC ルーティングの詳細については、「Amazon Redshift の拡張 VPC ルーティング」を参照してください。
- AWS IAM アイデンティティセンターのキャッシュ – AWS IAM アイデンティティセンターはセッション情報をキャッシュします。これにより、Redshift クエリエディタ v2 を介して Redshift データベースに接続しようとすると、予期しないアクセス問題が発生する可能性があります。これは、データベースユーザーが AWS コンソールからサインアウトした後でも、クエリエディタ v2 の関連付けられた AWS IAM アイデンティティセンターのセッションは有効なまま残るためです。キャッシュは 1 時間後に期限切れになるため、通常は、問題が修正されます。

Amazon Redshift で ID プロバイダーを設定する

このセクションでは、ネイティブ ID プロバイダーフェデレーションの通信を確立するように ID プロバイダーと Amazon Redshift を設定する手順を示します。ID プロバイダーの有効なアカウントが必要です。Amazon Redshift を設定する前に、Redshift をアプリケーションとして ID プロバイダーに登録し、管理者の同意を付与します。

Amazon Redshift で次の手順を完了します。

1. SQL ステートメントを実行して、Azure アプリケーションメタデータの説明を含む ID プロバイダーを登録します。ID プロバイダーを Amazon Redshift で作成するには、パラメータ値を置き換えた後に次のコマンドを実行します。issuer、client_id、client_secret、および audience。

これらのパラメータは Microsoft Azure AD 固有です。ID プロバイダー名を任意の名前に置き換え、名前空間を ID プロバイダーディレクトリからユーザーとロールを含む一意の名前に置き換えます。

```
CREATE IDENTITY PROVIDER oauth_standard TYPE azure
NAMESPACE 'aad'
PARAMETERS '{
"issuer":"https://sts.windows.net/2sdfdsf-d475-420d-b5ac-667adad7c702/",
"client_id":"<client_id>",
"client_secret":"BUAH~ewrqewrqwerUUY^%tHe1oNZShoiU7",
"audience":["https://analysis.windows.net/powerbi/connector/AmazonRedshift"]
}'
```

タイプ azure は、プロバイダーが特に Microsoft Azure AD との通信を容易にすることを示します。Azure は現在、唯一サポートされている ID プロバイダーです。

- issuer – トークンが受信されたときに信頼する発行者 ID。tenant_id の一意の識別子が発行者に付加されます。
- client_id – ID プロバイダーに登録されたアプリケーションの一意の公開識別子。これは、アプリケーション ID と呼びます。
- client_secret – ID プロバイダーと登録済みアプリケーションのみが認識するシークレット識別子、またはパスワード。
- audience – Azure でアプリケーションに割り当てられているアプリケーション ID。

共有クライアントシークレットを使用する代わりに、ID プロバイダーを作成する場合、証明書、プライベートキー、およびプライベートキーのパスワードを指定するパラメータを設定できます。

```
CREATE IDENTITY PROVIDER example_idp TYPE azure
NAMESPACE 'example_aad'
PARAMETERS '{"issuer":"https://sts.windows.net/2sdfdsf-d475-420d-
b5ac-667adad7c702/",
"client_id":"<client_id>",
"audience":["https://analysis.windows.net/powerbi/connector/AmazonRedshift"],
"client_x5t":"<certificate thumbprint>",
"client_pk_base64":"<private key in base64 encoding>",
"client_pk_password":"test_password"}';
```

プライベートキーのパスワードの client_pk_password は省略可能です。

2. オプション: Amazon Redshift で SQL コマンドを実行して、ユーザーとロールを事前に作成します。これにより、事前に権限を付与することが容易になります。Amazon Redshift でのロール名は次のようなものです。<Namespace>: <GroupName on Azure AD>。例えば、Microsoft Azure AD で名前空間が aad で、ロール名は aad:rsgroup の rsgroup といういうグループを作成するとします。Amazon Redshift のユーザー名とロール名は、ID プロバイダーの名前空間のこのようなユーザー名とグループメンバーシップから定義されます。

   ロールとユーザーのマッピングには、external_id 値が最新であることを確認する検証が含まれます。外部 ID は、ID プロバイダー内のグループまたはユーザーの識別子にマップされます。たとえば、ロールの外部 ID は、対応する Azure AD グループ ID にマップされます。同様に、各ユーザーの外部 ID は ID プロバイダーの ID にマップされます。

   ```
   create role "aad:rsgroup";
   ```

3. 要件に応じて、関連する権限をロールに付与します。例:

   ```
   GRANT SELECT on all tables in schema public to role "aad:rsgroup";
   ```

4. さらに、特定のユーザーにアクセス権限を付与することもできます。

   ```
   GRANT SELECT on table foo to aad:alice@example.com
   ```

   フェデレーション外部ユーザーのロールメンバーシップは、そのユーザーのセッションでのみ利用可能であることに注意してください。これはデータベースオブジェクト作成と密接に関連しています。例えば、フェデレーション外部ユーザーがビューやストアドプロシージャを作成した場合、その同じユーザーがそれらのオブジェクトの権限を他のユーザーやロールに委任することはできません。

名前空間の説明

名前空間は、ユーザーまたはロールを特定の ID プロバイダーにマッピングします。例えば、AWS IAM で作成したユーザーのプレフィックスは iam: です。このプレフィックスは、ユーザー名の衝突を防ぎ、複数の ID ストアをサポートできるようにします。aad 名前空間に登録されたアイデンティティソースからユーザー alice@example.com がログインしたが、そのユーザーが存在していない場合は Redshift でユーザー aad:alice@example.com が作成されます。ユーザーおよびロール

の名前空間には、クラスターに関連付けられた一意の識別子であるAmazon Redshift クラスター名前空間とは異なる機能があることに注意してください。

ID プロバイダーの Amazon Redshift ロールの自動作成

この機能を使用すると、ID プロバイダー (IdP) からのグループメンバーシップに基づいて、Redshift でロールを自動作成できます。ロールの自動作成は、ネイティブ IdP 統合による Azure Active Directory をサポートしています。

ロールの自動作成には、いくつかの利点があります。ロールを自動作成すると、Redshift は IdP にグループメンバーシップを持つロールを作成します。これにより、面倒な手動によるロールの作成とメンテナンスを行う必要がなくなります。また、Redshift ロールにマッピングされるグループをフィルタリングすることもできます。

仕組み

IdP ユーザーとして Redshift にログインすると、次の一連のイベントが発生します。

1. Redshift は、IdP からグループメンバーシップを取得します。
2. Redshift は、ロールの形式 *idp_namespace*:*rolename* を使用して、これらのグループへのロールマッピングを自動作成します。
3. Redshift は、マッピングされたロールに対するアクセス許可を付与します。

各ユーザーがログインするたびに、カタログには存在しないが、そのユーザーが所属するグループは自動作成されます。オプションで、Redshift ロールが作成された IdP グループを制御するため、含有フィルターと除外フィルターを設定できます。

自動作成ロールの設定

CREATE IDENTITY PROVIDER および ALTER IDENTITY PROVIDER コマンドを使用して、ロールの自動作成を有効にして設定します。

```
-- Create a new IdP with auto role creation enabled
CREATE IDENTITY PROVIDER <idp_name> TYPE azure
  NAMESPACE '<namespace>'
  APPLICATION_ARN 'app_arn'
  IAM_ROLE 'role_arn'
  AUTO_CREATE_ROLES TRUE;

-- Enable on existing IdP
```

```
ALTER IDENTITY PROVIDER <idp_name>
  AUTO_CREATE_ROLES TRUE;

-- Disable
ALTER IDENTITY PROVIDER <idp_name>
  AUTO_CREATE_ROLES FALSE;
```

グループのフィルタリング

必要に応じて、INCLUDE パターンおよび EXCLUDE パターンを使用して、Redshift ロールにマッピングされる IdP グループをフィルタリングできます。パターンが競合した場合、EXCLUDE は INCLUDE よりも優先されます。

```
-- Only create roles for groups with 'dev'
CREATE IDENTITY PROVIDER <idp_name> TYPE azure
  ...
  AUTO_CREATE_ROLES TRUE
  INCLUDE GROUPS LIKE '%dev%';

-- Exclude 'test' groups
ALTER IDENTITY PROVIDER <idp_name>
  AUTO_CREATE_ROLES TRUE
  EXCLUDE GROUPS LIKE '%test%';
```

例

次の例は、フィルタリングなしでロールの自動作成を有効にする方法を示しています。

```
CREATE IDENTITY PROVIDER prod_idc TYPE azure ...
  AUTO_CREATE_ROLES TRUE;
```

次の例では、開発グループは含まれ、テストグループは除外されます。

```
ALTER IDENTITY PROVIDER prod_idc
  AUTO_CREATE_ROLES TRUE
  INCLUDE GROUPS LIKE '%dev%'
  EXCLUDE GROUPS LIKE '%test%';
```

ベストプラクティス

ロールの自動作成を有効にする場合は、次のベストプラクティスを考慮してください。

- INCLUDE および EXCLUDE フィルターを使用して、ロールを取得するグループを制御します。
- 定期的にロールを監査し、未使用のロールをクリーンアップします。
- Redshift ロール階層を活用して、アクセス許可の管理を簡素化します。

Redshift を AWS IAM アイデンティティセンターに接続してシングルサインオンエクスペリエンスを提供する

信頼されたアイデンティティのプロパゲーションを通じて、Amazon Redshift データウェアハウスへのユーザーとグループのアクセスを管理できます。

信頼された ID の伝播は、接続された AWS のサービス の管理者がサービスデータへのアクセスを許可および監査するために使用できる AWS IAM アイデンティティセンター 機能です。このデータへのアクセスは、グループの関連付けなどのユーザー属性に基づいています。信頼された ID の伝播を設定するには、接続された AWS のサービス の管理者と IAM アイデンティティセンターの管理者とのコラボレーションが必要です。詳細については、「Prerequisites and considerations」を参照してください。

エンドツーエンドの例を説明すると、Amazon Quick ダッシュボードまたは Amazon Redshift クエリエディタ v2 を使用して Redshift にアクセスできます。この場合のアクセスは AWS IAM アイデンティティセンターグループに基づいています。Redshift は、ユーザーが誰かや、ユーザーのグループメンバーシップを判断できます。AWSIAM アイデンティティセンターを使用すると、Okta や PingOne などのサードパーティー ID プロバイダー (IdP) 経由でアイデンティティを接続して管理することもできます。

管理者は Redshift と AWS IAM アイデンティティセンター間の接続を設定した後、データへのユーザーアクセスを承認するために ID プロバイダーグループに基づいてきめ細かいアクセスを設定できます。

> **Important**
>
> AWS IAM アイデンティティセンターまたは接続された ID プロバイダー (IdP) ディレクトリからユーザーを削除しても、該当ユーザーは Amazon Redshift カタログから自動的には削除されません。Amazon Redshift カタログからユーザーを手動で削除するには、 DROP USER コマンドを実行して、AWS IAM アイデンティティセンターまたは IdP から削除されたユーザーを完全に削除します。ユーザーを削除する方法の詳細については、「Amazon Redshift データベースデベロッパーガイド」の「DROP USER」を参照してください。

Redshift と AWS IAM アイデンティティセンターの統合の利点

AWS IAM アイデンティティセンターと Redshift を連携すると、組織は次のとおりの利点が得られます。

- パスワードを再入力したり、管理者が複雑なアクセス許可を持つ IAM ロールを設定したりする必要なく、Amazon Quick のダッシュボード作成者は、Redshift データソースに接続できます。
- AWS IAM アイデンティティセンターは、 のワークフォースユーザーを一元管理する場所となります。AWSAWS IAM アイデンティティセンターを使用して、ユーザーとグループを直接作成したり、Okta、PingOne、Microsoft Entra ID (Azure AD) などの標準ベースの ID プロバイダーで管理する既存のユーザーやグループを接続したりできます。AWSIAM アイデンティティセンターは、ユーザーとグループの選択した信頼できるソースに認証を指示し、Redshift がアクセスできるようにユーザーとグループのディレクトリを管理します。詳細については、「AWS IAM アイデンティティセンターユーザーガイド」の「アイデンティティソースの管理」と「サポートされている ID プロバイダー」を参照してください。
- シンプルな自動検出および接続機能を使用して、単一の AWS IAM アイデンティティセンターインスタンスを複数の Redshift クラスターやワークグループと共有できます。これにより、クラスターごとに AWS IAM アイデンティティセンター接続を設定するという追加の労力を必要とせずにクラスターを迅速に追加できるようになり、すべてのクラスターとワークグループでユーザー、ユーザーの属性、グループを一貫して確認できるようになります。組織の AWS IAM アイデンティティセンターインスタンスは、接続先のすべての Redshift データ共有と同じリージョンにある必要があることに注意します。
- ユーザー ID は既知であり、データアクセスとともに記録されるため、AWS CloudTrail でユーザーアクセスを監査することでコンプライアンス規制への対応が容易になります。

アプリケーションの接続のための管理者ペルソナ

分析アプリケーションを Redshift 向けの AWS IAM アイデンティティセンターマネージドアプリケーションに接続するための重要なペルソナは次のとおりです。

- アプリケーション管理者 – アプリケーションを作成して、アイデンティティトークンの交換ができるようになるサービスを設定します。この管理者は、アプリケーションにアクセスできるユーザーまたはグループも指定します。
- データ管理者 – データに対するきめ細かいアクセスを設定します。AWS IAM アイデンティティセンターのユーザーとグループは、特定のアクセス許可にマッピングされます。

Amazon Quick を介した AWS IAM アイデンティティセンターでの Amazon Redshift への接続

Redshift が AWS IAM アイデンティティセンターに接続され、アクセスが管理されている場合に、Quick を使用して Redshift で認証する方法については、「Quick から Amazon Redshift クラスターへの接続の認可」で説明しています。このようなステップは Amazon Redshift Serverless にも適用されます。

Amazon Redshift クエリエディタ v2 を介した AWS IAM アイデンティティセンターでの Amazon Redshift への接続

Redshift での AWS IAM アイデンティティセンター接続のセットアップ手順を完了すると、ユーザーは AWS IAM アイデンティティセンターベースの名前空間プレフィックス付き ID を通じてデータベースとデータベース内の適切なオブジェクトにアクセスできるようになります。クエリエディタ v2 のサインインで Redshift データベースに接続する方法の詳細については、「クエリエディタ v2 を使用してデータベースのクエリを実行する」を参照してください。

複数の AWS リージョン間で AWS IAM アイデンティティセンターを使用する

Amazon Redshift は、複数の AWS リージョン間での AWS IAM アイデンティティセンターをサポートしています。AWS IAM アイデンティティセンターをプライマリの AWS リージョンから追加のリージョンに拡張して、ユーザーへの近接性と信頼性を高めてパフォーマンスを向上させることができます。AWS IAM アイデンティティセンターで新しいリージョンを追加すると、プライマリリージョンの ID をレプリケートすることなく、新しいリージョンに Redshift IAM アイデンティティセンターアプリケーションを作成できます。行レベル、列レベル、マスキングの各コントロールを有効にできる新しいリージョンで、AWS IAM アイデンティティセンターを使用して Amazon Redshift フェデレーションアクセス許可を設定できます。複数のリージョンで AWS IAM アイデンティティセンターの使用を開始する方法の詳細については、「AWSIAM アイデンティティセンターユーザーガイド」の「複数の AWS リージョンで AWS IAM アイデンティティセンターを管理する」を参照してください。

Amazon Redshift での AWS IAM アイデンティティセンターの統合の設定

Amazon Redshift クラスター管理者または Amazon Redshift Serverless 管理者は、Redshift を AWS IAM アイデンティティセンター対応アプリケーションとして設定するためにいくつかのステップを実行する必要があります。これにより、Redshift は AWS IAM アイデンティティセンターを自動的に検出して接続し、サインインサービスとユーザーディレクトリサービスを受信できるようになります。その後、Redshift 管理者がクラスターまたはワークグループを作成すると、新しいデータウェアハウスが AWS IAM アイデンティティセンターを使用してデータベースアクセスを管理できるようになります。

Redshift を AWS IAM アイデンティティセンターのマネージドアプリケーションとして有効にすべき理由は、AWS IAM アイデンティティセンター内から、または IAM アイデンティティセンターと統合されたサードパーティーの ID プロバイダーからユーザーとグループのアクセス許可を制御できるようにするためです。データベースユーザー (アナリストやデータサイエンティストなど) が Redshift データベースにサインインすると、AWS IAM アイデンティティセンターでユーザーのグループがチェックされ、これらが Redshift のロール名と一致します。Redshift データベースロールの名前を定義するグループは、このような方法で、例えば売上分析のためのテーブルセットにアクセスできます。以降のセクションでは、これを設定する方法について説明します。

前提条件

AWS IAM アイデンティティセンターを Amazon Redshift と統合するための前提条件は次のとおりです。

- アカウント設定 – クロスアカウントのユースケースを計画している場合、または同じ AWS IAM アイデンティティセンターインスタンスの別のアカウントで Redshift クラスターを使用する場合は、AWS 組織の管理アカウントで AWS IAM アイデンティティセンターを設定する必要があります。これには、アイデンティティソースの設定も含まれます。詳細については、「AWS IAM アイデンティティセンターユーザーガイド」の「開始方法」、「ワークフォースアイデンティティ」、「サポートされている ID プロバイダー」を参照してください。Redshift のデータに割り当てる前に、AWS IAM アイデンティティセンターでユーザーまたはグループを作成したか、アイデンティティソースからユーザーとグループを同期したことを確認する必要があります。

  > **Note**
  >
  > Redshift と AWS IAM アイデンティティセンターが同じアカウントにある場合は、AWS IAM アイデンティティセンターのアカウントインスタンスを使用するオプションがあります。Redshift クラスターまたはワークグループを作成したり構成したりする際には、ウィジェットを使用してこのインスタンスを作成できます。

- 信頼できるトークン発行者の設定 – 場合によっては、信頼できるトークンを発行したり検証したりできる信頼できるトークン発行者の使用が必要になることがあります。これを実行する以前に、AWS IAM アイデンティティセンターの統合を設定する Redshift 管理者が信頼できるトークン発行者を選択し、必要な属性を追加して設定を完了する前の準備ステップを実行する必要があります。この手順には、信頼できるトークン発行者となるように外部 ID プロバイダーを設定すること、AWS IAM アイデンティティセンターコンソールにその属性を追加することなどがあります。これらのステップを完了するには、「信頼できるトークン発行者によるアプリケーションの使用」を参照してください。

> **Note**
>
> 信頼できるトークン発行者の設定は、すべての外部接続で必要であるとは限りません。Amazon Redshift クエリエディタ v2 を使用して Redshift データベースに接続する場合には、信頼できるトークン発行者の設定は必要ありません。ただし、ID プロバイダーで認証するダッシュボードやカスタムアプリケーションなどのサードパーティーアプリケーションには、この設定を適用できます。

- IAM ロールの設定 – 以降のセクションで、設定する必要のあるアクセス許可について説明しています。IAM のベストプラクティスに従ってアクセス許可を追加する必要があります。特定のアクセス許可については、以降の手順で詳しく説明します。
- 拡張 VPC ルーティングのネットワーク設定 – 拡張 VPC ルーティングを有効にする場合、VPC から AWS IAM アイデンティティセンターのサービスにアクセスできる必要があります。この接続を提供するには、インターフェイス VPC エンドポイントを使用することをお勧めします。

> **Note**
>
> ユーザーがサインインする前に、この接続を設定してください。拡張 VPC ルーティングと AWS IAM アイデンティティセンターの使用の詳細については、「拡張 VPC ルーティングでの AWS IAM アイデンティティセンター認証の使用」を参照してください。

詳細については、「AWS IAM アイデンティティセンターの開始方法」を参照してください。

AWS IAM アイデンティティセンターと連携する ID プロバイダーの設定

ユーザーとグループのアイデンティティ管理を制御する最初のステップは、AWS IAM アイデンティティセンターに接続して ID プロバイダーを設定することです。AWS IAM アイデンティティセンター自体を ID プロバイダーとして使用することも、Okta などのサードパーティーのアイデンティティストアに接続することもできます。ID プロバイダーへの接続のセットアップと設定の詳細については、「AWS IAM アイデンティティセンターユーザーガイド」の「外部 ID プロバイダーに接続する」を参照してください。このプロセスの最後に、テスト目的で少数のユーザーとグループを AWS IAM アイデンティティセンターに追加していることを確認します。

管理権限

Redshift/AWS IAM アイデンティティセンターのアプリケーションライフサイクル管理に必要な権限

Redshift 管理者が IAM アイデンティティセンターで使用するように Redshift を設定するために使用する AWS IAM アイデンティティを作成する必要があります。一般的には、アクセス許可を付与した IAM ロールを作成し、必要に応じて他のアイデンティティに割り当てます。このロールには、次のアクションを実行するためのアクセス許可が必要です。

Redshift/AWS IAM アイデンティティセンターアプリケーションの作成

- `sso:PutApplicationAssignmentConfiguration` – セキュリティ用です。
- `sso:CreateApplication` – を使用 AWS IAM アイデンティティセンターのアプリケーションを作成するために使用されます。
- `sso:PutApplicationAuthenticationMethod` – Redshift 認証アクセスを許可します。
- `sso:PutApplicationGrant` – 信頼できるトークン発行者の情報を変更するために使用されます。
- `sso:PutApplicationAccessScope` – Redshift AWS IAM アイデンティティセンターのアプリケーションのセットアップ用です。これには、AWS Lake Formation 用と Amazon S3 Access Grants 用が含まれます。
- `redshift:CreateRedshiftIdcApplication` – Redshift AWS IAM アイデンティティセンターのアプリケーションを作成するために使用されます。

Redshift/AWS IAM アイデンティティセンターアプリケーションの説明

- `sso:GetApplicationGrant` – 信頼できるトークン発行者の情報を一覧表示するために使用されます。
- `sso:ListApplicationAccessScopes` — Redshift AWS IAM アイデンティティセンターアプリケーション設定で、AWS Lake Formation や S3 Access Grants などのダウンストリームインテグレーションを一覧表示できるようにします。
- `redshift:DescribeRedshiftIdcApplications` — 既存の AWS IAM アイデンティティセンターアプリケーションを説明するために使用されます。

Redshift/AWS IAM アイデンティティセンターアプリケーションの変更

- `redshift:ModifyRedshiftIdcApplication` – 既存の Redshift アプリケーションを変更するために使用されます。

- sso:UpdateApplication – AWS IAM アイデンティティセンターのアプリケーションの更新に使用されます。
- sso:GetApplicationGrant – 信頼できるトークン発行者の情報を取得します。
- sso:ListApplicationAccessScopes – Redshift AWS IAM アイデンティティセンターのアプリケーションのセットアップ用です。
- sso:DeleteApplicationGrant – 信頼できるトークン発行者の情報を削除します。
- sso:PutApplicationGrant – 信頼できるトークン発行者の情報を変更するために使用されます。
- sso:PutApplicationAccessScope – Redshift AWS IAM アイデンティティセンターのアプリケーションのセットアップ用です。これには、AWS Lake Formation 用と Amazon S3 Access Grants 用が含まれます。
- sso:DeleteApplicationAccessScope – Redshift AWS IAM アイデンティティセンターのアプリケーションの削除用です。これには、AWS Lake Formation 用と Amazon S3 Access Grants 用が含まれます。

Redshift/AWS IAM アイデンティティセンターアプリケーションの削除

- sso:DeleteApplication – AWS IAM アイデンティティセンターのアプリケーションを削除するために使用されます。
- redshift:DeleteRedshiftIdcApplication – 既存の Redshift AWS アイデンティティセンターアプリケーションを削除できます。

Redshift/クエリエディタ v2 アプリケーションのライフサイクル管理に必要な権限

Redshift 管理者が IAM アイデンティティセンターで使用するように Redshift を設定するために使用する AWS IAM アイデンティティを作成する必要があります。一般的には、アクセス許可を付与した IAM ロールを作成し、必要に応じて他のアイデンティティに割り当てます。このロールには、次のアクションを実行するためのアクセス許可が必要です。

クエリエディタ v2 アプリケーションの作成

- redshift:CreateQev2IdcApplication — QEV2 アプリケーションの作成に使用されます。
- sso:CreateApplication – AWS IAM アイデンティティセンターアプリケーションを作成する許可を付与します。
- sso:PutApplicationAuthenticationMethod – Redshift 認証アクセスを許可します。

- sso:PutApplicationGrant – 信頼できるトークン発行者の情報を変更するために使用されます。
- sso:PutApplicationAccessScope – Redshift AWS IAM アイデンティティセンターのアプリケーションのセットアップ用です。これにはクエリエディタ v2 などがあります。
- sso:PutApplicationAssignmentConfiguration – セキュリティ用です。

クエリエディタ v2 アプリケーションの説明

- redshift:DescribeQev2IdcApplications — AWS IAM アイデンティティセンター QEV2 アプリケーションを説明するために使用されます。

クエリエディタ v2 アプリケーションの変更

- redshift:ModifyQev2IdcApplication — AWS IAM アイデンティティセンター QEV2 アプリケーションを変更するために使用されます。
- sso:UpdateApplication — AWS IAM アイデンティティセンター QEV2 アプリケーションを変更するために使用されます。

クエリエディタ v2 アプリケーションの削除

- redshift:DeleteQev2IdcApplication — QEV2 アプリケーションの削除に使用されます。
- sso:DeleteApplication — QEV2 アプリケーションの削除に使用されます。

> **Note**
>
> Amazon Redshift SDK では、次の API は使用できません。
>
> - CreateQev2IdcApplication
> - DescribeQev2IdcApplications
> - ModifyQev2IdcApplication
> - DeleteQev2IdcApplication

これらのアクションは、AWS IAM アイデンティティセンターと Redshift QEV2 の統合を AWS コンソールで実行する場合に固有のものです。詳細については、「Actions defined by Amazon Redshift」を参照してください。

コンソールで新しいリソースに接続するために必要なデータベース管理者権限

このようなアクセス許可は、作成プロセス中に新たにプロビジョンされたクラスターまたは Amazon Redshift Serverless ワークグループに接続するために必要となります。このようなアクセス許可がある場合は、Redshift 用の AWS IAM アイデンティティセンターマネージドアプリケーションに接続するためのオプションがコンソールに表示されます。

- `redshift:DescribeRedshiftIdcApplications`
- `sso:ListApplicationAccessScopes`
- `sso:GetApplicationAccessScope`
- `sso:GetApplicationGrant`

ベストプラクティスとして、アクセス許可ポリシーを IAM ロールにアタッチし、それを必要に応じてユーザーやグループに割り当てることをお勧めします。詳細については、「Amazon Redshift での Identity and Access Management」を参照してください。

AWS IAM アイデンティティセンターでの AWS マネージドアプリケーションとしての Redshift の設定

AWS IAM アイデンティティセンターが Amazon Redshift でプロビジョンされたクラスターまたは Amazon Redshift Serverless ワークグループのアイデンティティを管理するには、Redshift 管理者は Redshift を AWS IAM アイデンティティセンターのマネージドアプリケーションにするステップを実行する必要があります。

1. Amazon Redshift または Amazon Redshift Serverless のコンソールメニューで [AWS IAM アイデンティティセンターの統合] を選択して、[AWS IAM アイデンティティセンターに接続] をクリックします。その後一連の選択を行い、AWS IAM アイデンティティセンター統合のプロパティを設定します。
2. Redshift AWS IAM アイデンティティセンターマネージドアプリケーションの表示名と一意の名前を選択します。

3. 組織の名前空間を指定します。これには通常、組織名の省略形を使用します。これは、AWS IAMアイデンティティセンターマネージドユーザーとロールのプレフィックスとして Redshift データベースに追加されます。
4. 使用する IAM ロールを選択します。この IAM ロールは、Redshift に使用されるその他のロールとは別に作成する必要があります。これ以外の目的には使用しないことをお勧めします。特定のポリシーに必要となるアクセス許可は次のとおりです。

   - `sso:DescribeApplication` – カタログに ID プロバイダー (IdP) エントリを作成するために必要です。
   - `sso:DescribeInstance` – IdP フェデレーションロールまたはユーザーを手動で作成する場合に使用します。
5. クライアント接続と信頼できるトークン発行者を設定します。信頼できるトークン発行者を設定すると、外部の ID プロバイダーとの関係が確立されるので、信頼できる ID のプロパゲーションが容易になります。ID のプロパゲーションにより、ユーザーは例えば、あるアプリケーションにサインインして、別のアプリケーションの特定のデータにアクセスできるようになります。これにより、ユーザーは別の場所からデータをよりシームレスに収集できます。このステップでは、信頼できる各トークン発行者の属性をコンソールで設定します。属性には名前と対象者クレーム (または aud claim) があります。これらはツールまたはサービスの設定属性から取得する必要がある場合があります。また、サードパーティーツールの JSON Web Token (JWT) からアプリケーション名を指定する必要がある場合もあります。

   > **Note**
   >
   > `aud claim` が各サードパーティーツールまたはサービスから必要とするのは、トークンタイプ (ID プロバイダーが発行するアクセストークン、または ID トークンのような別のタイプのトークン) によって異なる場合があります。各ベンダーにより異なる場合があります。信用されるアイデンティティのプロパゲーションを実装し、Redshift と統合する場合、サードパーティーツールが AWS に送信するトークンタイプに適切な aud 値を指定する必要があります。ツールまたはサービスベンダーの推奨事項を確認します。

   信頼できる ID のプロパゲーションの詳細については、AWS IAM アイデンティティセンターユーザーガイドの「信頼できる ID のプロパゲーションの概要」を参照してください。

Redshift 管理者が手順を完了して設定を保存すると、AWS IAM アイデンティティセンターのプロパティが Redshift コンソールに表示されます。システムビュー SVV_IDENTITY_PROVIDERS で

クエリを実行して、アプリケーションのプロパティを確認することもできます。プロパティには、アプリケーション名と名前空間などがあります。名前空間は、アプリケーションに関連付けられた Redshift データベースオブジェクトのプレフィックスとして使用します。上記のタスクを完了すると、Redshift は AWS IAM アイデンティティセンター対応アプリケーションになります。コンソールのプロパティには統合ステータスが含まれます。統合が完了すると、「有効」と表示されます。このプロセスが完了すると、新しいクラスターごとに AWS IAM アイデンティティセンター統合を有効にできます。

設定後、[ユーザー] または [グループ] タブをクリックして、[割り当てる] を選択すると、Redshift で AWS IAM アイデンティティセンターからユーザーとグループを含めることができます。

Amazon Redshift クラスターまたは Amazon Redshift Serverless ワークグループの AWS IAM アイデンティティセンターの統合の有効化

データベース管理者は、サインインとデータアクセスを容易にするために AWS IAM アイデンティティセンターと連携して動作するように新しい Redshift リソースを設定します。これは、プロビジョンされたクラスターまたはサーバーレスワークグループを作成する手順の一貫として実行されます。Redshift リソースを作成する権限を持つユーザーは、これらの AWS IAM アイデンティティセンター統合タスクを実行できます。プロビジョニングされたクラスターを作成するときは、まず Amazon Redshift コンソールで [クラスターの作成] を選択します。データベースの AWS IAM アイデンティティセンター管理を有効にする方法は、次の手順のとおりです。(クラスターを作成する手順がすべて含まれているわけではありません)。

1. クラスター作成ステップの [IAM アイデンティティセンターの統合]セクションで、[有効にする] を選択します。
2. 統合を有効にするプロセスにはステップがあり、これを実行するには、コンソールで [IAM アイデンティティセンターの統合を有効にする] を選択します。
3. 新しいクラスターまたはワークグループの場合、SQL コマンドを使用して Redshift でデータベースロールを作成します。このコマンドは、次のとおりです。

   ```
   CREATE ROLE <idcnamespace:rolename>;
   ```

   名前空間名とロール名は、次のとおりです。

   - IAM アイデンティティセンター名前空間プレフィックス – これは AWS IAM アイデンティティセンターと Redshift 間の接続の設定時に定義した名前空間です。
   - ロール名 – この Redshift データベースロールは AWS IAM アイデンティティセンターのグループ名と一致する必要があります。

Redshift は AWS IAM アイデンティティセンターに接続して、データベースロールを作成し、AWS IAM アイデンティティセンターグループにマッピングするために必要な情報を取得します。

新しいデータウェアハウスを作成すると、AWS IAM アイデンティティセンターの統合に指定された IAM ロールが、プロビジョンされたクラスターまたは Amazon Redshift Serverless ワークグループに自動的にアタッチされることに注意します。必要なクラスターのメタデータの入力を完了し、リソースを作成した後、プロパティで AWS IAM アイデンティティセンター統合のステータスを確認できます。AWS IAM アイデンティティセンターのグループ名にスペースが含まれている場合は、一致するロールを作成する際に SQL で引用符を使用する必要があります。

Redshift データベースを有効にしてロールを作成したら、Amazon Redshift クエリエディタ v2 または Amazon Quick を使用してデータベースに接続できるようになります。詳細については、以降のセクションで詳しく説明します。

API を使用したデフォルトの **`RedshiftIdcApplication`** の設定

セットアップはアイデンティティ管理者が行います。API を使用して、AWS IAM アイデンティティセンター内の Redshift アプリケーションを表す `RedshiftIdcApplication` を作成してデータを入力します。

1. まず、ユーザーを作成して、AWS IAM アイデンティティセンター内のグループに追加します。これは AWS IAM アイデンティティセンター の AWS コンソールで行います。
2. `create-redshift-idc-application` を呼び出して AWS IAM アイデンティティセンターアプリケーションを作成して、Redshift の使用に対応させます。必要な値を入力してアプリケーションを作成します。表示名とは AWS IAM アイデンティティセンターダッシュボードに表示される名前です。IAM ロール ARN は、AWS IAM アイデンティティセンターへのアクセス許可を持つ ARN であり、Redshift も引き受けることができます。

```
aws redshift create-redshift-idc-application
--idc-instance-arn 'arn:aws:sso:::instance/ssoins-1234a01a1b12345d'
--identity-namespace 'MYCO'
--idc-display-name 'TEST-NEW-APPLICATION'
--iam-role-arn 'arn:aws:redshift:us-east-1:012345678901:role/TestRedshiftRole'
--redshift-idc-application-name 'myredshiftidcapplication'
```

create-redshift-idc-application への呼び出しから返される RedshiftIdcApplication の応答サンプルは、次の例のとおりです。

```
"RedshiftIdcApplication": {
                "IdcInstanceArn": "arn:aws:sso:::instance/ssoins-1234a01a1b12345d",
                "RedshiftIdcApplicationName": "test-application-1",
                "RedshiftIdcApplicationArn": "arn:aws:redshift:us-
east-1:012345678901:redshiftidcapplication:12aaa111-3ab2-3ab1-8e90-b2d72aea588b",
                "IdentityNamespace": "MYCO",
                "IdcDisplayName": "Redshift-Idc-Application",
                "IamRoleArn": "arn:aws:redshift:us-east-1:012345678901:role/
TestRedshiftRole",
                "IdcManagedApplicationArn": "arn:aws:sso::012345678901:application/
ssoins-1234a01a1b12345d/apl-12345678910",
                "IdcOnboardStatus": "arn:aws:redshift:us-
east-1:123461817589:redshiftidcapplication",
                "RedshiftIdcApplicationArn": "Completed",
                "AuthorizedTokenIssuerList": [
                      "TrustedTokenIssuerArn": ...,
                      "AuthorizedAudiencesList": [...]...
                ]}
```

3. create-application-assignment を使用して、AWS IAM アイデンティティセンターマネージドアプリケーションに特定のグループまたは個別のユーザーを割り当てることができます。これにより、AWS IAM アイデンティティセンターを介して管理するグループを指定できます。データベース管理者が Redshift でデータベースロールを作成した場合、AWS IAM アイデンティティセンターのグループ名は Redshift のロール名にマップされます。ロールはデータベース内のアクセス許可を制御します。詳細については、「AWS IAM アイデンティティセンターコンソールでアプリケーションへのユーザーアクセスを割り当てる」を参照してください。
4. アプリケーションを有効にしたら、create-cluster を呼び出します。AWS IAM アイデンティティセンターからの Redshift マネージドアプリケーション ARN を含めます。これを実行すると、クラスターが AWS IAM アイデンティティセンターのマネージドアプリケーションに関連付けられます。

AWS IAM アイデンティティセンターアプリケーションの既存のクラスターまたはワークグループへの関連付け

AWS IAM アイデンティティセンターの統合を有効にする既存のクラスターまたはワークグループがある場合は、SQL コマンドを実行することでそれが可能になります。また、SQL コマンドを実行し

て統合の設定を変更することもできます。詳細については、「ALTER IDENTITY PROVIDER」を参照してください。

既存の ID プロバイダーを削除することもできます。CASCADE が ID プロバイダーにアタッチされているユーザーとロールを削除する方法は、次の例のとおりです。

```
DROP IDENTITY PROVIDER
<provider_name> [ CASCADE ]
```

ユーザーのアクセス許可のセットアップ

管理者は、ユーザーの ID 属性とグループメンバーシップに基づいて、ID プロバイダー内または AWS IAM アイデンティティセンター内で直接、さまざまなリソースへのアクセス許可を設定します。例えば、ID プロバイダーの管理者は、データベースエンジニアを役割に応じたグループに追加できます。このグループ名は Redshift データベースロール名にマップされます。このロールにより、Redshift の特定のテーブルまたはビューへのアクセスが提供されたり制限されたりします。

拡張 VPC ルーティングでの AWS IAM アイデンティティセンター認証の使用

拡張 VPC ルーティングを有効にすると、Redshift はインターネットではなく仮想プライベートクラウド (VPC) を介してトラフィックをルーティングします。これには、Redshift が AWS IAM アイデンティティセンターのユーザーを認証するために行う呼び出しが含まれます。これは、ユーザーがインタラクティブにサインインするか、信頼できる ID 伝播を通じて取得した AWS IAM アイデンティティセンターのトークンを使用して接続するかにかかわらず適用されます。これらの接続を成功させるには、Redshift が VPC から AWS IAM アイデンティティセンターにアクセスできる必要があります。この接続を提供するには、インターフェイス VPC エンドポイントを使用することをお勧めします。これにより、プロビジョニングされたクラスターまたは Amazon Redshift Serverless ワークグループが AWS ネットワーク経由で AWS IAM アイデンティティセンターにアクセスできるようになります。

> **Important**
>
> これらの要件は、AWS IAM アイデンティティセンターを介して認証されるすべての接続に適用されます。IAM ユーザーとローカルデータベースユーザーは異なるメカニズムで認証されるため、影響を受けません。拡張 VPC ルーティングがオフになっている場合、これらのエンドポイントは必要ありません。

> **Note**
>
> 拡張 VPC ルーティングを変更すると、プロビジョニングされたクラスターが再起動されます。拡張 VPC ルーティングを有効にする方法の詳細については、「拡張 VPC ルーティングを有効にする」を参照してください。

インターフェイス VPC エンドポイント

クラスターまたはワークグループが存在する VPC に、次のインターフェイス VPC エンドポイント (AWS PrivateLink) を作成します。両方のエンドポイントが必要です。いずれかのエンドポイントが欠落しているか到達できない場合、AWS IAM アイデンティティセンターへのサインインは失敗します。

`com.amazonaws.region.sso-oauth`

ユーザーがサインイン時に提示する AWS IAM アイデンティティセンターのアクセストークンを検証し、それを Redshift セッションにスコープされたトークンと交換します。

`com.amazonaws.region.identitystore`

AWS IAM アイデンティティセンターの ID ストアから、ユーザーとその AWS IAM アイデンティティセンターグループメンバーシップを解決します。

インターフェイス VPC エンドポイントの作成方法の詳細については、「Amazon VPC ユーザーガイド」の「VPC エンドポイントの作成」を参照してください。

エンドポイントを作成する前に、クラスターまたはワークグループが実行される VPC で DNS hostnames 属性と DNS resolution 属性が有効になっていることを確認します。プライベート DNS 名は両方の属性に依存します。詳細については、「Amazon VPC ユーザーガイド」の「VPC の DNS 属性」を参照してください。

拡張 VPC ルーティングを使用するには、プロビジョニングされたクラスターまたは Redshift Serverless ワークグループにパブリックにアクセスできないようにする必要があります。

各エンドポイントを作成するときは、次の手順に従ってください。

- プライベート DNS 名を有効にする – Redshift は標準の DNS 名を使用してこれらのサービスに接続します。プライベート DNS 名がない場合、これらの名前は引き続きパブリックアドレスに解決され、呼び出しはエンドポイントにルーティングされません。

- AWS IAM アイデンティティセンターのリージョンにエンドポイントを作成する – AWS IAM アイデンティティセンターはリージョナルサービスです。これらのエンドポイントは、AWS IAM アイデンティティセンターのインスタンスが実行されているリージョンに作成します。クラスターまたはワークグループが別のリージョンで実行されていて、AWS IAM アイデンティティセンターのマルチリージョンレプリケーションを使用しない場合は、クロスリージョンエンドポイントオプションを使用して、エンドポイントから AWS IAM アイデンティティセンターのリージョンにアクセスできるようにします。
- 同じアベイラビリティーゾーン内のサブネットを選択する – クラスターまたはワークグループで使用するアベイラビリティーゾーン内のサブネットを選択します。
- インバウンド HTTPS を許可する – 各エンドポイントにアタッチされたセキュリティグループは、クラスターまたはワークグループのサブネットから TCP ポート 443 へのインバウンドトラフィックを許可する必要があります。
- デフォルトのエンドポイントポリシーを保持する – デフォルトのポリシーでは、フルアクセスが許可されます。組織で制限的なエンドポイントポリシーが必要な場合は、アクションのリストではなく、プリンシパルごとにスコープを設定します。次に、サインインが引き続き成功することを確認します。

AWS Network Firewall の使用

AWS Network Firewall はオプションです。インターフェイス VPC エンドポイントを使用する代わりに AWS Network Firewall でアウトバウンドトラフィックをフィルタリングする場合、ファイアウォールポリシーで AWS IAM アイデンティティセンターのサービスドメインを許可する必要があります。そうしないと、Redshift からサービスドメインにアクセスできず、サインインが失敗します。以下のドメインをファイアウォールの許可リストに追加し、*region* を AWS IAM アイデンティティセンターのリージョンに置き換えてください。

`oidc.`*region*`.amazonaws.com`

AWS IAM アイデンティティセンターのアクセストークンを検証し、それを Redshift セッションにスコープされたトークンと交換します。

`.sso.`*region*`.amazonaws.com`

AWS IAM アイデンティティセンターのサービスにアクセスし、AWS IAM アイデンティティセンターのアプリケーションとその割り当てを解決します。

`identitystore.`*region*`.amazonaws.com`

ユーザーとその AWS IAM アイデンティティセンターグループメンバーシップを解決します。

ファイアウォールがデフォルトで拒否するポリシーを使用している場合は、Amazon S3 や AWS Glue (データレイクにクエリを実行する場合) など、拡張 VPC ルーティングで Redshift が一般的なオペレーションに使用するドメインも許可する必要があります。詳細については、「Amazon Redshift の拡張 VPC ルーティング」を参照してください。

「インターフェイス VPC エンドポイント」で説明しているように、インターフェイス VPC エンドポイントを使用すると、このトラフィックは AWS ネットワーク内に留まるため、ファイアウォールの許可リストは不要になります。

AWS IAM アイデンティティセンター の Amazon Redshift ロールの自動作成

この機能は AWS IAM アイデンティティセンター との統合であり、グループメンバーシップに基づいて Redshift でロールを自動作成できます。

ロールの自動作成には、いくつかの利点があります。ロールを自動作成すると、Redshift は IdP にグループメンバーシップを持つロールを作成します。これにより、面倒な手動によるロールの作成とメンテナンスを行う必要がなくなります。また、Redshift ロールにマッピングされるグループを、包含パターンと除外パターンでフィルタリングすることもできます。

仕組み

IdP ユーザーとして Redshift にログインすると、次の一連のイベントが発生します。

1. Redshift は、IdP からグループメンバーシップを取得します。
2. Redshift は、ロールの形式 *idp_namespace*:*rolename* を使用して、これらのグループへのロールマッピングを自動作成します。
3. Redshift は、マッピングされたロールに対するアクセス許可を付与します。

各ユーザーがログインするたびに、カタログには存在しないが、そのユーザーが所属するグループは自動作成されます。オプションで、Redshift ロールが作成された IdP グループを制御するため、含有フィルターと除外フィルターを設定できます。

自動作成ロールの設定

CREATE IDENTITY PROVIDER および ALTER IDENTITY PROVIDER コマンドを使用して、ロールの自動作成を有効にして設定します。

```
-- Create a new IdP with auto role creation enabled
CREATE IDENTITY PROVIDER <idp_name> TYPE AWSIDC
```

```
  NAMESPACE '<namespace>'
  APPLICATION_ARN 'app_arn'
  IAM_ROLE 'role_arn'
  AUTO_CREATE_ROLES TRUE;

-- Enable on existing IdP
ALTER IDENTITY PROVIDER <idp_name>
  AUTO_CREATE_ROLES TRUE;

-- Disable
ALTER IDENTITY PROVIDER <idp_name>
  AUTO_CREATE_ROLES FALSE;
```

グループのフィルタリング

必要に応じて、INCLUDE パターンおよび EXCLUDE パターンを使用して、Redshift ロールにマッピングされる IdP グループをフィルタリングできます。パターンが競合した場合、EXCLUDE は INCLUDE よりも優先されます。

```
-- Only create roles for groups with 'dev'
CREATE IDENTITY PROVIDER <idp_name> TYPE AWSIDC
  ...
  AUTO_CREATE_ROLES TRUE
  INCLUDE GROUPS LIKE '%dev%';

-- Exclude 'test' groups
ALTER IDENTITY PROVIDER <idp_name>
  AUTO_CREATE_ROLES TRUE
  EXCLUDE GROUPS LIKE '%test%';
```

例

次の例は、フィルタリングなしでロールの自動作成を有効にする方法を示しています。

```
CREATE IDENTITY PROVIDER prod_idc TYPE AWSIDC  ...
  AUTO_CREATE_ROLES TRUE;
```

次の例では、開発グループは含まれ、テストグループは除外されます。

```
ALTER IDENTITY PROVIDER prod_idc
  AUTO_CREATE_ROLES TRUE
```

```
  INCLUDE GROUPS LIKE '%dev%'
  EXCLUDE GROUPS LIKE '%test%';
```

ベストプラクティス

ロールの自動作成を有効にする場合は、次のベストプラクティスを考慮してください。

- INCLUDE および EXCLUDE フィルターを使用して、ロールを取得するグループを制御します。
- 定期的にロールを監査し、未使用のロールをクリーンアップします。
- Redshift ロール階層を活用して、アクセス許可の管理を簡素化します。

Amazon Redshift と Amazon S3 Access Grants の統合

Amazon S3 Access Grants との統合を利用すると、IAM アイデンティティセンターのアイデンティティをシームレスに伝達して、Amazon S3 データへのアクセスを制御できます。この統合により、IAM アイデンティティセンターのユーザーとグループに基づいて Amazon S3 データへのアクセスを承認できます。

Amazon S3 Access Grants の詳細については、「S3 Access Grants でのアクセス管理」を参照してください。

Amazon S3 Access Grants を使用すると、アプリケーションに次の利点があります。

- IAM アイデンティティセンターのアイデンティティに基づいて、Amazon S3 のデータへアクセスをきめ細かく制御できます。
- Amazon Redshift と Amazon S3 全体で IAM アイデンティティセンターのアイデンティティを一元的に管理できます。
- Amazon S3 アクセスに対して個別の IAM アクセス許可を管理する手間を省けます。

仕組み

アプリケーションを Amazon S3 Access Grants と統合するには、次の操作を行います。

- まず、AWS マネジメントコンソール または AWS CLI を使用して、Amazon Redshift を Amazon S3 Access Grants と統合するように設定します。
- 次に、IdC 管理者権限を持つユーザーが、Amazon S3 Access Grants サービスを使用して、Amazon S3 バケットまたはプレフィックスに特定の IdC ユーザー/グループへのアクセスを許可します。詳細については、「S3 Access Grants での許可の使用」を参照してください。

- Redshift に対して認証済みの IdC ユーザーが S3 にアクセスするクエリ (COPY、UNLOAD、Spectrum オペレーションなど) を実行すると、Amazon Redshift はその IdC アイデンティティに範囲を限定された一時的な S3 アクセス認証情報を Amazon S3 Access Grants サービスから取得します。
- 次に、Amazon Redshift は取得した一時的な認証情報を使用して、そのクエリに対して承認された Amazon S3 の場所にアクセスします。

Amazon S3 Access Grants との統合を設定する

Amazon Redshift の Amazon S3 Access Grants との統合を設定するには、次の手順を実行します。

トピック

- AWS マネジメントコンソール を使用して Amazon S3 Access Grants との統合を設定する
- AWS CLI を使用して Amazon S3 Access Grants との統合を有効にする

AWS マネジメントコンソール を使用して Amazon S3 Access Grants との統合を設定する

1. Amazon Redshift コンソールを開きます。
2. [クラスター] ペインでクラスターを選択します。
3. クラスターの詳細ページの [ID プロバイダー統合] セクションで、S3 Access Grants サービスとの統合を有効にします。

> **Note**
>
> IAM アイデンティティセンターが設定されていない場合は、[ID プロバイダー統合] セクションは表示されません。詳細については、「Enabling AWS IAM アイデンティティセンター」を参照してください。

AWS CLI を使用して Amazon S3 Access Grants との統合を有効にする

1. S3 統合を有効にして新しい Amazon Redshift IdC アプリケーションを作成するには、次の手順を実行します。

```
aws redshift create-redshift-idc-application <other parameters>
  --service-integrations '[ {"S3AccessGrants": [{"ReadWriteAccess":
 {"Authorization": "Enabled"}}]} ]'
```

2. 既存のアプリケーションを変更して S3 Access Grants との統合を有効にするには、次の手順を実行します。

```
aws redshift modify-redshift-idc-application <other parameters>
  --service-integrations '[ {"S3AccessGrants": [{"ReadWriteAccess":
 {"Authorization": "Enabled"}}]} ]'
```

3. 既存のアプリケーションを変更して S3 Access Grants との統合を無効にするには、次の手順を実行します。

```
aws redshift modify-redshift-idc-application <other parameters>
  --service-integrations '[ {"S3AccessGrants": [{"ReadWriteAccess":
 {"Authorization": "Disabled"}}]} ]'
```

S3 Access Grants との統合を使用する

S3 Access Grants との統合を設定すると、S3 データにアクセスするクエリ (COPY、UNLOAD、Spectrum クエリなど) は IdC アイデンティティを承認に使用します。IdC を使用して認証されていないユーザーもこれらのクエリを実行できますが、これらのユーザーアカウントは、IdC が提供する一元管理を活用できません。

次の例は、S3 Access Grants 統合で実行されるクエリを示しています。

```
COPY table FROM 's3://mybucket/data';  // -- Redshift uses IdC identity
UNLOAD ('SELECT * FROM table') TO 's3://mybucket/unloaded/'    // -- Redshift uses IdC
 identity
```

AWS Lake Formation を介したデータのクエリ

AWS Lake Formation を使用すると、データレイクの一元管理と保護、データアクセスの提供が容易になります。AWS IAM アイデンティティセンターと Redshift を介して Lake Formation へのアイデンティティプロパゲーションを設定すると、管理者は組織の ID プロバイダー (IdP) グループに基づいて、Amazon S3 データレイクへのきめ細かいアクセスを許可できるようになります。このようなグループは AWS IAM アイデンティティセンターを通じて管理されます。このセクションでは、データレイクからのクエリとデータ共有からのクエリといういくつかのユースケースを設定する方法を示します。このようなユースケースでは、AWS IAM アイデンティティセンターと Redshift を活用して Lake Formation が管理するリソースに接続する方法が示されます。

データレイクのクエリのための AWS IAM アイデンティティセンターと Redshift 接続の使用

Redshift に接続された AWS IAM アイデンティティセンターを使用して、Lake Formation が管理するデータレイクにクエリを実行するユースケースを次の手順で説明します。

前提条件

この手順には、次のとおりの前提条件のステップがあります。

1. AWS IAM アイデンティティセンターを、Redshift による認証とアイデンティティ管理をサポートするように設定する必要があります。コンソールから AWS IAM アイデンティティセンターを有効にして、ID プロバイダー (IdP) ソースを選択できます。その後、IdP ユーザーのセットを AWS IAM アイデンティティセンターと同期します。また、このドキュメントで前述した手順に従って、AWS IAM アイデンティティセンターと Redshift 間の接続を設定する必要があります。
2. 新しい Amazon Redshift クラスターを作成して、設定ステップで AWS IAM アイデンティティセンターを介したアイデンティティ管理を有効にします。
3. Lake Formation 向けのマネージド AWS IAM アイデンティティセンターアプリケーションを作成して設定します。その後、AWS IAM アイデンティティセンターと Redshift 間の接続をセットアップします。手順は次のとおりです。
   a. AWS CLI で `modify-redshift-idc-application` コマンドを使用して、Lake Formation サービスと Redshift の AWS IAM アイデンティティセンターマネージドアプリケーションとの統合を有効にします。この呼び出しは、Lake Formation への認可を有効にする設定文字列値に設定された `service-integrations` パラメータを含みます。
   b. `create-lake-formation-identity-center-configuration` コマンドを使用して Lake Formation を設定します。これにより、Lake Formation 向けの AWS IAM アイデンティティセンターアプリケーションが作成され、AWS IAM アイデンティティセンターポータルに表示されます。管理者は `--cli-input-json` 引数を設定する必要があります。引数の値は、すべての AWS CLI API コールの標準形式を使用する JSON ファイルへのパスです。次の値を含める必要があります。
      - `CatalogId` – Lake Formation カタログ ID
      - `InstanceArn` – AWS IAM アイデンティティセンターインスタンスの ARN 値

管理者が前提条件の設定を完了すると、データベース管理者はデータレイクをクエリするための外部スキーマを作成できます。

1. 管理者が外部スキーマを作成する – Redshift データベース管理者は、次の SQL ステートメントを使用してデータベースに接続し、外部スキーマを作成します。

```
CREATE EXTERNAL SCHEMA if not exists my_external_schema from DATA CATALOG database
 'my_lf_integrated_db' catalog_id '12345678901234';
```

この場合、アクセスは AWS IAM アイデンティティセンターを介して管理されるため、IAM ロールを指定する必要はないことに注意します。

2. 管理者がアクセス許可を付与する – 管理者は AWS IAM アイデンティティセンターグループに使用権限を付与します。これにより、Redshift リソースへのアクセス許可が付与されます。これには、次のような SQL ステートメントを実行します。

```
GRANT USAGE ON SCHEMA "my_external_schema" to "MYCO:sales";
```

その後、管理者は AWS CLI を使用して、組織の要件に基づいてオブジェクトに対する Lake Formation のアクセス許可を付与します。

```
aws lakeformation grant-permissions ...
```

3. ユーザーがクエリを実行する – この説明の例では、この時点で営業グループに属する AWS IAM アイデンティティセンターユーザーは、クエリエディタ v2 を使用して Redshift データベースにログインできます。その後、次のサンプルのように、外部スキーマのテーブルにアクセスするクエリを実行できます。

```
SELECT * from my_external_schema.table1;
```

AWS IAM アイデンティティセンターと Redshift 接続を使用したデータ共有への接続

AWS IAM アイデンティティセンターを介してアクセスが管理されている場合、別の Redshift データウェアハウスからデータ共有にアクセスできます。これを行うには、クエリを実行して外部データベースを設定します。この手順を完了する前に、上記の手順の説明のとおり、Redshift と AWS IAM アイデンティティセンター間の接続を設定し、AWS Lake Formation アプリケーションを作成していることが前提となります。

1. 外部データベースの作成 — 管理者はデータ共有のための外部データベースを作成し、ARN を通じて参照します。次のサンプルは、この方法を説明しています。

```
CREATE DATABASE "redshift_external_db" FROM ARN 'arn:aws:glue:us-
east-1:123456789012:database/redshift_external_db-iad' WITH NO DATA CATALOG SCHEMA;
```

アイデンティティ管理のために AWS IAM アイデンティティセンターと Redshift を使用するこのようなユースケースでは、IAM ロールは含まれていません。

2. 管理者がアクセス許可を設定する – データベースを作成した後、管理者が AWS IAM アイデンティティセンターグループに使用権限を付与します。これにより、Redshift リソースに対するアクセス許可が付与されます。

```
GRANT USAGE ON DATABASE "my_external_db" to "MYCO:sales";
```

管理者は、AWS CLI を使用して、オブジェクトに対する Lake Formation のアクセス許可も付与します。

```
aws lakeformation grant-permissions ...
```

3. ユーザーがクエリを実行する – 営業グループのユーザーは、割り当てられたアクセス許可に基づいてデータベース内のテーブルに対してクエリを実行できます。

```
select * from redshift_external_db.public.employees;
```

データレイクに対するアクセス許可の付与とデータ共有アクセス許可の付与の詳細については、「ユーザーとグループへのアクセス許可の付与」を参照してください。スキーマまたはデータベースへの使用権限の詳細については、「GRANT」を参照してください。

信頼できるトークン発行者を使用したアプリケーションまたはツールと OAuth の統合

作成するクライアントツールに Redshift に接続するための機能を追加するには、AWS IAM アイデンティティセンターの接続を利用できます。Redshift と AWS IAM アイデンティティセンターの統合を設定済みである場合は、このセクションで詳述しているプロパティを使用して接続をセットアップします。

AWS IAM アイデンティティセンターを使用して Redshift に接続するための認証プラグイン

AWS IAM アイデンティティセンターを使用して、次のドライバープラグインを使用して Amazon Redshift に接続できます。

- `BrowserIdcAuthPlugin` – このプラグインは、AWS IAM アイデンティティセンターとのシームレスなシングルサインオン統合を容易にします。これにより、ユーザーが企業 ID プロバイダーで定義されたユーザー認証情報でサインインするためのブラウザウィンドウが作成されます。

- `IdpTokenAuthPlugin` – このプラグインは、Amazon Redshift ドライバーが AWS IAM アイデンティティセンター認証用のブラウザウィンドウを開くのではなく、認証フローを自分で管理したいアプリケーションで使用する必要があります。Okta、PingOne、Microsoft Entra ID (Azure AD) など、AWS IAM アイデンティティセンターに接続されているウェブ ID プロバイダーから、AWS IAM アイデンティティセンターが販売したアクセストークンまたは OpenID Connect (OIDC) JSON ウェブトークン (JWT) を受け入れます。クライアントアプリケーションは、この必須のアクセストークン/JWT を生成する必要があります。

`BrowserIdcAuthPlugin` による認証

Amazon Redshift ドライバーに応じて、次のプラグイン名を使用して `BrowserIdcAuthPlugin` を使用して接続します。

ドライバー	接続オプションキー	値	注意事項
JDBC	`plugin_name`	com.amazon.redshift.plugin.BrowserIdcAuthPlugin	接続時には、プラグインの完全修飾クラス名を入力する必要があります。
ODBC	`plugin_name`	BrowserIdcAuthPlugin	
Python	`credentials_provider`	BrowserIdcAuthPlugin	Python ドライバーに利用可能な `plugin_name` オプションはありません。代わりに `credentials_provider` を使用してください。

`BrowserIdcAuthPlugin` プラグインには、以下に示す追加の接続オプションがあります。

オプション名	必須?	説明	例
idc_region	必須	AWS IAM アイデンティティセンターイ	us-east-1

オプション名	必須?	説明	例
		ンスタンスが配置されている AWS リージョン リージョン。	
issuer_url	必須	AWS IAM アイデンティティセンターサーバーのインスタンスエンドポイント。この値は、AWS IAM アイデンティティセンターコンソールを使用して確認できます。	https://identitycenter.amazonaws.com/ssoins-g5j2k70sn4yc5nsc
listen_port	オプションです。	Amazon Redshift ドライバーがブラウザリダイレクトを介して AWS IAM アイデンティティセンターから `auth_code` レスポンスを受信するために使用するポート。	7890
idc_client_display_name	オプションです。	AWS IAM アイデンティティセンターのシングルサインオン同意ポップアップで、AWS IAM アイデンティティセンタークライアントがアプリケーションに使用する名前。	Amazon Redshift ドライバー

オプション名	必須?	説明	例
idp_response_timeout	オプションです。	Redshift ドライバーが、認証フローが完了するまでに待機する秒単位の時間。	60

上記の値は、ツールを作成して、このツールに接続する際の接続プロパティに入力します。詳細については、各ドライバーの接続オプションに関するドキュメントを参照してください。

- JDBC ドライバーバージョン 2.x の設定オプション
- ODBC ドライバーオプション
- Amazon Redshift Python コネクタの設定オプション

`IdpTokenAuthPlugin` による認証

Amazon Redshift ドライバーに応じて、次のプラグイン名を使用して `IdpTokenAuthPlugin` を使用して接続します。

ドライバー	接続オプションキー	値	注意事項
JDBC	`plugin_name`	com.amazon.redshift.plugin.IdpTokenAuthPlugin	接続時には、プラグインの完全修飾クラス名を入力する必要があります。
ODBC	`plugin_name`	IdpTokenAuthPlugin	
Python	`credentials_provider`	IdpTokenAuthPlugin	Python ドライバーに利用可能な `plugin_name` オプションはありません。代わりに `credentials_provider` を使用してください。

IdpTokenAuthPlugin プラグインには、以下に示す追加の接続オプションがあります。

オプション名	必須?	説明
トークン	必須	AWS IAM アイデンティティセンターが提供するアクセストークン、または AWS IAM アイデンティティセンターに接続されているウェブ ID プロバイダーが提供する OpenID Connect (OIDC) JSON ウェブトークン (JWT)。アプリケーションは、このトークンを生成するために、AWS IAM アイデンティティセンターまたは AWS IAM アイデンティティセンターに接続されている ID プロバイダーにより、アプリケーションユーザーを認証する必要があります。
token_type	必須	IdpTokenAuthPlugin に使用されるトークンのタイプ。利用できる値には以下のとおりです。 • ACCESS_TOKEN — AWS IAM アイデンティティセンターが提供するアクセストークンを使用する場合は、これを入力します。 • EXT_JWT — AWS IAM アイデンティティセンターに接続されたウェブベースの ID プロバイダーが提供する OpenID Connect (OIDC) JSON ウェブトー

オプション名	必須?	説明
		クン (JWT) を使用する場合は、これを指定します。

上記の値は、ツールを作成して、このツールに接続する際の接続プロパティに入力します。詳細については、各ドライバーの接続オプションに関するドキュメントを参照してください。

- JDBC ドライバーバージョン 2.x の設定オプション
- ODBC ドライバーオプション
- Amazon Redshift Python コネクタの設定オプション

Amazon Redshift クエリエディタ v2 からの接続のトラブルシューティング

このリストは、よく発生するエラーの詳細を示しており、AWS IAM アイデンティティセンター ID を使用してクエリエディタ v2 で Redshift データベースに接続する際に役立ちます。

- エラー: 接続の問題: アイデンティティセンターのセッション情報がありません。– このエラーが発生した場合は、ブラウザのセキュリティとプライバシーの設定を確認してください。これらのブラウザ設定 (特に Firefox の包括的 Cookie 保護機能などの安全な Cookie に関する設定) では、Amazon Redshift クエリエディタ v2 から Redshift データベースへの接続がブロックされる可能性があります。ブラウザで詳しく説明している修復ステップに従います。
  - Firefox – 現在、サードパーティの Cookie はデフォルトでブロックされます。ブラウザのアドレスバーで盾のアイコンをクリックし、クエリエディタ v2 での強化された追跡保護をオフに切り替えます。
  - Chrome のシークレットモード – Chrome のシークレットモードは、デフォルトでサードパーティの Cookie をブロックします。アドレスバーの目のアイコンをクリックして、クエリエディタ v2 でのサードパーティの Cookie を許可します。Cookie を許可するように設定を変更すると、アドレスバーに目のアイコンが表示されない場合があります。
  - Safari – Mac で Safari アプリを開きます。[設定]、[詳細] の順に選択します。オフに切り替える: すべての Cookie をブロックします。
  - Edge – [設定]、[Cookie とサイトのアクセス許可] の順に選択します。次に、[Cookie とサイトデータの管理と削除] を選択し、[サードパーティの Cookie をブロックする] をオフにします。

  設定の変更後に接続しようとして、「接続の問題: アイデンティティセンターのセッション情報がありません」というエラーメッセージが引き続き表示される場合は、AWS IAM アイデンティティ

センターとの接続を更新することをお勧めします。これを行うには、Redshift データベースインスタンスを右クリックし、[更新] を選択します。認証に使用できる新しいウィンドウが表示されます。

- エラー: 接続の問題: アイデンティティセンターのセッションが有効期限切れであるか、無効です。– Redshift でプロビジョニングされたクラスターまたはサーバーレスワークグループと AWS IAM アイデンティティセンターを統合した場合、ユーザーがクエリエディタ v2 から Redshift データベースに接続しようとすると、このエラーが表示される場合があります。これは、接続の試行が成功した後に起こる場合があります。この場合は、再認証することをお勧めします。これを行うには、Redshift データベースインスタンスを右クリックし、[更新] を選択します。認証に使用できる新しいウィンドウが表示されます。
- エラー: 無効なスコープ。ユーザー認証情報には Redshift に接続する権限がありません。Redshift でプロビジョニングされたクラスターまたはサーバーレスワークグループと AWS IAM アイデンティティセンターを統合した場合、ユーザーがクエリエディタ v2 から Redshift データベースに接続しようとすると、このエラーが表示される場合があります。この場合、クエリエディタ v2 を使用して AWS IAM アイデンティティセンター経由でユーザーを正常に接続して認証し、正しいリソースにアクセスできるようにするには、管理者が Redshift コンソールで Redshift AWS IAM アイデンティティセンターアプリケーションにユーザーを割り当てる必要があります この操作は [IAM アイデンティティセンターの接続] で実行します。これに続いて、ユーザーは 1 時間後に正常に接続を確立できます。この時間は、AWS IAM アイデンティティセンターセッションのキャッシュの上限です。
- エラー: データベースを一覧表示できませんでした。致命的: クラスターが自動的に一時停止すると、クエリが失敗します。– Amazon Redshift Serverless データベースがアイドル状態で、ワークロードを処理していない場合、AWS IAM アイデンティティセンター ID に接続すると、データベースは一時停止したままになることがあります。この問題を修正するには、別の認証方法で口グインして Serverless ワークグループを再開します。次に、AWS IAM アイデンティティセンター ID を使用してデータベースに接続します。
- エラー: AWS IAM アイデンティティセンターとのフェデレーションの試行中にエラーが発生しました。Amazon Redshift 管理者は、Redshift コンソールを使用して AWS IAM アイデンティティセンター QEV2 アプリケーションを削除して再作成する必要があります。– このエラーは通常、クエリエディタ v2 に関連付けられた AWS IAM アイデンティティセンターのアプリケーションインスタンスを削除したときに発生します。この問題を修正するには、Amazon Redshift 管理者が AWS IAM アイデンティティセンターで Redshift およびクエリエディタ v2 アプリケーションを削除して再作成する必要があります。これを行うには、Redshift コンソールまたは https://docs.aws.amazon.com/cli/latest/reference/redshift/delete-redshift-idc-application.html CLI コマンドを使用できます。

Amazon Redshift でのサービスにリンクされたロールの使用

Amazon Redshift は、AWS Identity and Access Management (IAM) サービスにリンクされたロールを使用しています。サービスにリンクされたロールは、Amazon Redshift に直接リンクされた一意のタイプの IAM ロールです。サービスにリンクされたロールは、Amazon Redshift によって事前定義されており、サービスがお客様の Amazon Redshift クラスターの代わりに AWS のサービスを呼び出すのに必要なアクセス許可がすべて含まれています。

サービスにリンクされたロールを使用することで、必要なアクセス許可を手動で追加する必要がなくなるため、Amazon Redshift の設定が簡単になります。このロールは、Amazon Redshift ユースケースにリンクされており、事前定義されたアクセス許可があります。Amazon Redshift のみがロールを引き受けることができ、サービスにリンクされたロールのみが事前定義されたアクセス権限のポリシーを使用できます。Amazon Redshift は、クラスターまたは Redshift マネージド VPC エンドポイントの初回作成時に、サービスリンクロールをアカウントに作成します。このサービスリンクロールは、アカウントで Amazon Redshift クラスターや Redshit マネージド VPC エンドポイントをすべて削除した後でのみ削除できます。これにより、Amazon Redshift リソースへの必要なアクセス許可を不注意に削除することがなくなり、リソースが保護されます。

Amazon Redshift では、これが利用可能なすべてのリージョンにおいて、サービスにリンクされたロールの使用がサポートされています。詳細については、 AWS リージョンとエンドポイントを参照してください。

サービスリンクロールをサポートする他のサービスについては、「IAM と連携する AWS サービス」ページで、[サービスリンクロール] 列が [はい] になっているサービスを参照してください。サービスにリンクされたロールに関するドキュメントをサービスで表示するには、 はい リンクを選択します。

Amazon Redshift へのサービスにリンクされたロールのアクセス許可

Amazon Redshift が AWSServiceRoleForRedshift という名前のサービスにリンクされたロールを使用 – Amazon Redshift がユーザーに代わって AWS のサービスを呼び出すことを許可します。このサービスリンクロールは、マネージドポリシー `AmazonRedshiftServiceLinkedRolePolicy` にアタッチされます。このポリシーの更新情報については、「Amazon Redshift の AWS 管理 (事前定義) ポリシー」を参照してください。

AWSServiceRoleForRedshift サービスにリンクされたロールは、 **`redshift.amazonaws.com`** のみを信頼し、ロールを引き受けます。

AWSServiceRoleForRedshift サービスにリンクされたロールのアクセス権限ポリシーでは、Amazon Redshift はすべての関連リソースで以下のことを実行することができます。

- ec2:DescribeVpcs
- ec2:DescribeSubnets
- ec2:DescribeNetworkInterfaces
- ec2:DescribeAddress
- ec2:AssociateAddress
- ec2:DisassociateAddress
- ec2:CreateNetworkInterface
- ec2:DeleteNetworkInterface
- ec2:ModifyNetworkInterfaceAttribute
- ec2:CreateVpcEndpoint
- ec2:DeleteVpcEndpoints
- ec2:DescribeVpcEndpoints
- ec2:ModifyVpcEndpoint
- ec2:DescribeVpcAttribute
- ec2:DescribeSecurityGroups
- ec2:DescribeInternetGateways
- ec2:DescribeSecurityGroupRules
- ec2:DescribeAvailabilityZones
- ec2:DescribeNetworkAcls
- ec2:DescribeRouteTables
- ec2:AssignIpv6Addresses
- ec2:UnassignIpv6Addresses

ネットワークリソースに対するアクセス許可

次のアクセス許可では、セキュリティグループルールを作成して管理するためのアクションを Amazon EC2 で実行できます。これらのセキュリティグループおよびルールは、Amazon Redshift aws:RequestTag/Redshift のリソースタグに厳密に関連付けられています。これにより、アクセス許可の範囲は特定の Amazon Redshift のリソースに制限されます。

- ec2:CreateSecurityGroup
- ec2:AuthorizeSecurityGroupEgress

- ec2:AuthorizeSecurityGroupIngress
- ec2:RevokeSecurityGroupEgress
- ec2:RevokeSecurityGroupIngress
- ec2:ModifySecurityGroupRules
- ec2:DeleteSecurityGroup

Service Quotas のアクセス許可

次のアクセス許可により、呼び出し元は Service Quotas を取得できます。

servicequotas:GetServiceQuota

次の JSON フラグメントは、Service Quotas のアクションとリソーススコープを示しています。

```
{
   "Sid": "ServiceQuotasToCheckCustomerLimits",
   "Effect": "Allow",
   "Action": [
      "servicequotas:GetServiceQuota"
   ],
   "Resource": [
      "arn:aws:servicequotas:*:*:ec2/L-0263D0A3",
      "arn:aws:servicequotas:*:*:vpc/L-29B6F2EB"
   ]
}
```

クォータコードは次のとおりです。

- L-0263D0A3 – EC2-VPC Elastic IP のクォータコード。
- L-29B6F2EB – VPC あたりのインターフェイス VPC エンドポイントのクォータコード。

詳細については、「AWS サービスクォータ」を参照してください。

監査ログ記録のためのアクション

logs プレフィックスとともにリストされたアクションは、監査ログ記録と関連機能と関係があります。具体的には、ロググループとログストリームの作成と管理に関連します。

- logs:CreateLogGroup

- logs:PutRetentionPolicy
- logs:CreateLogStream
- logs:PutLogEvents
- logs:DescribeLogStreams
- logs:GetLogEvents

次の JSON は、監査ログ記録用の Amazon Redshift に対する、アクションとリソースの範囲を示しています。

```
[
    {
        "Sid": "EnableCreationAndManagementOfRedshiftCloudwatchLogGroups",
        "Effect": "Allow",
        "Action": [
            "logs:CreateLogGroup",
            "logs:PutRetentionPolicy"
        ],
        "Resource": [
            "arn:aws:logs:*:*:log-group:/aws/redshift/*"
        ]
    },
    {
        "Sid": "EnableCreationAndManagementOfRedshiftCloudwatchLogStreams",
        "Effect": "Allow",
        "Action": [
            "logs:CreateLogStream",
            "logs:PutLogEvents",
            "logs:DescribeLogStreams",
            "logs:GetLogEvents"
        ],
        "Resource": [
            "arn:aws:logs:*:*:log-group:/aws/redshift/*:log-stream:*"
        ]
    }
]
```

サービスリンクロールと、AWS での用途については、「サービスにリンクされたロールの使用」を参照してください。Amazon Redshift での特定のアクション、およびその他の IAM リソースの詳細については、「Amazon Redshift のアクション、リソース、および条件キー」を参照してください。

を使用して管理者認証情報を管理するためのアクションAWS Secrets Manager

secretsmanager プレフィックス付きでリストされているアクションは、Amazon Redshift を使用して管理者認証情報を管理するためのものです。このようなアクションを実行すると、Amazon Redshift は AWS Secrets Manager を使用して、管理者認証情報シークレットの作成と管理を行います。

次の JSON は、AWS Secrets Manager で管理者認証情報を管理するための Amazon Redshift に対するアクションとリソースの範囲を示しています。

```
[
    {
        "Effect": "Allow",
        "Action": [
            "secretsmanager:DescribeSecret",
            "secretsmanager:DeleteSecret",
            "secretsmanager:PutSecretValue",
            "secretsmanager:UpdateSecret",
            "secretsmanager:UpdateSecretVersionStage",
            "secretsmanager:RotateSecret"
        ],
        "Resource": [
            "arn:aws:secretsmanager:*:*:secret:redshift!*"
        ],
        "Condition": {
            "StringEquals": {
                "secretsmanager:ResourceTag/aws:secretsmanager:owningService":
 "redshift"
            }
        }
    },
    {
        "Effect": "Allow",
        "Action": [
            "secretsmanager:GetRandomPassword"
        ],
        "Resource": "*"
    }
]
```

クラスターとサーバーレス名前空間を AWS Glue Data Catalog に登録するためのアクション

glue というプレフィックスが付いたアクションは、プロビジョニング済みクラスターまたはサーバーレス名前空間の登録を通じて AWS Glue Data Catalog に作成されたカタログへのアクセスに関連しています。詳細については、「Amazon Redshift Database デベロッパーガイド」の「Apache Iceberg の Amazon Redshift との互換性」を参照してください。

次の JSON は、Amazon Redshift に対する、AWS Glue Data Catalog のカタログへのアクセスに関するアクションとリソースの範囲を示しています。

```
[
    {
        "Sid": "DiscoverRedshiftCatalogs",
        "Effect": "Allow",
        "Action": [
            "glue:GetCatalogs",
            "glue:GetCatalog"
        ],
        "Resource": [
            "arn:aws:glue:*:*:catalog",
            "arn:aws:glue:*:*:catalog/*"
        ],
   "Condition":
    {
        "Bool":
        {
            "glue:EnabledForRedshiftAutoDiscovery": "true"
        },
        "StringEquals": {
             "aws:ResourceAccount": "${aws:PrincipalAccount}"
        }
    }
 },
 {
    "Sid": "LakeFormationGetMetadataAccessForFederatedCatalogs",
    "Effect": "Allow",
    "Action": [
        "lakeformation:GetDataAccess"
    ],
    "Resource": [ "*" ],
    "Condition":
    {
        "Bool":
        {
            "lakeformation:EnabledOnlyForMetaDataAccess":"true"
```

```
        },
        "StringEquals": {
             "aws:ResourceAccount": "${aws:PrincipalAccount}"
        },
        "ForAnyValue:StringEquals":
        {
            "aws:CalledVia": "glue.amazonaws.com"
        }
    }
 }
    }
]
```

glue:GetCatalog および glue:GetCatalogs アクセス許可には条件 glue:EnabledForRedshiftAutoDiscovery:true があり、Amazon Redshift はカタログを自動的に検出するための IAM アクセスを付与します。オプトアウトするには、AWS Glue アカウントレベルのリソースポリシーを追加して、サービスにリンクされたロールのカタログへのアクセスを選択的に拒否します。サービスリンクロールにはポリシーに明示的な許可アクションがすでにあるため、オプトアウトポリシーはそのアクションを明示的に拒否する必要があります。追加のポリシーが Amazon Redshift の自動検出を拒否する次の例を考えてみましょう。

JSON

```
{
  "Version":"2012-10-17",
  "Statement" : {
            "Effect": "Deny",
            "Action": [
                "glue:GetCatalog",
                "glue:GetCatalogs"
            ],
            "Principal" : {
            "AWS" : "arn:aws:iam::111122223333:role/aws-service-role/
redshift.amazonaws.com/AWSServiceRoleForRedshift"
            },
            "Resource": [
                "arn:aws:glue:*:*:catalog/<s3_table_catalog_name>",
                "arn:aws:glue:*:*:catalog/<s3_table_catalog_name>/*"
            ]
        }
}
```

IAM エンティティが AWSServiceRoleForRedshift サービスにリンクされたロールを作成するには

```
{
    "Effect": "Allow",
    "Action": [
        "iam:CreateServiceLinkedRole"
    ],
    "Resource": "arn:aws:iam::<AWS-account-ID>:role/aws-service-role/
redshift.amazonaws.com/AWSServiceRoleForRedshift",
    "Condition": {"StringLike": {"iam:AWSServiceName": "redshift.amazonaws.com"}}
}
```

IAM エンティティが AWSServiceRoleForRedshift サービスにリンクされたロールを削除するには

以下のポリシーステートメントを IAM エンティティのアクセス許可に追加します。

```
{
    "Effect": "Allow",
    "Action": [
        "iam:DeleteServiceLinkedRole",
        "iam:GetServiceLinkedRoleDeletionStatus"
    ],
    "Resource": "arn:aws:iam::<AWS-account-ID>:role/aws-service-role/
redshift.amazonaws.com/AWSServiceRoleForRedshift",
    "Condition": {"StringLike": {"iam:AWSServiceName": "redshift.amazonaws.com"}}
}
```

または AWS 管理ポリシーを使用して、Amazon Redshift への フルアクセスを許可 することもできます。

Amazon Redshift のサービスにリンクされたロールの作成

AWSServiceRoleForRedshift サービスにリンクされたロールを手動で作成する必要はありません。Amazon Redshift は、お客様のサービスにリンクされたロールを作成します。AWSServiceRoleForRedshift サービスにリンクされたロールがアカウントより削除されている場合は、新しい Amazon Redshift クラスターを起動すると、Amazon Redshift によってロールが作成されます。

> **Important**
>
> 2017 年 9 月 18 日より前に Amazon Redshift サービスを使用していた場合、サービスにリンクされたロールのサポートが開始された時点で、Amazon Redshift により

AWSServiceRoleForRedshift ロールがアカウントに作成済されています。詳細については[IAM アカウントに新しいロールが表示される]を参照してください。

Amazon Redshift のサービスにリンクされたロールの編集

Amazon Redshift では、サービスにリンクされたロールである AWSServiceRoleForRedshift を編集できません。サービスにリンクされたロールを作成すると、多くのエンティティによってロールが参照される可能性があるため、ロール名を変更することはできません。ただし、ロールの説明は IAM コンソール、AWS Command Line Interface (AWS CLI)、または IAM API を使用して編集することができます。詳細については、 IAM ユーザーガイドの ロールの変更 を参照してください。

Amazon Redshift のサービスにリンクされたロールの削除

サービスにリンクされたロールが必要な機能またはサービスが不要になった場合には、そのロールを削除することをお勧めします。そうすることで、使用していないエンティティがアクティブにモニタリングされたり、メンテナンスされたりすることがなくなります。

アカウントのサービスにリンクされたロールを削除するには、アカウントのクラスターをシャットダウンした上で削除する必要があります。詳細については、「クラスターのシャットダウンと削除」を参照してください。

サービスにリンクされたロールは、IAM コンソール、AWS CLI、 IAM API を使用して削除することができます。詳細については、「IAM ユーザーガイド」の「サービスにリンクされたロールの削除」を参照してください。

IAM 認証を使用したデータベースユーザー認証情報の生成

AWS Identity and Access Management (IAM) のアクセス許可ポリシーによって付与されるアクセス許可に基づいて一時的なデータベース認証情報を生成し、Amazon Redshift データベースに対するユーザーのアクセスを管理できます。

一般的に、Amazon Redshift データベースユーザーは、データベースのユーザー名とパスワードを提供してデータベースにログオンします。ただし、Amazon Redshift データベースでユーザー名とパスワードを維持する必要はありません。ユーザーがユーザー認証情報を作成し、IAM 認証情報に基づいてデータベースにログオンできるようにシステムを設定できます。

Amazon Redshift は、一時的データベースユーザー認証情報を生成する GetClusterCredentials API オペレーションを提供します。`GetClusterCredentials` オペレーションを呼び出すプロセスを管理する Amazon Redshift JDBC、または ODBC ドライバーで SQL クライアントを設定できます。

これは、データベースのユーザー認証情報を取得し、SQL クライアントと Amazon Redshift データベース間の接続を確立することにより行われます。また、データベースアプリケーションを使用してプログラムで GetClusterCredentials オペレーションを呼び出し、データベースのユーザー認証情報を取得して、データベースに接続することもできます。

既に AWSの外部でユーザー ID を管理している場合、Security Assertion Markup Language (SAML) 2.0 に準拠した ID プロバイダー (IdP) を使用して、Amazon Redshift リソースへのアクセスを管理できます。IAM ロールへのフェデレーティッドユーザーアクセスを許可するよう IdP を設定します。この IAM ロールにより、一時的データベース認証情報を生成し、Amazon Redshift データベースにログオンできます。

SQL クライアントには、自動的に GetClusterCredentials オペレーションを呼び出すためのアクセス許可が必要です。IAM ロールを作成し、GetClusterCredentials オペレーションと関連アクションへのアクセスを付与または制限する IAM アクセス権限ポリシーを添付して、それらのアクセス許可を管理します。ベストプラクティスとして、アクセス許可ポリシーを IAM ロールにアタッチし、それを必要に応じてユーザーやグループに割り当てることをお勧めします。詳細については、「Amazon Redshift での Identity and Access Management」を参照してください。

このポリシーでは、Amazon Redshift クラスター、データベース、データベースユーザー名、ユーザーグループ名など特定のリソースへのアクセスが付与または制限されます。

> **Note**
>
> Amazon Redshift JDBC、または ODBC ドライバーを使用して GetClusterCredentials オペレーションを呼び出し、データベースにログオンするプロセスを管理することをお勧めします。分かりやすいように、このトピック全体で JDBC または ODBC ドライバーとともに SQL クライアントを使用するものと仮定します。
> GetClusterCredentials オペレーションまたは並列 get-cluster-credentials CLI コマンドの使用の具体的な詳細と例については、GetClusterCredentials および get-cluster-credentials を参照してください。

認証と認可を一元的に管理するため、Amazon Redshift では IAM を使用したデータベース認証がサポートされ、エンタープライズフェデレーションによるユーザー認証が可能になります。ユーザーを作成する代わりに、AWS Directory Service、エンタープライズユーザーディレクトリ、またはウェブ ID プロバイダーの既存のアイデンティティを使用できます。このようなユーザーはフェデレーティッドユーザーと呼ばれます。 AWS では、IdP を通じてアクセスが要求されたとき、フェデレーティッドユーザーにロールを割り当てます。

組織内のユーザーまたはクライアントアプリケーションに Amazon Redshift API オペレーションを呼び出すフェデレーティッドアクセスを提供するには、SAML 2.0 をサポートした JDBC または ODBC ドライバーを使用して、組織の IdP から認証をリクエストすることもできます。この場合、組織のユーザーは Amazon Redshift に直接アクセスすることはできません。

詳細については、「IAM ユーザーガイド」の「ID プロバイダーとフェデレーション」を参照してください。

一時的な IAM 認証情報の作成

このセクションでは、IAM ベースの一時的データベースユーザー認証情報を生成し、新しい認証情報を使用してデータベースにログオンするようシステムを設定する方法を示しています。

プロセスのおおまかな流れは次のとおりです。

1. ステップ 1: IAM シングルサインオンアクセス用の IAM ロールを作成する

   (任意) IAM 認証と、サードパーティー認証プロバイダ (IdP) を統合して、Amazon Redshift データベースへのアクセスをユーザーに認証できます。
2. ステップ 2: IdP の SAML アサーションを設定する

   (任意) IdP を使って IAM 認証を使用するには、組織のユーザーまたはグループを IAM ロールにマッピングするクレームルールを IdP アプリケーションで定義する必要があります。オプションで、属性要素を含めて `GetClusterCredentials` パラメータを設定できます。
3. ステップ 3: GetClusterCredentialsWithIAM または GetClusterCredentials を呼び出すアクセス許可を持つ IAM ロールを作成する

   SQL クライアントアプリケーションは、`GetClusterCredentials` オペレーションを呼び出すときにユーザーを引き受けます。ID プロバイダアクセス用の IAM ロールを作成したら、そのロールに必要なアクセス権限を追加できます。
4. ステップ 4: データベースユーザーとデータベースグループを作成する

   (任意) デフォルトでは、ユーザー名が存在しない場合、`GetClusterCredentials` は認証情報を新しいユーザーを作成します。ログオン時にユーザーが参加するユーザーグループを指定することもできます。デフォルトでは、データベースユーザーは PUBLIC グループに参加します。
5. ステップ 5: IAM 認証情報を使用するように JDBC または ODBC 接続を設定する

   Amazon Redshift データベースに接続するには、Amazon Redshift JDBC または ODBC ドライバーを使用するよう SQL クライアントを設定します。

ステップ 1: IAM シングルサインオンアクセス用の IAM ロールを作成する

シングルサインオンアクセスに ID プロバイダを利用しない場合は、このステップを省略できます。

既に AWSの外部でユーザー ID を管理している場合、IAM 認証とサードパーティーの SAML-2.0 ID プロバイダー (IdP) を統合して、Amazon Redshift データベースへのユーザーのアクセスを認証できます。

詳細については、「IAM ユーザーガイド」の「ID プロバイダーとフェデレーション」を参照してください。

Amazon Redshift IdP 認証を使用する前に、AWS SAML ID プロバイダーを作成します。IAM コンソールで IdP を作成し、IdP およびその設定について AWS に通知できます。これにより、AWS アカウントと IdP 間の信頼が確立されます。ロールを作成する手順については、IAM ユーザーガイドから SAML 2.0 フェデレーション用のロール作成 (コンソール) を参照してください。

ステップ 2: IdP の SAML アサーションを設定する

IAM ロールを作成したら、組織のユーザーまたはグループを IAM ロールにマッピングするクレームルールを IdP アプリケーションで定義する必要があります。詳細については、IAM ユーザーガイドの 認証応答の SAML アサーションを設定する を参照してください。

オプションの `GetClusterCredentials` パラメータ `DbUser`、`AutoCreate`、`DbGroups`を使用する場合は、2 つのオプションがあります。JDBC 接続または ODBC 接続でパラメータの値を設定することも、SAML 属性要素を IdP に追加して値を設定することもできます。`DbUser`、`AutoCreate`、`DbGroups` パラメータの詳細については、ステップ 5: IAM 認証情報を使用するように JDBC または ODBC 接続を設定するを参照してください。

> **Note**
>
> 「`${redshift:DbUser}`」で説明されているように、IAM ポリシー変数 GetClusterCredentials のリソースポリシー を使用する場合、`DbUser`の値は API オペレーションのリクエストコンテキストから取得した値で置き換えられます。Amazon Redshift ドライバーは、SAML 属性として提供される値ではなく、接続 URL から提供される `DbUser` 変数の値を使用します。
> この接続の安全性を維持するには、IAM ポリシーの条件を使用して、`DbUser` 値を `RoleSessionName`で検証することが推奨されます。IAM ポリシーを使用して条件を設定する方法の例は、例 8: GetClusterCredentials を使用する IAM ポリシーで参照できます。

DbUser、AutoCreate、DbGroups パラメータを設定するように IdP を設定するには、次の Attribute 要素を含めます。

- Attribute 属性を持つ Name 要素は「https://redshift.amazon.com/SAML/Attributes/DbUser」に設定します。

  AttributeValue 要素は、Amazon Redshift データベースに接続するユーザー名に設定します。

  AttributeValue 要素の値は小文字のみで、先頭は文字にし、英数字、アンダースコア (_)、プラス記号 (+)、ドット (.)、アットマーク (@)、ハイフン (_) のみを含み、128 文字未満である必要があります。通常、ユーザー名はユーザー ID (例: bobsmith) またはメールアドレス (例: bobsmith@example.com) になります。値にスペースを含めることはできません (ユーザーの表示名を Bob Smith にするなど)。

```
<Attribute Name="https://redshift.amazon.com/SAML/Attributes/DbUser">
    <AttributeValue>user-name</AttributeValue>
</Attribute>
```

- Name 属性を持つ Attribute 要素は「https://redshift.amazon.com/SAML/Attributes/AutoCreate」に設定します。

  存在していない場合に、新しいデータベースユーザーを作成するには、AttributeValue 要素を true に設定します。データベースユーザーが Amazon Redshift データベースに存在している必要があることを指定するには、AttributeValue を false に設定します。

```
<Attribute Name="https://redshift.amazon.com/SAML/Attributes/AutoCreate">
    <AttributeValue>true</AttributeValue>
</Attribute>
```

- Attribute 属性を持つ Name 要素は「https://redshift.amazon.com/SAML/Attributes/DbGroups」に設定します。

  この要素には、1 つ以上の AttributeValue 要素が含まれています。Amazon Redshift データベースに接続するとき、セッションの期間中に DbUser が参加するデータベースグループ名に各 AttributeValue 要素を設定します。

```
<Attribute Name="https://redshift.amazon.com/SAML/Attributes/DbGroups">
    <AttributeValue>group1</AttributeValue>
    <AttributeValue>group2</AttributeValue>
    <AttributeValue>group3</AttributeValue>
```

```
</Attribute>
```

ステップ 3: GetClusterCredentialsWithIAM または GetClusterCredentials を呼び出すアクセス許可を持つ IAM ロールを作成する

SQL クライアントには、自動的に GetClusterCredentialsWithIAM または GetClusterCredentials オペレーションを呼び出すための許可が必要です。その許可を付与するには、ユーザーまたはロールを作成し、必要なアクセス許可を付与するポリシーをアタッチします。どちらのオペレーションもクラスター認証情報の取得に使用できますが、認証方法は異なります。GetClusterCredentialsWithIAM は IAM ロールを使用して、ロールにマッピングするデータベースユーザーを自動的に作成します。これは、IAM ロールレベルでのアクセス許可の管理に有益である一方、GetClusterCredentials はデータベース内の特定のユーザー名の認証情報を提供します。

GetClusterCredentialsWithIAM を呼び出すアクセス許可を持つ IAM ロールを作成するには

1. IAM サービスを使用して、ユーザーまたはロールを作成します。既存のユーザーまたはロールを使用することもできます。たとえば、アイデンティティプロバイダアクセス用の IAM ロールを作成したら、そのロールに必要な IAM ポリシーをアタッチできます。
2. redshift:GetClusterCredentialsWithIAM オペレーションを呼び出すアクセス許可を持つアクセス許可ポリシーをアタッチします。次のポリシーサンプルは、特定のクラスターとデータベース、クラスター内の任意のデータベース、任意のクラスター内の任意のデータベースに対して オペレーションを許可するオプションを示しています。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "SpecificClusterAndDBName",
            "Effect": "Allow",
            "Action": "redshift:GetClusterCredentialsWithIAM",
            "Resource": [
                "arn:aws:redshift:us-east-1:123456789012:dbname:testcluster/
testdatabase"
            ]
        },
        {
            "Sid": "SpecificClusterAndAnyDBName",
            "Effect": "Allow",
```

```
            "Action": "redshift:GetClusterCredentialsWithIAM",
            "Resource": "arn:aws:redshift:us-
east-1:123456789012:dbname:examplecluster/*",
        },
        {
            "Sid": "AnyClusterAnyDatabase",
            "Effect": "Allow",
            "Action": "redshift:GetClusterCredentialsWithIAM",
            "Resource": "*"
        }
    ]
}
```

GetClusterCredentials を呼び出すアクセス許可を持つ IAM ロールを作成するには

1. IAM サービスを使用して、ユーザーまたはロールを作成します。既存のユーザーまたはロールを使用することもできます。たとえば、アイデンティティプロバイダアクセス用の IAM ロールを作成したら、そのロールに必要な IAM ポリシーをアタッチできます。
2. `redshift:GetClusterCredentials` オペレーションを呼び出すアクセス許可を持つアクセス許可ポリシーをアタッチします。指定するオプションパラメータにより、追加のアクションやリソースをポリシーで許可または制限することもできます。

   - SQL クライアントによるクラスター ID、AWS リージョン、およびポートの取得を許可するには、`redshift:DescribeClusters` オペレーションを呼び出すアクセス許可を Redshift クラスターリソースに含めます。
   - `AutoCreate` オプションを使用する場合、`redshift:CreateClusterUser` を呼び出すアクセス許可を dbuser リソースに含めます。次の Amazon リソースネーム (ARN) は、Amazon Redshift dbuserを指定します。*region*、*account-id*、*cluster-name* を、お使いの AWS リージョン、アカウント、クラスターの値に置き換えてください。*dbuser-name*に、クラスターデータベースへのログインに使用するユーザー名を指定します。

     ```
     arn:aws:redshift:region:account-id:dbuser:cluster-name/dbuser-name
     ```

   - (任意) 次の形式で Amazon Redshift dbname リソースを指定する ARN を追加します。*region*、*account-id*、*cluster-name* を、お使いの AWS リージョン、アカウント、クラスターの値に置き換えてください。*database-name* に、ユーザーがログインするデータベースの名前を指定します。

```
arn:aws:redshift:region:account-id:dbname:cluster-name/database-name
```

- DbGroups オプションを使用する場合、次の形式で、redshift:JoinGroup オペレーションを呼び出すアクセス許可を Amazon Redshift dbgroup リソースに含めます。*region*、*account-id*、*cluster-name* を、お使いの AWS リージョン、アカウント、クラスターの値に置き換えてください。*dbgroup-name* には、ユーザーがログイン時に参加するユーザーグループの名前を指定します。

```
arn:aws:redshift:region:account-id:dbgroup:cluster-name/dbgroup-name
```

詳細な説明と例については、GetClusterCredentials のリソースポリシーを参照してください。

ステップ 4: データベースユーザーとデータベースグループを作成する

(任意) クラスターデータベースにログインするために使用するデータベースユーザーを作成できます。既存のユーザー用の一時的ユーザー認証情報を作成する場合、ユーザーのパスワードを無効にして、一時パスワードでログオンするようユーザーに強制できます。または、GetClusterCredentials Autocreate 自動作成オプションを使用して、自動的に新しいデータベースユーザーを作成できます。

アクセス許可により、IAM データベースユーザーがログイン時に参加するデータベースユーザーグループを作成できます。GetClusterCredentials オペレーションを呼び出す場合、新しいユーザーがログイン時に参加するユーザーグループ名のリストを指定できます。これらのグループメンバーシップは、特定のリクエストで生成された認証情報を使用して作成されたセッションに対してのみ有効です。

データベースユーザーとデータベースグループを作成するには

1. Amazon Redshift データベースにログインし、CREATE USER を使用してデータベースユーザーを作成するか、ALTER USERを使用して既存のユーザーを変更します。
2. (任意) ユーザーがパスワードを使用することを防ぐために PASSWORD DISABLE オプションを指定します。ユーザーのパスワードが無効になると、ユーザーは一時的な認証情報を使用してのみログオンできます。パスワードが無効になっていない場合、ユーザーはパスワードまたは一時的な認証情報を使用してログオンできます。スーパーユーザーのパスワードを無効にすることはできません。

AWS マネジメントコンソール の外部で AWS を操作するには、プログラマチックアクセス権が必要です。プログラムによるアクセス権を付与する方法は、AWS にアクセスしているユーザーのタイプによって異なります。

ユーザーにプログラムによるアクセス権を付与するには、以下のいずれかのオプションを選択します。

プログラムによるアクセス権を必要とするユーザー	目的	方法
IAM	(推奨) 一時的な認証情報としてコンソール認証情報を使用して、AWS CLI、AWS SDK、または AWS API へのプログラマチックリクエストに署名します。	使用するインターフェイスの指示に従ってください。 • AWS CLI については、「AWS Command Line Interface ユーザーガイド」の「AWS ローカル開発用のログイン」を参照してください。 • AWS SDK については、「AWS SDK およびツールリファレンスガイド」の「AWS ローカル開発用のログイン」を参照してください。

プログラムによるアクセス権を必要とするユーザー	目的	方法
ワークフォースアイデンティティ (IAM アイデンティティセンターで管理されているユーザー)	一時的な認証情報を使用して、AWS CLI、AWS SDK、または AWS API へのプログラムによるリクエストに署名します。	使用するインターフェイスの指示に従ってください。 • AWS CLI については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM アイデンティティセンター を使用するための AWS CLI の設定」を参照してください。 • AWS SDK、ツール、および AWS API については、AWS SDK とツールリファレンスガイドの「IAM アイデンティティセンター認証」を参照してください。
IAM	一時的な認証情報を使用して、AWS CLI、AWS SDK、または AWS API へのプログラムによるリクエストに署名します。	IAM ユーザーガイドの「AWS リソースでの一時的な認証情報の使用」の指示に従ってください。

プログラムによるアクセス権を必要とするユーザー	目的	方法
IAM	(非推奨) 長期的な認証情報を使用して、AWS CLI、AWS SDK、または AWS API へのプログラムによるリクエストに署名します。	使用するインターフェイスの指示に従ってください。 • AWS CLI については、「AWS Command Line Interface ユーザーガイド」の「IAM ユーザー認証情報を使用した認証」を参照してください。 • AWS SDK とツールについては、「AWS SDK とツールリファレンスガイド」の「長期認証情報を使用して認証する」を参照してください。 • AWS API については、「IAM ユーザーガイド」の「IAM ユーザーのアクセスキーの管理」を参照してください。

次の例では、パスワードを無効にしてユーザーを作成します。

```
create user temp_creds_user password disable;
```

次の例では、既存のユーザーのパスワードを無効にします。

```
alter user temp_creds_user password disable;
```

3. CREATE GROUPを使用してデータベースユーザーグループを作成します。
4. GRANT コマンドを使用して、グループのアクセス権限を定義します。

ステップ 5: IAM 認証情報を使用するように JDBC または ODBC 接続を設定する

Amazon Redshift JDBC または ODBC ドライバーを使用して SQL クライアントを設定できます。このドライバーは、データベースユーザーの認証情報を作成し、SQL クライアントと Amazon Redshift データベース間の接続を確立するプロセスを管理します。

認証に アイデンティティプロバイダーを使用する場合は、認証情報プロバイダープラグインの名前を指定します。Amazon Redshift JDBC および ODBC ドライバーには、次の SAML ベースの認証情報プロバイダーのプラグインが含まれます。

- Active Directory フェデレーションサービス (AD FS)
- PingOne
- Okta
- Azure Client ID

  アイデンティティプロバイダーとして Microsoft Azure AD をセットアップする手順については、JDBC または ODBC シングルサインオン認証の設定を参照してください。

IAM 認証情報を使用するように JDBC 接続を設定するには

1. 「Amazon Redshift での JDBC ドライバーバージョン 2.x による接続の構成」ページから最新の Amazon Redshift JDBC ドライバーをダウンロードします。
2. 次のいずれかの形式で IAM 認証情報オプションを使用して JDBC URL を作成します。IAM 認証を使用するには、次の例に示すように、`jdbc:redshift:` に続いて Amazon Redshift JDBC URL に `iam:` を追加します。

   ```
   jdbc:redshift:iam://
   ```

   `cluster-name`、`region`、および `account-id` を追加します。JDBC ドライバーでは IAM アカウント情報とクラスター名を使用することで、クラスター IDおよび AWS リージョンを取得できます。そのためには、ユーザーまたはロールには、指定したクラスターで `redshift:DescribeClusters` オペレーションを呼び出すアクセス許可が必要です。ユーザーまたはロールに `redshift:DescribeClusters` オペレーションを呼び出すアクセス許可がない場合は、次の例に示すように、クラスター ID、AWS リージョン、およびポートを含めます。ポート番号はオプションです。

```
jdbc:redshift:iam://examplecluster.abc123xyz789.us-
west-2.redshift.amazonaws.com:5439/dev
```

3. JDBC オプションを追加して IAM 認証情報を提供します。さまざまな組み合わせの JDBC オプションを使用して、IAM 認証情報を提供できます。詳細については、「データベースユーザー認証情報を作成するための JDBC および ODBC のオプション」を参照してください。

   次の URL は、ユーザーの AccessKeyID および SecretAccessKey を指定します。

   ```
   jdbc:redshift:iam://examplecluster:us-west-2/dev?
   AccessKeyID=AKIAIOSFODNN7EXAMPLE&SecretAccessKey=wJalrXUtnFEMI/K7MDENG/
   bPxRfiCYEXAMPLEKEY
   ```

   次の例では、IAM 認証情報を含む名前付きプロファイルを指定します。

   ```
   jdbc:redshift:iam://examplecluster:us-west-2/dev?Profile=user2
   ```

4. JDBC ドライバーを使用して `GetClusterCredentials` API オペレーションを呼び出す JDBC オプションを追加します。`GetClusterCredentials` API オペレーションをプログラムで呼び出す場合、これらのオプションは含めないでください。

   次の例では JDBC `GetClusterCredentials` オプションを含めています。

   ```
   jdbc:redshift:iam://examplecluster:us-west-2/dev?
   plugin_name=com.amazon.redshift.plugin.AzureCredentialsProvider&UID=user&PWD=password&idp_t
   ```

IAM 認証情報を使用するように ODBC 接続を設定するには

次の手順では、IAM 認証を設定するステップのみを示します。データベースユーザー名とパスワードを使用して標準認証を使用するステップについては、Amazon Redshift での ODBC ドライバーのバージョン 2.x による接続の設定を参照してください。

1. オペレーティングシステム用の最新の Amazon Redshift OBDC ドライバーをインストールおよび設定します。詳細については、Amazon Redshift での ODBC ドライバーのバージョン 2.x による接続の設定 ページを参照してください。

> **Important**
>
> Amazon Redshift ODBC ドライバーはバージョン 1.3.6.1000 以降である必要があります。

2. 接続設定に関するオペレーティングシステムのステップに従います。
3. Microsoft Windows オペレーティングシステムでは、Amazon Redshift ODBC ドライバーの DSN 設定ウィンドウにアクセスします。

   a. [接続設定] で以下の情報を入力します。

      - データソース名
      - サーバー (オプション)
      - ポート (オプション)
      - データベース

      ユーザーまたはロールに `redshift:DescribeClusters` オペレーションを呼び出すアクセス許可がある場合は、データソース名とデータベースのみが必要です。Amazon Redshift は ClusterId と リージョン を使用して、`DescribeCluster` オペレーションを呼び出すことでサーバーとポートを取得します。

      ユーザーまたはロールに `redshift:DescribeClusters` オペレーションを呼び出すアクセス許可がない場合は、サーバーとポートを指定します。

   b. Authentication (認証)で、Auth Type (認証タイプ)の値を選択します。

      認証タイプごとに、次のように値を入力します。

      AWS プロファイル

         次の情報を入力します。

         - ClusterID
         - リージョン
         - プロファイル名

           ODBC 接続オプションの値を含む AWS 設定ファイルのプロファイル名を入力します。詳細については、「設定プロファイルを使用する」を参照してください。

(任意) `GetClusterCredentials` API オペレーションの呼び出すために ODBC ドライバーが使用するオプションの詳細を指定します。

- DbUser
- User AutoCreate
- DbGroups

詳細については、データベースユーザー認証情報を作成するための JDBC および ODBC のオプション を参照してください。

IAM 認証情報

次の情報を入力します。

- ClusterID
- リージョン
- AccessKeyID と SecretAccessKey

  IAM データベース認証用に設定された IAM ロールまたはユーザーのアクセスキー ID とシークレットアクセスキー。
- SessionToken

  SessionToken は、一時的認証情報を持つ IAM ロールに必要です。詳細については Temporary Security Credentialsを参照してください。

`GetClusterCredentials` API オペレーションを呼び出すために ODBC ドライバーが使用するオプションの詳細を指定します。

- DbUser (必須)
- User AutoCreate (オプション)
- DbGroups (オプション)

詳細については、データベースユーザー認証情報を作成するための JDBC および ODBC のオプション を参照してください。

アイデンティティプロバイダ : AD FS

AD FS を使用した Windows 統合認証の場合、ユーザー および パスワード は空白にしておきます。

IdP の詳細を指定します。

- IdP Host

  企業アイデンティティプロバイダホストの名前。この名前には、スラッシュ (/) を含めないでください。

- IdP Port (任意)

  アイデンティティプロバイダが使用するポート。デフォルトは 443です。

- 優先ロール

  SAML アサーションの `AttributeValue` 属性の複数の値を持つ `Role` 要素からの IAM ロールの Amazon リソースネーム (ARN)。IdP 管理者と連携して、目的のロールに適した値を見つけます。詳細については、「ステップ 2: IdP の SAML アサーションを設定する」を参照してください。

(任意) `GetClusterCredentials` API オペレーションの呼び出すために ODBC ドライバーが使用するオプションの詳細を指定します。

- DbUser
- User AutoCreate
- DbGroups

詳細については、「データベースユーザー認証情報を作成するための JDBC および ODBC のオプション」を参照してください。

ID プロバイダー : PingFederate

[ユーザー] と [パスワード] に、IdP のユーザー名とパスワードを入力します。

IdP の詳細を指定します。

- IdP Host

企業アイデンティティプロバイダホストの名前。この名前には、スラッシュ (/) を含めないでください。

- IdP Port (任意)

  アイデンティティプロバイダが使用するポート。デフォルトは 443です。

- 優先ロール

  SAML アサーションの `AttributeValue` 属性の複数の値を持つ `Role` 要素からの IAM ロールの Amazon リソースネーム (ARN)。IdP 管理者と連携して、目的のロールに適した値を見つけます。詳細については、「ステップ 2: IdP の SAML アサーションを設定する」を参照してください。

(任意) `GetClusterCredentials` API オペレーションの呼び出すために ODBC ドライバーが使用するオプションの詳細を指定します。

- DbUser
- User AutoCreate
- DbGroups

詳細については、データベースユーザー認証情報を作成するための JDBC および ODBC のオプション を参照してください。

アイデンティティプロバイダー : Okta

ユーザー と パスワードに、IdP のユーザー名とパスワードを入力します。

IdP の詳細を指定します。

- IdP Host

  企業アイデンティティプロバイダホストの名前。この名前には、スラッシュ (/) を含めないでください。

- IdP Port

  この値は Okta では使用されません。

- 優先ロール

  SAML アサーションの `AttributeValue` 属性の `Role` 要素からの IAM ロールの Amazon リソースネーム (ARN)。IdP 管理者と連携して、目的のロールに適した値を

見つけます。詳細については、「ステップ 2: IdP の SAML アサーションを設定する」を参照してください。

- Okta App ID

  Okta アプリケーションの ID。アプリ ID の値は、Okta アプリケーションの埋め込みリンクの「amazon_aws」に続きます。IdP 管理者と連携して、この値を取得します。

(任意) `GetClusterCredentials` API オペレーションの呼び出すために ODBC ドライバーが使用するオプションの詳細を指定します。

- DbUser
- User AutoCreate
- DbGroups

詳細については、データベースユーザー認証情報を作成するための JDBC および ODBC のオプション を参照してください。

アイデンティティプロバイダー : Azure AD

ユーザー と パスワードに、IdP のユーザー名とパスワードを入力します。

クラスター ID と リージョンにクラスター ID と Amazon Redshift クラスターの AWS リージョンを入力します。

データベースに、Amazon Redshift クラスター用に作成したデータベースを入力します。

IdP の詳細を指定します。

- IdP Tenant

  Azure AD に使用されるテナント。

- Azure Client Secret

  Azure の Amazon Redshift エンタープライズアプリのクライアントシークレット。

- Azure Client ID

  Azure の Amazon Redshift エンタープライズアプリケーションのクライアント ID (アプリケーション ID)。

(任意) `GetClusterCredentials` API オペレーションの呼び出すために ODBC ドライバーが使用するオプションの詳細を指定します。

- DbUser
- User AutoCreate
- DbGroups

詳細については、データベースユーザー認証情報を作成するための JDBC および ODBC のオプション を参照してください。

IAM 認証情報の提供オプション

JDBC または ODBC 接続の IAM 認証情報を提供するには、次のいずれかの認証タイプを選択します。

- AWS プロファイル

  JDBC または ODBC 設定形式で認証情報の値を指定する代わりとして、名前付きプロファイルで値を指定できます。詳細については、設定プロファイルを使用する を参照してください。

- IAM 認証情報

  JDBC または ODBC 設定の形式で、AccessKeyID、SecretAccessKey、および必要に応じて SessionToken の値を指定します。SessionToken は、一時的認証情報を持つ IAM ロールにのみ必要です。詳細については、IAM 認証情報を提供するための JDBC および ODBC のオプション を参照してください。

- アイデンティティプロバイダーフェデレーション

  アイデンティティプロバイダーフェデレーションを使用して、アイデンティティプロバイダーのユーザーを Amazon Redshift で認証できるようにする場合は、認証情報プロバイダープラグインの名前を指定します。詳細については、認証情報プロバイダープラグイン を参照してください。

  Amazon Redshift JDBC および ODBC ドライバーには、次の SAML ベースの ID フェデレーション認証情報プロバイダーのプラグインが含まれます。

  - Microsoft Active Directory Federation Services (AD FS)
  - PingOne
  - Okta
  - Microsoft Azure Active Directory (Azure AD)

JDBC または ODBC 設定の形式、またはプロファイルを使用して、プラグイン名前と関連する値を指定できます。詳細については、「JDBC ドライバーバージョン 2.x の設定オプション」を参照してください。

詳細については、「ステップ 5: IAM 認証情報を使用するように JDBC または ODBC 接続を設定する」を参照してください。

設定プロファイルを使用する

GetClusterCredentials 設定ファイルの名前付きプロファイルで、IAM 認証情報オプションと AWS オプションを設定として指定できます。プロファイルの JDBC オプションを使用してプロファイル名を指定します。この設定は、ホームディレクトリの config という名前のフォルダにある credentials または .aws というファイルに保存されます。

Amazon Redshift JDBC ドライバーまたは ODBC ドライバーに含まれる SAML ベースの認証情報プロバイダプラグインの場合は、前述の 認証情報プロバイダープラグインで説明した設定を使用できます。plugin_name を使用しない場合、他のオプションは無視されます。

2 つのプロファイルのある ~/.aws/credentials ファイルの例を以下に示します。

```
[default]
aws_access_key_id=AKIAIOSFODNN7EXAMPLE
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

[user2]
aws_access_key_id=AKIAI44QH8DHBEXAMPLE
aws_secret_access_key=je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY
session_token=AQoDYXdzEPT//////////
wEXAMPLEtc764bNrC9SAPBSM22wDOk4x4HIZ8j4FZTwdQWLWsKWHGBuFqwAeMicRXmxfpSPfIeoIYRqTflfKD8YUuwthAx7
qkPpKPi/kMcGd
QrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburEDy0KPkyQDYwT7WZ0wq5VSXDvp75YU
9HFvlRd8Tx6q6fE8YQcHNVXAkiY9q6d+xo0rKwT38xVqr7ZD0u0iPPkUL64lIZbqBAz
+scqKmlzm8FDrypNC9Yjc8fPOLn9FX9KSYvKTr4rvx3iSIlTJabIQwj2ICCR/oLxBA==
```

user2 の例の認証情報を使用するには、JDBC URL で Profile=user2 を指定します。

プロファイルの使用についての詳細は、「AWS Command Line Interface ユーザーガイド」の「設定ファイルと認証情報ファイルの設定」を参照してください。

JDBC ドライバーのプロファイルを使用する方法の詳細については、「プロファイルの指定」を参照してください。

ODBC ドライバーのプロファイルの使用方法の詳細については、「認証方法」を参照してください。

IAM 認証情報を提供するための JDBC および ODBC のオプション

以下の表に、IAM 認証情報を提供するための JDBC および ODBC のオプションを示します。

オプション	説明
`Iam`	ODBC 接続文字列内でのみ使用できます。IAM 認証を使用するには、1 に設定します。

`AccessKeyID`
 	
`SecretAccessKey`
 	
`SessionToken`
	IAM データベース認証用に設定された IAM ロールまたはユーザーのアクセスキー ID とシークレットアクセスキー。`SessionToken` は、一時的な認証情報を持つ IAM ロールにのみ必要です。ユーザーには SessionToken を使用しません。詳細については、 Temporary Security Credentialsを参照してください。

`plugin_name`	認証情報プロバイダーを実装する完全修飾クラス名。Amazon Redshift JDBC ドライバーには SAML ベースの認証情報プロバイダープラグインが含まれます。`plugin_name` を提供する場合は、他の関連オプションも提供できます。詳細については、「認証情報プロバイダープラグイン」を参照してください。
`Profile`	JDBC 接続オプションの値を含む AWS 認証情報または設定ファイルのプロファイル名。詳細については、「設定プロファイルを使用する」を参照してください。

データベースユーザー認証情報を作成するための JDBC および ODBC のオプション

Amazon Redshift JDBC または ODBC ドライバを使用してデータベースユーザー認証情報を作成するには、JDBC または ODBC オプションとしてデータベースユーザー名を指定します。オプションで、存在しない場合は新しいデータベースユーザーをドライバで作成し、ユーザーがログイン時に参加するデータベースユーザーグループのリストを指定できます。

ID プロバイダ (IdP) を使用する場合は、IdP 管理者と連携して、これらのオプションに合った正しい値を確認します。IdP 管理者は、JDBC または ODBC のオプションとして提供する必要がない場合、それらのオプションを提供するように IdP を設定できます。詳細については、「ステップ 2: IdP の SAML アサーションを設定する」を参照してください。

> **Note**
>
> `${redshift:DbUser}`で説明されているように、IAM ポリシー変数 GetClusterCredentials のリソースポリシー を使用する場合、`DbUser`の値は API オペレーションのリクエストコンテキストから取得した値で置き換えられます。Amazon Redshift ドライバは、SAML 属性として提供される値ではなく、接続 URL から提供される `DbUser` 変数の値を使用します。この接続の安全性を維持するには、IAM ポリシーの条件を使用して、`DbUser` 値を `RoleSessionName`で検証することが推奨されます。IAM ポリシーを使用して条件を設定する方法の例は、例 8: GetClusterCredentials を使用する IAM ポリシーで参照できます。

次の表は、データベースユーザー認証情報を作成するためのオプションを示しています。

オプション	説明
DbUser	データベースユーザーの名前。DbUser というユーザーがデータベースに存在する場合、一時的ユーザー認証情報のアクセス許可は、既存のユーザーと同じになります。DbUser がデータベースに存在せず、AutoCreate が true の場合、DbUser という名前の新しいユーザーが作成されます。必要に応じて、既存ユーザーのパスワードを無効にします。詳細については、ALTER_USERを参照してください。
AutoCreate	DbUser に指定した名前でデータベースユーザーを作成するには (存在しない場合)、`true` を指定します。デフォルト: false。
`DbGroups`	データベースユーザーが現在のセッションで参加する 1 つまたは複数の既存データベースグループの名前のカンマ区切りリスト。デフォルトでは、新しいユーザーは PUBLIC のみに追加されます。

認証情報プロバイダープラグイン

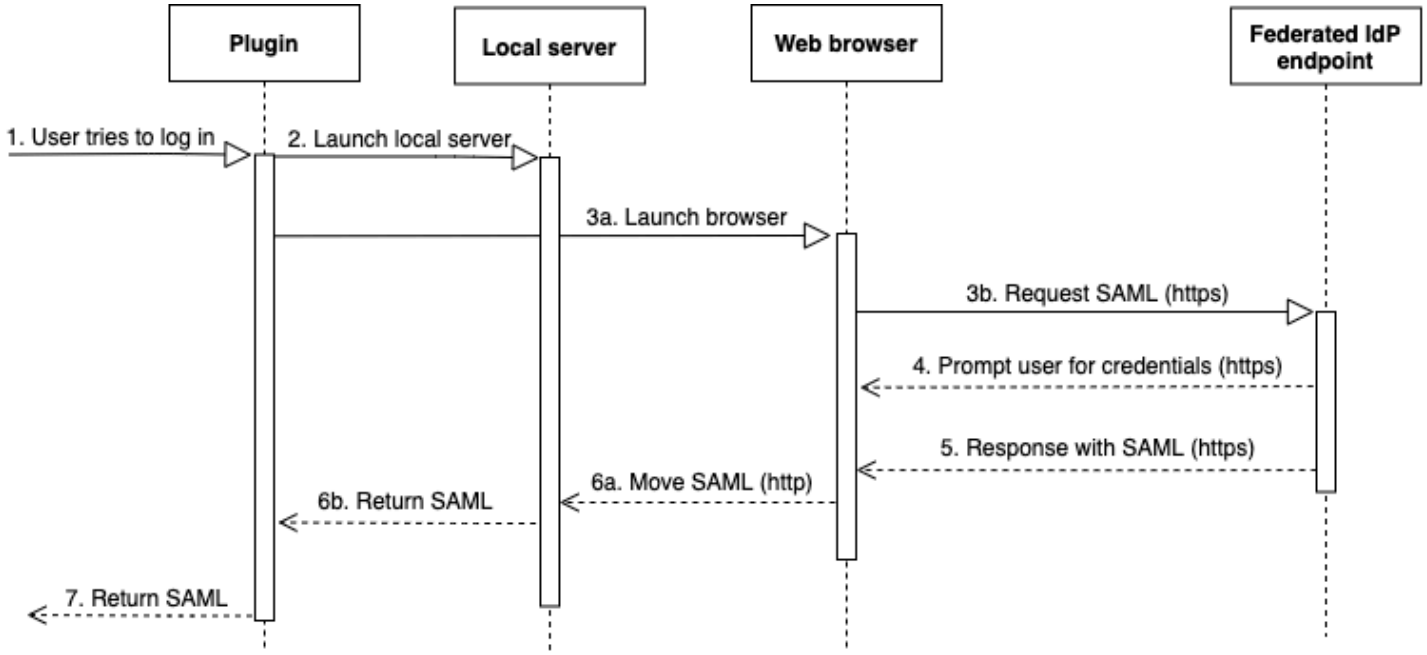
Amazon Redshift はシングルサインオン認証に認証情報プロバイダープラグインを使用します。

シングルサインオン認証をサポートするために、Amazon Redshift は Microsoft Azure Active Directory 用の Azure AD プラグインを提供します。このプラグインの設定方法については、JDBC または ODBC シングルサインオン認証の設定を参照してください。

多要素認証
多要素認証

多要素認証 (MFA) をサポートするために、Amazon Redshift にはブラウザベースのプラグインが用意されています。Okta、PingOne、Active Directory Federation Services にはブラウザ SAML プラグインを使用し、Microsoft Azure Active Directory にはブラウザ Azure AD プラグインを使用します。

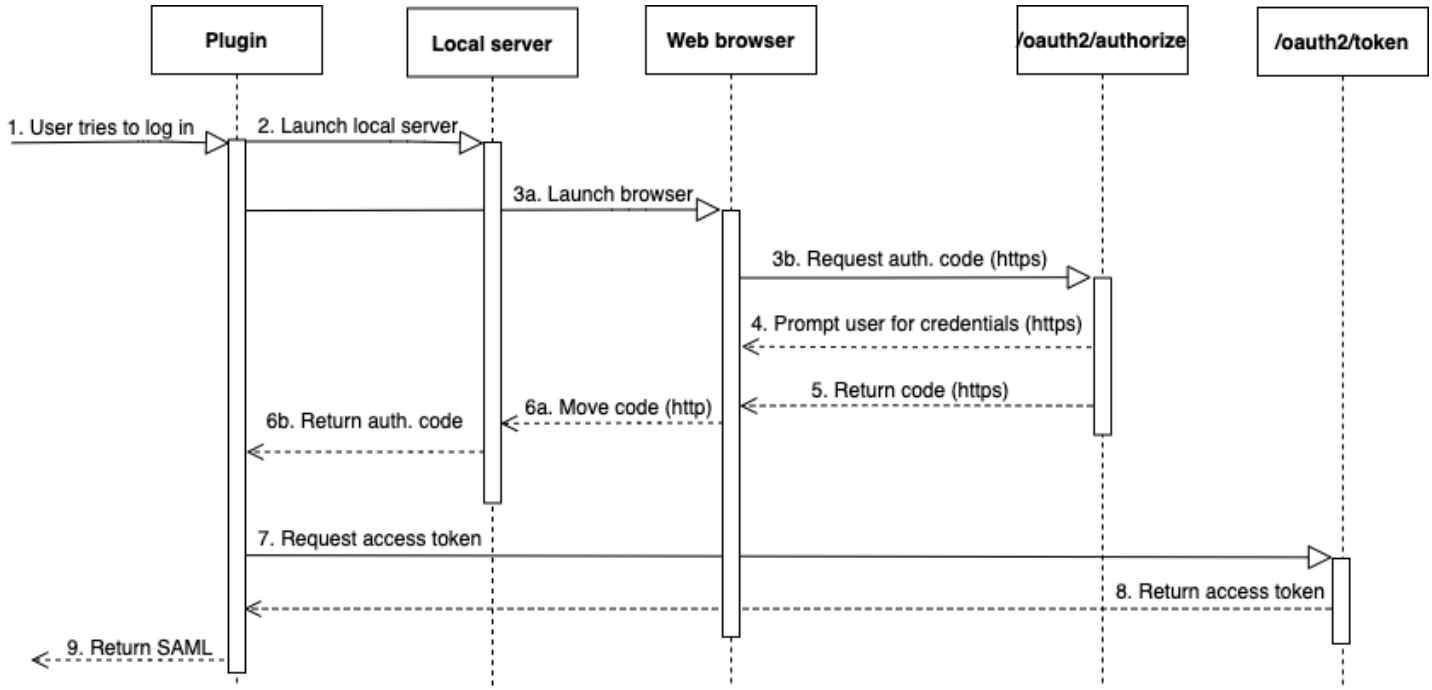
ブラウザの SAML プラグインでは、OAuth 認証は次のようになります。



1. ユーザーがログインを試行します。
2. プラグインは、ローカルホスト上の着信接続をリッスンするローカルサーバーを起動します。
3. プラグインはウェブブラウザを起動し、指定されたシングルサインオンログイン URL フェデレーテッド ID プロバイダーのエンドポイントから HTTPS 経由で SAML 応答を要求します。
4. ウェブブラウザはリンクをたどり、ユーザーに認証情報の入力を要求します。
5. ユーザーが認証され、同意が付与されると、フェデレーテッド ID プロバイダーのエンドポイントは、`redirect_uri`に示される URI に HTTPS 経由で SAML 応答を返します。
6. ウェブブラウザは、SAML 応答を含む応答メッセージを、指定された `redirect_uri`に移動します。

7. ローカルサーバーは着信接続を受け入れ、プラグインは SAML 応答を取得して Amazon Redshift に渡します。

ブラウザ Azure AD プラグインでは、SAML 認証は次のようになります。



1. ユーザーがログインを試行します。
2. プラグインは、ローカルホスト上の着信接続をリッスンするローカルサーバーを起動します。
3. プラグインはウェブブラウザを起動して、Azure AD `oauth2/authorize` エンドポイントから認可コードを要求します。
4. ウェブブラウザは、HTTPS 経由で生成されたリンクをたどり、ユーザーに認証情報の入力を要求します。リンクは、テナントや client_id などの設定プロパティを使用して生成されます。
5. ユーザーが認証され、同意が付与されると、Azure AD `oauth2/authorize` エンドポイントは HTTPS 経由で応答を返し、認可コードとともに示された `redirect_uri` に送信します。
6. ウェブブラウザは、SAML レスポンスを含む応答メッセージを、指定された `redirect_uri`に移動します。
7. ローカルサーバーは、着信接続とプラグイン要求を受け入れ、認可コードを取得し、Azure AD `oauth2/token` エンドポイントに POST リクエストを送信します。
8. Azure AD `oauth2/token` エンドポイントは、アクセストークンを含む応答を、示された `redirect_uri`に返します。
9. プラグインは SAML 応答を取得し、Amazon Redshift に渡します。

以下のセクションを参照してください。

- Active Directory フェデレーションサービス (AD FS)

  詳細については、「JDBC または ODBC シングルサインオン認証の設定」を参照してください。
- PingOne (Ping)

  Ping は、フォーム認証を使用する既定の PingOne IdP Adapter でのみサポートされます。

  詳細については、「JDBC または ODBC シングルサインオン認証の設定」を参照してください。
- Okta

  Okta は、AWS マネジメントコンソールで使用される Okta 提供アプリケーションでのみサポートされます。

  詳細については、「JDBC または ODBC シングルサインオン認証の設定」を参照してください。
- Microsoft Azure Active Directory (Azure AD)

  詳細については、「JDBC または ODBC シングルサインオン認証の設定」を参照してください。

プラグインオプション
プラグインオプション

SAML ベースの認証情報プロバイダープラグインを使用するには、名前付きプロファイルで JDBC または ODBC オプションを使用して次のオプションを指定します。plugin_name が指定されていない場合、ほかのオプションは無視されます。

オプション	説明
plugin_name	JDBC の場合、認証情報プロバイダを実装するクラス名。次のいずれかを指定します。 • Active Directory Federation Services の場合 `com.amazon.redshift.plugin.AdfsCredentialsProvider` • Okta の場合

オプション	説明

	`com.amazon.redshift.plugin.OktaCredentialsProvider`
	• PingFederate の場合
	`com.amazon.redshift.plugin.PingCredentialsProvider`
	• Microsoft Azure Active Directory (Azure AD) の場合
	`com.amazon.redshift.plugin.AzureCredentialsProvider`
	• SAML MFA の場合
	`com.amazon.redshift.plugin.BrowserSamlCredentialsProvider`
	• MFA を使用した Microsoft Azure Active Directory シングルサインオンの場合
	`com.amazon.redshift.plugin.BrowserAzureCredentialsProvider`
	
	ODBC の場合、次のいずれかを指定します。
	• Active Directory Federation Services の場合: `adfs`
	• Okta の場合: `okta`
	• PingFederate の場合: `ping`
	• Microsoft Azure Active Directory の場合: `azure`
	• SAML MFA の場合: `browser saml`
	• MFA を使用した Microsoft Azure Active Directory シングルサインオンの場合: `browser azure ad`

`idp_host`	企業アイデンティティプロバイダホストの名前。この名前には、スラッシュ (/) を含めないでください。Okta ID プロバイダーの場合、`idp_host` の値は `.okta.com` で終わる必要があります。
`idp_port`	アイデンティティプロバイダーが使用するポート。デフォルトは 443 です。ポートは Okta に対して無視されます。

オプション	説明
preferred_role	SAML アサーションの AttributeValue 属性の Role 要素からのロールの Amazon リソースネーム (ARN)。IdP 管理者と連携して、目的のロールに適した値を見つけます。詳細については、「ステップ 2: IdP の SAML アサーションを設定する」を参照してください。
user	該当する場合はドメインを含む、社内ユーザー名。たとえば、Active Directory の場合、必要なドメイン名の形式は domain\usernameです。
password	社内ユーザーのパスワード。このオプションを使用しないことをお勧めします。代わりに、SQL クライアントを使用してパスワードを指定します。
app_id	Okta アプリケーションの ID。Okta でのみ使用します。app_id の値は、Okta アプリケーションの埋め込みリンクの amazon_aws に続きます。IdP 管理者と連携して、この値を取得します。以下に、アプリケーション埋め込みリンクの例を示します。:-https://example.okta.com/home/amazon_aws/0oa2hylwrpM8UGehd1t7/272
idp_tenant	Azure AD で使用されるテナント。Azure でのみ使用します。
client_id	Azure AD の Amazon Redshift エンタープライズアプリケーションのクライアント ID。Azure でのみ使用します。

Amazon Redshift CLI または API を使用して IAM ID のデータベース認証情報を生成する

プログラムで一時的なデータベースユーザー認証情報を生成するため、Amazon Redshift では AWS Command Line Interface (AWS CLI) および GetClusterCredentials API オペレーション用の get-cluster-credentials コマンドを利用できます。または、GetClusterCredentials オペレーションを呼び出し、データベースユーザー認証情報を取得して、SQL クライアントと Amazon Redshift データベース間の接続を確立するプロセスを管理する、Amazon Redshift JDBC または ODBC ドライバーを使用して SQL クライアントを設定できます。詳細については、データベースユーザー認証情報を作成するための JDBC および ODBC のオプション を参照してください。

> **Note**
>
> Amazon Redshift JDBC または ODBC ドライバを使用して、データベースユーザー認証情報を生成することをお勧めします。

このセクションでは、プログラムで `GetClusterCredentials` オペレーションまたは get-cluster-credentials コマンドを呼び出し、データベースのユーザー認証情報を取得して、データベースに接続する手順を示します。

一時的データベース認証情報を生成して使用するには

1. 必要なアクセス許可を使用してユーザーまたはロールを作成または変更します。IAM 許可の詳細については、「ステップ 3: GetClusterCredentialsWithIAM または GetClusterCredentials を呼び出すアクセス許可を持つ IAM ロールを作成する」を参照してください。
2. 前のステップで認可したユーザーまたはロールとして、get-cluster-credentials CLI コマンドを実行するか、`GetClusterCredentials` API オペレーションを呼び出して以下の値を指定します。

   - クラスター識別子 – データベースを含むクラスターの名前。
   - データベースユーザー名 – 既存の、または新しいデータベースユーザーの名前。
     - ユーザーがデータベースに存在せず、AutoCreate が true の場合、PASSWORD を無効にして新しいユーザーが作成されます。
     - ユーザーが存在せず、AutoCreate が false の場合、要求は失敗します。
     - この例では、データベースユーザー名は `temp_creds_user`です。
   - Autocreate – (オプション) データベースユーザー名が存在しない場合は新しいユーザーを作成します。
   - データベース名 – (オプション) ユーザーがログオンする権限が付与されているデータベースの名前。データベース名を指定しない場合、ユーザーは任意のクラスターデータベースにログインできます。
   - データベースグループ – (オプション) 既存のデータベースユーザーグループのリスト。正常にログインすると、データベースユーザーが、指定されたユーザーグループに追加されます。グループを指定しない場合、ユーザーは PUBLIC 許可のみを持ちます。ユーザーグループ名は、ユーザーまたはロールにアタッチされた IAM ポリシーに指定されている dbgroup リソース ARN と一致する必要があります。

- 有効期限 – (任意) 一時的認証情報が期限切れになるまでの時間 (秒単位)。900 秒 (15 分) と 3600 秒 (60 分間) の間の値を指定できます。デフォルト値は 900 秒です。

3. Amazon Redshift は、指定したリソースで GetClusterCredentials オペレーションを呼び出すアクセス許可をユーザーが持っていることを確認します。
4. Amazon Redshift は一時パスワードとデータベースユーザー名を返します。

   次の例は、Amazon Redshift CLI を使用して、temp_creds_userという既存のユーザー用に一時的データベース認証情報を生成します。

```
aws redshift get-cluster-credentials --cluster-identifier examplecluster --db-user
 temp_creds_user --db-name exampledb --duration-seconds 3600
```

   結果は次のようになります。

```
{
  "DbUser": "IAM:temp_creds_user",
  "Expiration": "2016-12-08T21:12:53Z",
  "DbPassword": "EXAMPLEjArE3hcnQj8zt4XQj9Xtma8oxYEM8OyxpDHwXVPyJYBDm/
gqX2Eeaq6P3DgTzgPg=="
}
```

   次の例では、Amazon Redshift CLI と autocreate を使用して、新しいユーザー用の一時的データベース認証情報を生成し、そのユーザーをグループ example_groupに追加します。

```
aws redshift get-cluster-credentials --cluster-identifier examplecluster --db-user
 temp_creds_user --auto-create --db-name exampledb --db-groups example_group --
duration-seconds 3600
```

   結果は次のようになります。

```
{
  "DbUser": "IAMA:temp_creds_user:example_group",
  "Expiration": "2016-12-08T21:12:53Z",
  "DbPassword": "EXAMPLEjArE3hcnQj8zt4XQj9Xtma8oxYEM8OyxpDHwXVPyJYBDm/
gqX2Eeaq6P3DgTzgPg=="
}
```

5. Amazon Redshift クラスターとの Secure Sockets Layer (SSL) 認証接続を確立し、GetClusterCredentialsの応答からのユーザー名とパスワードとともにログイン要

求を送信します。ユーザー名には IAM: または IAMA: というプレフィックスを含めます (例: IAM:temp_creds_user または IAMA:temp_creds_user)。

> **Important**
>
> SSL を必要とするように SQL クライアントを設定します。それ以外の場合、SQL クライアントが自動的に SSL との接続を試みた場合、なんらかの失敗があると、非 SSL にフォールバックする可能性があります。その場合、最初の接続の試行は、認証情報が期限切れまたは無効なために失敗する可能性があり、2 番目の接続の試行は、接続が SSL でないために失敗します。その場合、最初のエラーメッセージは表示されない可能性があります。SSL を使用したクラスターへの接続の詳細については、接続のセキュリティオプションを設定するを参照してください。

6. 接続で SSL を使用しない場合、接続の試行は失敗します。
7. クラスターは authentication 要求を SQL クライアントに送信します。
8. 次に、SQL クライアントは一時パスワードをクラスターに送信します。
9. パスワードが有効で期限切れでない場合、クラスターは接続を完了します。

JDBC または ODBC シングルサインオン認証の設定

外部 ID プロバイダー (IdP) を利用して、Amazon Redshift クラスターにアクセスするユーザーを認証および承認することで、ユーザー管理を簡素化してセキュリティを強化できます。これにより、複数のサービスにわたって、一元化されたユーザー管理、ロールベースのアクセスコントロール、監査機能が可能になります。一般的なユースケースとして、さまざまなユーザーグループの認証の合理化、一貫したアクセスポリシーの適用、規制要件の遵守などがあります。

以下のページでは、Redshift クラスターとの IdP 統合の設定について説明します。AWS を IdP のサービスプロバイダーとして設定する方法については、IAM ユーザーガイドの 証明書利用者の信頼とクレームの追加によって SAML 2.0 IdP を設定する を参照してください。

AD FS

このチュートリアルでは、AD FS を ID プロバイダー (IdP) として使用して、Amazon Redshift クラスターにアクセスする方法を説明します。

ステップ 1: AD FS と AWS アカウントを相互に信頼するように設定する

以下の手順では、信頼関係を設定する方法を説明します。

1. AD FS ユーザーが接続するための既存の Amazon Redshift クラスターを作成または使用します。接続を設定するには、クラスター識別子など、このクラスターの特定のプロパティが必要です。詳細については、「クラスターの作成」を参照してください。
2. Microsoft マネジメントコンソールで Amazon Redshift アクセスを制御するために AD FS を設定します。

    1. ADFS 2.0を選択し、証明書利用者信頼の追加を選択します。証明書利用者信頼の追加ウィザード ページで、開始を選択します。
    2. データソースの選択 ページで、オンラインまたはローカルネットワーク上で公開されている証明書利用者に関するデータのインポートを選択します。
    3. フェデレーションメタデータアドレス (ホスト名またはURL) に **https://signin.aws.amazon.com/saml-metadata.xml**と入力します。メタデータ XML ファイルは、AWS を証明書利用者として記述する標準の SAML メタデータドキュメントです。
    4. 表示名の指定 ページで、表示名に値を入力します。
    5. 発行認可ルールの選択 ページで、この証明書利用者へのアクセスをすべてのユーザーに許可または拒否する発行認可ルールを選択します。
    6. 信頼の追加の準備 ページで、設定を確認します。
    7. 終了 ページで、ウィザードが終了すると、この証明書利用者信頼のための [クレームルールの編集] ダイアログが開きますを選択します。
    8. コンテキスト (右クリック) メニューで、証明書利用者信頼を選択します。
    9. 証明書利用者については、コンテキスト (右クリック) メニューを開き、クレームルールの編集を選択します。[クレームルールの編集] ページで、[ルールを追加] を選択します。
    10. クレームルールテンプレートで、受信クレームの変換を選択し、ルールの編集 – NameIdページで、次の操作を行います。
        - クレームルール名に NameIdを入力します。
        - 受信クレーム名で、Windows アカウント名を選択します。
        - 発信クレーム名で、名前 IDを選択します。
        - 発信者名 ID フォームで、永続的識別子を選択します。
        - すべてのクレーム値を通過を選択します。
    11. [クレームルールの編集] ページで、[ルールを追加] を選択します。ルールテンプレートの選択ページの クレームルールテンプレートで、LDAP 属性をクレームとして送信を選択します。
    12. ルールの設定 ページで、次の手順を実行します。
        - クレームルール名に、RoleSessionNameと入力します。

- 属性を保存するで アクティブディレクトリを選択します。
- LDAP 属性で E-Mail-Addressesを選択します。
- 送信クレームタイプで https://aws.amazon.com/SAML/Attributes/RoleSessionNameを選択します。

13[クレームルールの編集] ページで、[ルールを追加] を選択します。[ルールテンプレートの選択] ページの [クレームルールテンプレート]で、[カスタムルールによるクレーム送信] を選択します。

14ルールの編集 - AD グループの取得 ページで、クレームルール名に AD グループの取得と入力します。

15カスタムルールで、次のように入力します。

```
c:[Type ==
                                    "http://schemas.microsoft.com/ws/2008/06/
identity/claims/windowsaccountname",
                                    Issuer == "AD AUTHORITY"] => add(store =
 "Active Directory",
                                    types = ("http://temp/variable"), query =
 ";tokenGroups;{0}",
                                    param = c.Value);
```

16[クレームルールの編集] ページで、[ルールを追加] を選択します。[ルールテンプレートの選択] ページの [クレームルールテンプレート]で、[カスタムルールによるクレーム送信] を選択します。

17ルールの編集 - ロール ページの クレームルール名に ロールと入力します。

18カスタムルールで、次のように入力します。

```
c:[Type == "http://temp/variable", Value =~ "(?i)^AWS-"] =>
 issue(Type = "https://aws.amazon.com/SAML/Attributes/Role", Value =
 RegExReplace(c.Value, "AWS-", "arn:aws:iam::123456789012:saml-provider/
ADFS,arn:aws:iam::123456789012:role/ADFS-"));
```

引き受ける SAML プロバイダーとロールの ARN をメモします。この例では、`arn:aws:iam:123456789012:saml-provider/ADFS`は SAML プロバイダーの ARN で、`arn:aws:iam:123456789012:role/ADFS-`はロールの ARN です。

3. `federationmetadata.xml` ファイルをダウンロードしたことを確認します。ドキュメントの内容に無効な文字が含まれていないことを確認してください。これは、AWSとの信頼関係を設定する際に使用するメタデータファイルです。

4. IAM コンソールで、IAM SAML アイデンティティプロバイダーを作成します。指定したメタデータドキュメントは、Azure エンタープライズアプリケーションの設定時に保存したフェデレーションメタデータ XML ファイルです。詳細な手順については、「IAM ユーザーガイド」の「IAM ID プロバイダーを作成および管理する (コンソール)」を参照してください。
5. IAM コンソールで、SAML 2.0 フェデレーション用の IAM ロールを作成します。詳細なステップについては、「IAM ユーザーガイド」の「SAML 用のロールの作成」を参照してください。
6. IAM コンソールで SAML 2.0 フェデレーション用に作成した IAM ロールにアタッチできる IAM ポリシーを作成します。詳細な手順については、「IAM ユーザーガイド」の「IAM ポリシーの作成 (コンソール)」を参照してください。Azure AD の例については、「JDBC または ODBC シングルサインオン認証の設定」を参照してください。

ステップ 2: AD FS への認証用に JDBC または ODBC を設定する

JDBC

以下の手順では、AD FS と JDBC の関係を設定する方法を説明します。

- AD FS シングルサインオンを使用して JDBC 経由でクラスターに接続するようにデータベースクライアントを構成します。

  JDBC ドライバーを使用する任意のクライアントを使用して AD FS シングルサインオンを使用して接続、または Java などの言語を使用してスクリプトを使用して接続することもできます。インストールおよび設定の詳細については、「Amazon Redshift での JDBC ドライバーバージョン 2.x による接続の構成」を参照してください。

  たとえば、SQLWorkbench/J をクライアントとして使用できます。SQLWorkbench/J を設定する場合、データベースの URL は次の形式を使用します。

  ```
  jdbc:redshift:iam://cluster-identifier:us-west-1/dev
  ```

  SQLWorkbench/J をクライアントとして使用する場合は、次の手順を実行します。

  a. 接続プロファイルの選択 ページで、プロファイルグループを追加します (例: **ADFS**)。
  b. 接続プロファイルに、接続プロファイル名を入力します (例: **ADFS**)。
  c. [ドライバの管理] を選択し、[Amazon Redshift] を選択します。[フォルダを開く] アイコン、[ライブラリ] を選択し、適切な JDBC .jar ファイルを選択します。
  d. 接続プロファイルの選択ページで、次のように接続プロファイルに情報を追加します。

- ユーザーには、AD FS ユーザー名を入力します。これは、使用認証しようとしているクラスターに対するアクセス許可を持つ、シングルサインオンに使用しているアカウントのユーザー名です。
- パスワードには、AD FS パスワードを入力します。
- [ドライバ] では、[Amazon Redshift (com.amazon.redshift.jdbc.Driver)] を選択します。
- [URL] には、**jdbc:redshift:iam://*your-cluster-identifier*:*your-cluster-region*/*your-database-name*** と入力します。

e. 拡張プロパティを選択します。[plugin_name] には、**com.amazon.redshift.plugin.AdfsCredentialsProvider** と入力します。この値は、認証方法として AD FS シングルサインオンを使用するドライバーに指定します。

ODBC

AD FS への認証用に ODBC を設定するには

- AD FS シングルサインオンを使用して ODBC 経由でクラスターに接続するようにデータベースクライアントを構成します。

  Amazon Redshift には Linux、Windows、macOS オペレーティングシステム用の ODBC ドライバが用意されています。ODBC ドライバをインストールする前に、SQL クライアントツールが 32 ビットであるか 64 ビットであるかを調べます。SQL クライアントツールの要件と一致する ODBC ドライバをインストールします。

  Windows の場合、Amazon Redshift ODBC ドライバ DSN セットアップ ページの [接続設定] で、次の情報を入力します。

  - [データソース名] には、***your-DSN*** と入力します。ODBC プロファイル名として使用するデータソース名を指定します。
  - Auth typeで アイデンティティプロバイダー: SAMLを選択します。これは、ODBC ドライバが AD FS シングルサインオンを使用した認証に使用する認証方法です。
  - [クラスター ID] には、***your-cluster-identifier*** と入力します。
  - [リージョン] には、***your-cluster-region*** と入力します。
  - [データベース] には、***your-database-name*** と入力します。

- [ユーザー] には、***your-adfs-username*** と入力します。これは、使用認証しようとしているクラスターに対するアクセス許可を持つ、シングルサインオンに使用している AD FS アカウントのユーザー名です。Auth type が ID プロバイダー: SAML である場合にのみ使用します。
- [パスワード] には、***your-adfs-password*** と入力します。Auth type が ID プロバイダー: SAMLである場合にのみ使用します。

macOS および Linux では、次のように `odbc.ini` ファイルを編集します。

> **Note**
>
> すべてのエントリでは、大文字と小文字は区別されません。

- [clusterid] には、***your-cluster-identifier*** と入力します。これは、作成された Amazon Redshift クラスターの名前です。
- [リージョン] には、***your-cluster-region*** と入力します。これは、作成された Amazon Redshift クラスターの AWS リージョンです。
- [データベース] には、***your-database-name*** と入力します。これは、Amazon Redshift クラスター上でアクセスしようとしているデータベースの名前です。
- [ロケール] には、**en-us** と入力します。これは、エラーメッセージが表示される言語です。
- [iam] には、**1** と入力します。この値は、IAM 認証情報を使用して認証するためにドライバに指定します。
- plugin_name には、次のいずれかの操作を行います。
  - MFA 設定を使用した AD FS シングルサインオンの場合は、**BrowserSAML** と入力します。これは、ODBC ドライバが AD FS への認証に使用する認証方法です。
  - AD FS シングルサインオン設定の場合は、**ADFS** と入力します。これは、ODBC ドライバが Azure AD シングルサインオンを使用した認証に使用する認証方法です。
- [uid] には、***your-adfs-username*** と入力します。これは、認証しようとしているクラスターに対するアクセス許可を持つ、シングルサインオンに使用しているMicrosoft Azure アカウントのユーザー名です。これは、plugin_name が ADFS である場合にのみ使用します。

- [pwd] には、***your-adfs-password*** と入力します。これは、plugin_name が ADFSである場合にのみ使用します。

macOS と Linux では、プロファイル設定を編集して、次のエクスポートも追加できます。

```
export ODBCINI=/opt/amazon/redshift/Setup/odbc.ini
```

```
export ODBCINSTINI=/opt/amazon/redshift/Setup/odbcinst.ini
```

Azure

Amazon Redshift クラスターにアクセスするためのアイデンティティプロバイダー (IdP) として Microsoft の Azure AD を使用できます。このチュートリアルでは、Azure を ID プロバイダー (IdP) として使用して、Amazon Redshift クラスターにアクセスする方法を説明します。

Microsoft Azure AD シングルサインオンで Amazon Redshift アクセスをフェデレーションする方法については、Amazon Redshift のアクセスを Microsoft Azure AD のシングルサインオンとフェデレーションする という動画をご覧ください。

ステップ 1: Azure と AWS アカウントを相互に信頼するように設定する

以下の手順では、信頼関係を設定する方法を説明します。

相互に信頼するように Azure AD と AWS アカウントを設定するには

1. Azure AD ユーザーが接続するための既存の Amazon Redshift クラスターを作成または使用します。接続を設定するには、クラスター識別子など、このクラスターの特定のプロパティが必要です。詳細については、「クラスターの作成」を参照してください。
2. Microsoft Azure ポータルで AWS に使用される Azure Active Directory、グループ、ユーザーを設定します。
3. AWS コンソールへのシングルサインオンおよび Amazon Redshift へのフェデレーテッドログインに使用する Microsoft Azure ポータルに、エンタープライズアプリケーションとして Amazon Redshift を追加します。Enterprise application を選択します。
4. +New application を選択します。[アプリケーションを追加] ページが表示されます。
5. 検索フィールドで **AWS** を検索します。
6. Amazon Web Services (AWS) を選択し、追加 をクリックします。これにより、AWS アプリケーションが作成されます。

7. 管理 で、シングルサインオンを選択します。
8. SAML を選択します。[Amazon Web Services (AWS) | SAML ベースのサインオン] ページが表示されます。
9. はい を選択して、[SAML によるシングルサインオンのセットアップ] ページに進みます。このページには、事前設定されたシングルサインオン関連属性のリストが表示されます。
10. ベーシック SAML 設定 で、編集アイコンを選択し、保存 を選択します。
11. 複数のアプリケーションを設定する場合は、識別子の値を指定します。たとえば、***https://signin.aws.amazon.com/saml#2***と入力します。2 番目のアプリケーション以降では、この形式を # 記号とともに使用して、一意の SPN 値を指定します。
12. ユーザー属性とクレーム セクションで、編集アイコンを選択します。

    既定では、一意のユーザー識別子 (UID)、ロール、RoleSessionName、SessionDuration クレームは事前に設定されています。
13. + 新しいクレームの追加 を選択して、データベースユーザーのクレームを追加します。

    [名前] に **DbUser** と入力します。

    [Namespace] に **https://redshift.amazon.com/SAML/Attributes** と入力します。

    [出典] で、[属性] を選択します。

    出典属性で、user.userprincipalnameを選択します。次に、[保存] を選択します。
14. + 新しいクレームの追加 を選択して、AutoCreate のクレームを追加します。

    [名前] に **AutoCreate** と入力します。

    [Namespace] に **https://redshift.amazon.com/SAML/Attributes** と入力します。

    [出典] で、[属性] を選択します。

    出典属性で、trueを選択します。次に、[保存] を選択します。

    ここで、*123456789012* は AWS アカウント、*AzureSSO* はユーザーが作成した IAM ロール、*AzureADProvider* は IAM プロバイダーです。

クレーム名	値
一意のユーザー識別子 (名前 ID)	user.userprincipalname

クレーム名	値
https://aws.amazon.com/SAML/Attributes/SessionDuration	"900"
https://aws.amazon.com/SAML/Attributes/Role	arn:aws:iam::*123456789012* :role/*AzureSSO*,arn:aws:iam::*123456789012* :saml-provider/*AzureADProvider*
https://aws.amazon.com/SAML/Attributes/RoleSessionName	user.userprincipalname
https://redshift.amazon.com/SAML/Attributes/AutoCreate	"true"
https://redshift.amazon.com/SAML/Attributes/DbGroups	user.assignedroles
https://redshift.amazon.com/SAML/Attributes/DbUser	user.userprincipalname

15. アプリの登録 > ***your-application-name*** > 認証で、モバイルとデスクトップアプリケーションを追加します。URL を http://localhost/redshift と指定します。
16. 証明書の SAML 署名 セクションで ダウンロード を選択し、IAM SAMLアイデンティティプロバイダーを作成するときに使用するフェデレーションメタデータ XML ファイルをダウンロードして保存します。このファイルは、シングルサインオンアイデンティティの作成に使用されます。
17. IAM コンソールで、IAM SAML アイデンティティプロバイダーを作成します。指定したメタデータドキュメントは、Azure エンタープライズアプリケーションの設定時に保存したフェデレーションメタデータ XML ファイルです。詳細なステップついては IAM ユーザーガイドの IAM アイデンティティプロバイダー (コンソール) の作成と管理 を参照してください。
18. IAM コンソールで、SAML 2.0 フェデレーション用の IAM ロールを作成します。詳細なステップについては、「IAM ユーザーガイド」の「SAML 用のロールの作成」を参照してください。
19. IAM コンソールで SAML 2.0 フェデレーション用に作成した IAM ロールにアタッチできる IAM ポリシーを作成します。詳細な手順については、「IAM ユーザーガイド」の「IAM ポリシーの作成 (コンソール)」を参照してください。

    環境に合わせて次のポリシー (JSON 形式) を変更します。

- クラスターの AWS リージョンを *us-west-1*に置き換えます。
- AWS アカウントを *123456789012*に置き換えます。
- クラスター識別子 (またはすべてのクラスターの *) を *cluster-identifier*に置き換えます。
- データベース (またはすべてのクラスターの *) を *dev*に置き換えます。
- IAM ロールの一意の識別子を *AROAJ2UCCR6DPCEXAMPLE*に置き換えます。
- テナントまたは会社の E メールドメインを *example.com*に置き換えます。
- ユーザーに割り当てる予定のデータベースグループを *my_dbgroup*に置き換えます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "redshift:GetClusterCredentials",
            "Resource": [
                "arn:aws:redshift:us-west-1:123456789012:dbname:cluster-
identifier/dev",
                "arn:aws:redshift:us-west-1:123456789012:dbuser:cluster-identifier/
${redshift:DbUser}",
                "arn:aws:redshift:us-west-1:123456789012:cluster:cluster-
identifier"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:userid": "AROAJ2UCCR6DPCEXAMPLE:
${redshift:DbUser}@example.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "redshift:CreateClusterUser",
            "Resource": "arn:aws:redshift:us-west-1:123456789012:dbuser:cluster-
identifier/${redshift:DbUser}"
        },
        {
            "Effect": "Allow",
            "Action": "redshift:JoinGroup",
```

```
            "Resource": "arn:aws:redshift:us-west-1:123456789012:dbgroup:cluster-
identifier/my_dbgroup"
        },
        {
            "Effect": "Allow",
            "Action": [
                "redshift:DescribeClusters",
                "iam:ListRoles"
            ],
            "Resource": "*"
        }
    ]
}
```

このポリシーは、以下のアクセス許可を付与します。

- 最初のセクションでは、GetClusterCredentials API オペレーションに、指定したクラスターの一時的な認証情報を取得するアクセス許可を付与します。この例では、リソースは *cluster-identifier* で、データベースは *dev*、アカウントは *123456789012*、AWS リージョンは *us-west-1*です。${redshift:DbUser} 句を使用すると、Azure AD で指定された DbUser 値に一致するユーザーのみが接続できます。
- 条件句は、特定のユーザーだけが強制的に一時認証情報を取得します。これらのユーザーは、会社の E メールドメインの E メールアドレスで識別される IAM アカウントのロール固有 ID *AROAJ2UCCR6DPCEXAMPLE* で指定されたロールのユーザーです。固有 ID の詳細については、IAM ユーザーガイドの 固有 ID を参照してください。

  IdP (この場合は Azure AD) を使用したセットアップによって、条件句の記述方法が決まります。従業員の E メールが johndoe@example.com の場合は、まず、${redshift:DbUser} を従業員のユーザー名 johndoeに一致するスーパーフィールドに設定します。次に、この条件を機能させるため、AWS SAML RoleSessionName フィールドを従業員の E メール johndoe@example.comに一致するスーパーフィールドに設定します。このアプローチを使用する場合は、以下の点を考慮します。

  - ${redshift:DbUser} を従業員の E メールに設定した場合は、例の JSON の @example.com を削除して、RoleSessionNameに一致させます。
  - 従業員のユーザー名だけ RoleSessionId に設定した場合は、例の @example.com を削除してRoleSessionNameと一致させます。
  - 例の JSON では、${redshift:DbUser} と RoleSessionName の両方が従業員の E メールに設定されます。この例では、JSONで Amazon Redshift データベースのユーザー名

@example.com を使用して、クラスターにアクセスするためにユーザーをサインインします。

- 2 番目のセクションでは、指定したクラスターに dbuser 名を作成する権限を付与します。この例の JSON では、作成を ${redshift:DbUser} に制限しています。
- 3 番目のセクションでは、どの dbgroup にユーザーが参加できるかを指定する権限を付与します。この例の JSON では、ユーザーは指定したクラスターの my_dbgroup グループに参加できます。
- 4 番目のセクションでは、ユーザーがすべてのリソースに対して実行できるアクションへのアクセス許可を付与します。この例の JSON では、ユーザーは redshift:DescribeClusters を呼び出し、クラスターエンドポイント、AWS リージョン、ポートなどのクラスター情報を取得できます。また、ユーザーは iam:ListRoles を呼び出して、ユーザーが引き受けることができるロールを確認することもできます。

ステップ 2: Azure への認証用に JDBC または ODBC を設定する

JDBC

Microsoft Azure AD への認証用に JDBC を設定するには

- Azure AD シングルサインオンを使用して JDBC 経由でクラスターに接続するようにデータベースクライアントを構成します。

  JDBC ドライバーを使用する任意のクライアントを使用して Azure AD シングルサインオンを使用して接続、または Java などの言語を使用してスクリプトを使用して接続することもできます。インストールおよび設定の詳細については、「Amazon Redshift での JDBC ドライバーバージョン 2.x による接続の構成」を参照してください。

  たとえば、SQLWorkbench/J をクライアントとして使用できます。SQLWorkbench/J を設定する場合、データベースの URL は次の形式を使用します。

  ```
  jdbc:redshift:iam://cluster-identifier:us-west-1/dev
  ```

  SQLWorkbench/J をクライアントとして使用する場合は、次の手順を実行します。

  a. SQL Workbench/J を起動します。 接続プロファイルの選択 ページで、**AzureAuth**という プロファイルグループ を追加します。

  b. 接続プロファイルには、**Azure**と入力します。

c. [ドライバの管理] を選択し、[Amazon Redshift] を選択します。[フォルダを開く] アイコン、[ライブラリ] を選択し、適切な JDBC .jar ファイルを選択します。

d. 接続プロファイルの選択ページで、次のように接続プロファイルに情報を追加します。

- ユーザーには、マイクロソフトの Azure ユーザー名を入力します。これは、使用認証しようとしているクラスターに対するアクセス許可を持つ、シングルサインオンに使用しているマイクロソフトの Azure アカウントのユーザー名です。
- パスワードには、マイクロソフトの Azure のパスワードを入力します。
- [ドライバ] では、[Amazon Redshift (com.amazon.redshift.jdbc.Driver)] を選択します。
- [URL] には、**jdbc:redshift:iam://*your-cluster-identifier*:*your-cluster-region*/*your-database-name*** と入力します。

e. 拡張されたプロパティ を選択して、次のように接続プロパティに情報を追加します。

Azure AD シングルサインオン設定の場合は、次のように追加情報を追加します。

- plugin_name には、**com.amazon.redshift.plugin.AzureCredentialsProvider** と入力します。この値は、認証方法として Azure Single Sign-On を使用するドライバーに指定します。
- [idp_tenant] には、***your-idp-tenant*** と入力します。Microsoft Azure AD のみに使用されます。これは、Azure AD で設定されている会社のテナント名です。この値は、テナント名またはハイフン付きのテナント固有 ID のいずれかになります。
- [client_secret] には、***your-azure-redshift-application-client-secret*** と入力します。Microsoft Azure AD のみに使用されます。これは、Azure Single Sign-On 設定をセットアップするときに作成した Amazon Redshift アプリケーションのクライアントシークレットです。これは、ccom.amazon.redshift.plugin.AzureCredentialsProvider プラグインにのみ適用されます。
- [client_id] には、***your-azure-redshift-application-client-id*** と入力します。Microsoft Azure AD のみに使用されます。これは、Azure シングルサインオン設定をセットアップするときに作成した Amazon Redshift アプリケーションのクライアント ID (ハイフン付き) です。

MFA 設置を使用した Azure AD シングルサインオンの場合は、次のように接続プロパティに情報を追加します。

- [plugin_name] には、**com.amazon.redshift.plugin.BrowserAzureCredentialsProvider** と入力します。この値は、認証方法として MFA と Azure シングルサインオンを使用するようドライバに指定します。
- [idp_tenant] には、***your-idp-tenant*** と入力します。Microsoft Azure AD のみに使用されます。これは、Azure AD で設定されている会社のテナント名です。この値は、テナント名またはハイフン付きのテナント固有 ID のいずれかになります。
- [client_id] には、***your-azure-redshift-application-client-id*** と入力します。Microsoft Azure AD のみに使用されます。これは、MFA 設定で Azure AD シングルサインオン設定をセットアップするときに作成した Amazon Redshift アプリケーションのクライアント ID (ハイフン付き) です。
- [listen_port] には、***your-listen-port***と入力します。これは、ローカルサーバーがリッスンしているポートです。デフォルト値は 7890 です。
- [idp_response_timeout] には、***the-number-of-seconds*** と入力します。IdP サーバーが応答を返すときにタイムアウトするまで待機する秒数です。最小秒数は 10 である必要があります。接続を確立するときにこのしきい値よりも時間がかかると、接続は中止されます。

ODBC

Microsoft Azure AD への認証用に ODBC を設定するには

- Azure AD シングルサインオンを使用して ODBC 経由でクラスターに接続するようにデータベースクライアントを構成します。

  Amazon Redshift には Linux、Windows、macOS オペレーティングシステム用の ODBC ドライバが用意されています。ODBC ドライバをインストールする前に、SQL クライアントツールが 32 ビットであるか 64 ビットであるかを調べます。SQL クライアントツールの要件と一致する ODBC ドライバをインストールします。

  Windows の場合、Amazon Redshift ODBC ドライバ DSN セットアップ ページの [接続設定] で、次の情報を入力します。

- [データソース名] には、***your-DSN*** と入力します。ODBC プロファイル名として使用するデータソース名を指定します。
- Azure AD シングルサインオン設定の Auth type には、**Identity Provider: Azure AD** を選択します。これは、ODBC ドライバが Azure シングルサインオンを使用した認証に使用する認証方法です。
- MFA設定の Azure AD シングルサインオンの Auth type には、**Identity Provider: Browser Azure AD** を選択します。これは、ODBC ドライバが MFA と Azure シングルサインオンを使用した認証に使用する認証方法です。
- [クラスター ID] には、***your-cluster-identifier*** と入力します。
- [リージョン] には、***your-cluster-region*** と入力します。
- [データベース] には、***your-database-name*** と入力します。
- [ユーザー] には、***your-azure-username*** と入力します。これは、使用認証しようとしているクラスターに対するアクセス許可を持つ、シングルサインオンに使用しているマイクロソフトの Azure アカウントのユーザー名です。これは、Auth Type が アイデンティティプロバイダー: Azure AD のみに使用します。
- [パスワード] には、***your-azure-password*** と入力します。これは、Auth Type が アイデンティティプロバイダー: Azure ADのみに使用します。
- IdP Tenantには、***your-idp-tenant***と入力します。これは、IdP (Azure) で設定されている会社のテナント名です。この値は、テナント名またはハイフン付きのテナント固有 ID のいずれかになります。
- Azure Client Secretには、***your-azure-redshift-application-client-secret***と入力します。これは、Azure シングルサインオン設定をセットアップするときに作成した Amazon Redshift アプリケーションのクライアントの秘密です。
- Azure Client IDには、***your-azure-redshift-application-client-id***と入力します。これは、Azure シングルサインオン設定をセットアップするときに作成した Amazon Redshift アプリケーションのクライアント ID (ハイフン付き) です。
- [Listen Port] には、***your-listen-port*** と入力します。これは、ローカルサーバーがリッスンしているデフォルトのリッスンポートです。デフォルト値は 7890 です。これは、ブラウザ Azure AD プラグインのみに適用されます。
- [応答タイムアウト] には、***the-number-of-seconds*** と入力します。IdP サーバーが応答を返すときにタイムアウトするまで待機する秒数です。最小秒数は 10 である必要があります。接続を確立するときにこのしきい値よりも時間がかかると、接続は中止されます。これは、ブラウザ Azure AD プラグインのみに適用されます。

macOS および Linux では、次のように odbc.ini ファイルを編集します。

> **Note**
>
> すべてのエントリでは、大文字と小文字は区別されません。

- [clusterid] には、***your-cluster-identifier*** と入力します。これは、作成された Amazon Redshift クラスターの名前です。
- [リージョン] には、***your-cluster-region*** と入力します。これは、作成された Amazon Redshift クラスターの AWS リージョンです。
- [データベース] には、***your-database-name*** と入力します。これは、Amazon Redshift クラスター上でアクセスしようとしているデータベースの名前です。
- [ロケール] には、**en-us** と入力します。これは、エラーメッセージが表示される言語です。
- [iam] には、**1** と入力します。この値は、IAM 認証情報を使用して認証するためにドライバに指定します。
- Azure AD シングルサインオン設定の plugin_name には、**AzureAD** と入力します。これにより、認証方法として Azure シングルサインオンを使用するようドライバに指定します。
- MFA 設定の Azure AD シングルサインオンの plugin_name には、**BrowserAzureAD** と入力します。これにより、認証方法として MFA と Azure Single Sign-On を使用するようドライバに指定します。
- [uid] には、***your-azure-username*** と入力します。これは、認証しようとしているクラスターに対するアクセス許可を持つ、シングルサインオンに使用しているMicrosoft Azure アカウントのユーザー名です。これは、plugin_name が AzureAD である場合にのみ使用します。
- [pwd] には、***your-azure-password*** と入力します。これは、plugin_name が AzureADである場合のみに使用します。
- [idp_tenant] には、***your-idp-tenant*** と入力します。これは、IdP (Azure) で設定されている会社のテナント名です。この値は、テナント名またはハイフン付きのテナント固有 ID のいずれかになります。

- [client_secret] には、***your-azure-redshift-application-client-secret*** と入力します。これは、Azure シングルサインオン設定をセットアップするときに作成した Amazon Redshift アプリケーションのクライアントの秘密です。
- [client_id] には、***your-azure-redshift-application-client-id*** と入力します。これは、Azure シングルサインオン設定をセットアップするときに作成した Amazon Redshift アプリケーションのクライアント ID (ハイフン付き) です。
- [listen_port] には、***your-listen-port***と入力します。これは、ローカルサーバーがリッスンしているポートです。デフォルト値は 7890 です。これは、ブラウザ Azure AD プラグインに適用されます。
- [idp_response_timeout] には、***the-number-of-seconds*** と入力します。これは、Azure からの応答を待機する秒単位の指定された期間です。これは、ブラウザ Azure AD プラグインに適用されます。

macOS と Linux では、プロファイル設定を編集して、次のエクスポートも追加できます。

```
export ODBCINI=/opt/amazon/redshift/Setup/odbc.ini
```

```
export ODBCINSTINI=/opt/amazon/redshift/Setup/odbcinst.ini
```

トラブルシューティング

ブラウザ Azure AD プラグインに関する問題のトラブルシューティングを行うには、以下を検討します。

- ブラウザ Azure AD プラグインを使用するには、リクエストで指定される返信 URL を、アプリケーションで構成された返信 URL と一致するように設定する必要があります。Microsoft Azure ポータルの SAML を使用したシングルサインオンのセットアップ ページに移動します。次に、返信 URL が http://localhost/redshift/ に設定されていることを確認します。
- IdP テナントエラーが発生した場合は、IdP テナント 名が、Microsoft Azure で Active Directory をセットアップするために最初に使用したドメイン名と一致することを確認します。

  Windows の場合、接続の設定 の セクション Amazon Redshift ODBC DSN セットアップページに移動します。次に、IdP (Azure) で構成されている会社のテナント名が、Microsoft Azure で Active Directory をセットアップするために最初に使用したドメイン名と一致することを確認します。

macOS と Linux では、odbc.ini ファイルを見つけます。次に、IdP (Azure) で構成されている会社のテナント名が、Microsoft Azure で Active Directory をセットアップするために最初に使用したドメイン名と一致することを確認します。

- リクエストで指定された返信 URL がアプリケーションに設定されている返信 URL と一致しないというエラーが表示された場合は、リダイレクト URI が返信 URL と同じであることを確認します。

  Microsoft Azure ポータルでアプリケーションの アプリ登録 ページに移動します。次に、リダイレクト URI が返信 URL と一致することを確認します。

- 予期しない応答: 不正なエラーが表示された場合は、モバイルとデスクトップアプリケーション の構成が完了したことを確認します。

  Microsoft Azure ポータルでアプリケーションの アプリ登録 ページに移動します。次に、認証 に移動し、リダイレクト URI として http://localhost/redshift/ を使用するように モバイルとデスクトップアプリケーション が構成されていることを確認します。

Ping Identity

Amazon Redshift クラスターにアクセスするには、ID プロバイダー (IdP) として Ping Identity を使用できます。このチュートリアルでは、Ping Identity を ID プロバイダー (IdP) として使用して、Amazon Redshift クラスターにアクセスする方法を説明します。

ステップ 1: Ping Identity と AWS アカウントを相互に信頼するように設定する

以下の手順では、PingOne ポータルを使用して信頼関係を設定する方法を説明します。

相互に信頼するように Ping ID と AWS アカウントを設定するには

1. Ping Identity ユーザーが接続するための既存の Amazon Redshift クラスターを作成または使用します。接続を設定するには、クラスター識別子など、このクラスターの特定のプロパティが必要です。詳細については、「クラスターの作成」を参照してください。
2. Amazon Redshift を新しい SAML アプリケーションとして PingOne ポータルに追加します。詳細な手順については、Ping Identity のドキュメントを参照してください。

   1. マイアプリケーションに移動します。
   2. アプリケーションの追加で、新しい SAML アプリケーションを選択します。
   3. アプリケーション名には **`Amazon Redshift`**と入力します。

4. プロトコルバージョンには、SAML v2.0を選択します。
5. カテゴリには、*your-application-category*を選択します。
6. Assertion Consumer Service (ACS)には、*your-redshift-local-host-url*と入力します。これは、SAML アサーションがリダイレクトするローカルホストとポートです。
7. エンティティ IDには、`urn:amazon:webservices`と入力します。
8. [署名] には、[アサーションに署名] を選択します。
9. SSO 属性マッピング セクションで、次の表に示すように要求を作成します。

アプリケーション属性	リテラル値の ID ブリッジ属性
https://aws.amazon.com/SAML/Attributes/Role	arn:aws:iam::*123456789012* :role/*Ping*,arn:aws:iam::*123456789012* :saml-provider/*PingProvider*
https://aws.amazon.com/SAML/Attributes/RoleSessionName	email
https://redshift.amazon.com/SAML/Attributes/AutoCreate	"true"
https://redshift.amazon.com/SAML/Attributes/DbUser	email
https://redshift.amazon.com/SAML/Attributes/DbGroups	「DbGroups」属性のグループには、@directory プレフィックスが含まれています。これを削除するには、ID ブリッジで、memberOfと入力します。関数で、ExtractByRegularExpressionを選択します。式に (.*)[\@](?:.*)と入力します。

3. グループアクセスで、必要に応じて次のグループアクセスを設定します。

- https://aws.amazon.com/SAML/Attributes/Role
- https://aws.amazon.com/SAML/Attributes/RoleSessionName
- https://redshift.amazon.com/SAML/Attributes/AutoCreate
- https://redshift.amazon.com/SAML/Attributes/DbUser

4. セットアップを確認し、必要に応じて変更を加えます。
5. ブラウザ SAML プラグインのログイン URL として Single Sign-On (SSO) 用 URL の初期化 を使用します。
6. IAM コンソールで、IAM SAML アイデンティティプロバイダーを作成します。指定したメタデータドキュメントは、Ping Identity のセットアップ時に保存したフェデレーションメタデータ XML ファイルです。詳細な手順については、「IAM ユーザーガイド」の「IAM ID プロバイダーを作成および管理する (コンソール)」を参照してください。
7. IAM コンソールで、SAML 2.0 フェデレーション用の IAM ロールを作成します。詳細なステップについては、「IAM ユーザーガイド」の「SAML 用のロールの作成」を参照してください。
8. IAM コンソールで SAML 2.0 フェデレーション用に作成した IAM ロールにアタッチできる IAM ポリシーを作成します。詳細な手順については、「IAM ユーザーガイド」の「IAM ポリシーの作成 (コンソール)」を参照してください。Azure AD の例については、「JDBC または ODBC シングルサインオン認証の設定」を参照してください。

ステップ 2: Ping Identity への認証用に JDBC または ODBC を設定する

JDBC

Ping Identity への認証用に JDBC を設定するには

- Ping Identity シングルサインオンを使用して JDBC 経由でクラスターに接続するようにデータベースクライアントを設定します。

  JDBC ドライバーを使用する任意のクライアントを使用して Ping Identity シングルサインオンを使用して接続、または Java などの言語を使用してスクリプトを使用して接続することもできます。インストールおよび設定の詳細については、「Amazon Redshift での JDBC ドライバーバージョン 2.x による接続の構成」を参照してください。

  たとえば、SQLWorkbench/J をクライアントとして使用できます。SQLWorkbench/J を設定する場合、データベースの URL は次の形式を使用します。

  ```
  jdbc:redshift:iam://cluster-identifier:us-west-1/dev
  ```

  SQLWorkbench/J をクライアントとして使用する場合は、次の手順を実行します。

  a. [Select Connection Profile] ページで、[Profile Group] を追加します (例: **Ping**)。

b. [接続プロファイル] に、***your-connection-profile-name*** と入力します (例: **Ping**)。

c. [Manage Drivers (ドライバーの管理)] を選択し、[Amazon Redshift] を選択します。[フォルダを開く] アイコン、[ライブラリ] を選択し、適切な JDBC .jar ファイルを選択します。

d. 接続プロファイルの選択ページで、次のように接続プロファイルに情報を追加します。

- ユーザーに、PingOne ユーザー名を入力します。これは、使用認証しようとしているクラスターに対するアクセス許可を持つ、シングルサインオンに使用している PingOne アカウントのユーザー名です。
- パスワードに、PingOne のパスワードを入力します。
- [ドライバ] では、[Amazon Redshift (com.amazon.redshift.jdbc.Driver)] を選択します。
- [URL] には、**jdbc:redshift:iam://*your-cluster-identifier*:*your-cluster-region*/*your-database-name*** と入力します。

e. [拡張プロパティ] を選択し、次のいずれかの操作を行います。

- [login_url] には、***your-ping-sso-login-url*** と入力します。この値は、ログインするための認証としてシングルサインオンを使用する URL を指定します。
- Ping Identity の場合は、plugin_nameに **com.amazon.redshift.plugin.PingCredentialsProvider**と入力します。この値は、認証方法として Ping Identity シングルサインオンを使用するドライバーに指定します。
- シングルサインオンを使用した Ping Identity の場合は、plugin_nameに **com.amazon.redshift.plugin.BrowserSamlCredentialsProvider** と入力します。この値は、シングルサインオンを使用した Ping Identity PingOneを認証方法として使用するようにドライバーに指定します。

ODBC

Ping Identity への認証用に ODBC を設定するには

- Ping Identity PingOne シングルサインオンを使用して ODBC 経由でクラスターに接続するように、データベースクライアントを設定します。

Amazon Redshift には Linux、Windows、macOS オペレーティングシステム用の ODBC ドライバが用意されています。ODBC ドライバをインストールする前に、SQL クライアントツールが 32 ビットであるか 64 ビットであるかを調べます。SQL クライアントツールの要件と一致する ODBC ドライバをインストールします。

Windows の場合、Amazon Redshift ODBC ドライバ DSN セットアップ ページの [接続設定] で、次の情報を入力します。

- [データソース名] には、***your-DSN*** と入力します。ODBC プロファイル名として使用するデータソース名を指定します。
- Auth typeで、以下のいずれかを実行します。
  - Ping Identity の設定で、アイデンティティプロバイダー: Ping Federateを選択します。これは、ODBC ドライバが Ping Identity シングルサインオンを使用して認証するために使用する認証方法です。
  - MFA を使用した Ping Identity PingOne シングルサインオンの設定で、[Identity Provider: Browser SAML] (ID プロバイダー: ブラウザ SAML) を選択します。これは、ODBC ドライバーがシングルサインオンを使用した Ping Identity で認証するために使用する認証方法です。
- [クラスター ID] には、***your-cluster-identifier*** と入力します。
- [リージョン] には、***your-cluster-region*** と入力します。
- [データベース] には、***your-database-name*** と入力します。
- [ユーザー] には、***your-ping-username*** と入力します。これは、使用認証しようとしているクラスターに対するアクセス許可を持つ、シングルサインオンに使用しているマイクロソフトの PingOne アカウントのユーザー名です。これは、Auth type が アイデンティティプロバイダー: PingFederate である場合にのみ使用します。
- [パスワード] には、***your-ping-password*** と入力します。これは、Auth type が アイデンティティプロバイダー: PingFederateである場合にのみ使用します。
- [Listen Port] には、***your-listen-port*** と入力します。これは、ローカルサーバーがリッスンしているポートです。デフォルト値は 7890 です。これは、ブラウザの SAML プラグインにのみ適用されます。
- [応答タイムアウト] には、***the-number-of-seconds*** と入力します。IdP サーバーが応答を返すときにタイムアウトするまで待機する秒数です。最小秒数は 10 である必要があります。接続を確立するときにこのしきい値よりも時間がかかると、接続は中止されます。これは、ブラウザの SAML プラグインにのみ適用されます。

- ログイン URLには、***your-login-url***と入力します。これは、ブラウザの SAML プラグインにのみ適用されます。

macOS および Linux では、次のように `odbc.ini` ファイルを編集します。

> **Note**
>
> すべてのエントリでは、大文字と小文字は区別されません。

- [clusterid] には、***your-cluster-identifier*** と入力します。これは、作成された Amazon Redshift クラスターの名前です。
- [リージョン] には、***your-cluster-region*** と入力します。これは、作成された Amazon Redshift クラスターの AWS リージョンです。
- [データベース] には、***your-database-name*** と入力します。これは、Amazon Redshift クラスター上でアクセスしようとしているデータベースの名前です。
- [ロケール] には、**en-us** と入力します。これは、エラーメッセージが表示される言語です。
- [iam] には、**1** と入力します。この値は、IAM 認証情報を使用して認証するためにドライバに指定します。
- [plugin_name] には、次のいずれかの操作を行います。
  - Ping Identity 設定の場合は、**BrowserSAML**を入力します。これは、ODBC ドライバが Ping Identity に対して認証するために使用する認証方法です。
  - シングルサインオン設定を使用した Ping Identity の場合は、**Ping** と入力します。これは、ODBC ドライバーがシングルサインオンを使用した Ping Identity で認証するために使用する認証方法です。
- [uid] には、***your-ping-username*** と入力します。これは、認証しようとしているクラスターに対するアクセス許可を持つ、シングルサインオンに使用しているMicrosoft Azure アカウントのユーザー名です。これは、plugin_name が Ping である場合にのみ使用します。
- [pwd] には、***your-ping-password*** と入力します。これは、plugin_name が Pingである場合にのみ使用します。
- [login_url] には、***your-login-url*** と入力します。これは、SAML 応答を返すシングルサインオンの開始 URL です。これは、ブラウザの SAML プラグインにのみ適用されます。

- [idp_response_timeout] には、***the-number-of-seconds*** と入力します。これは、PingOne Identity からの応答を待機する秒単位の指定された期間です。これは、ブラウザの SAML プラグインにのみ適用されます。
- [listen_port] には、***your-listen-port***と入力します。これは、ローカルサーバーがリッスンしているポートです。デフォルト値は 7890 です。これは、ブラウザの SAML プラグインにのみ適用されます。

macOS および Linux では、プロファイル設定を編集して、次のエクスポートも追加できます。

```
export ODBCINI=/opt/amazon/redshift/Setup/odbc.ini
```

```
export ODBCINSTINI=/opt/amazon/redshift/Setup/odbcinst.ini
```

Okta

Okta を ID プロバイダー (IdP) として使用して、Amazon Redshift クラスターにアクセスできます。このチュートリアルでは、Okta を ID プロバイダー (IdP) として使用して、Amazon Redshift クラスターにアクセスする方法を説明します。

ステップ 1: Okta と AWS アカウントを相互に信頼するように設定する

以下の手順では、信頼関係を設定する方法を説明します。

相互に信頼するように Okta と AWS アカウントを設定するには

1. Okta ユーザーが接続するための既存の Amazon Redshift クラスターを作成または使用します。接続を設定するには、クラスター識別子など、このクラスターの特定のプロパティが必要です。詳細については、「クラスターの作成」を参照してください。
2. Okta ポータルに新しいアプリケーションとして Amazon Redshift を追加します。詳細な手順については、Okta のドキュメントを参照してください。
   - アプリケーションの追加を選択します。
   - アプリケーションの追加で、新しいアプリの作成を選択します。
   - 新規追加のアプリケーション統合の作成 ページの プラットフォームで、ウェブを選択します。

- サインオンメソッドで、SAML v2.0を選択します。
- 一般設定 ページで、アプリ名に ***your-redshift-saml-sso-name***と入力します。これは、アプリケーションの名前です。
- SAML 設定 ページの シングルサインオンの URLに ***your-redshift-local-host-url***と入力します。これは、SAML アサーションがリダイレクトするローカルホストとポート (例: `http://localhost:7890/redshift/`) です。

3. シングルサインオン URL 値を 受取人URL と 送信先 URLとして使用します。
4. [署名] には、[アサーションに署名] を選択します。
5. 対象 URI (SP エンティティ ID)に、次の表に示すように、クレームに対して **`urn:amazon:webservices`** を入力します。
6. [高度な設定] セクションの [SAML 発行者 ID] に、[セットアップ手順を表示] セクションに表示されている ***your-Identity-Provider-Issuer-ID*** を入力します。
7. 属性ステートメント セクションで、次の表に示すようにクレームを作成します。

クレーム名	値
https://aws.amazon.com/SAML/Attributes/Role	arn:aws:iam::*123456789012* :role/*Okta*,arn:aws:iam::*123456789012* :saml-provider/*Okta*
https://aws.amazon.com/SAML/Attributes/RoleSessionName	user.email
https://redshift.amazon.com/SAML/Attributes/AutoCreate	"true"
https://redshift.amazon.com/SAML/Attributes/DbUser	user.email

8. App Embed Link セクションで、ブラウザ SAML プラグインのログイン URL として使用できる URL を見つけます。
9. IAM コンソールで、IAM SAML アイデンティティプロバイダーを作成します。指定したメタデータドキュメントは、Okta の設定時に保存したフェデレーションメタデータ XML ファイルです。詳細な手順については、「IAM ユーザーガイド」の「IAM ID プロバイダーを作成および管理する (コンソール)」を参照してください。

10. IAM コンソールで、SAML 2.0 フェデレーション用の IAM ロールを作成します。詳細なステップについては、「IAM ユーザーガイド」の「SAML 用のロールの作成」を参照してください。
11. IAM コンソールで SAML 2.0 フェデレーション用に作成した IAM ロールにアタッチできる IAM ポリシーを作成します。詳細な手順については、「IAM ユーザーガイド」の「IAM ポリシーの作成 (コンソール)」を参照してください。Azure AD の例については、「JDBC または ODBC シングルサインオン認証の設定」を参照してください。

ステップ 2: Okta への認証用に JDBC または ODBC を設定する

JDBC

Okta への認証用に JDBC を設定するには

- Okta シングルサインオンを使用して JDBC 経由でクラスターに接続するようにデータベースクライアントを構成します。

  JDBC ドライバーを使用する任意のクライアントを使用して Okta シングルサインオンを使用して接続、または Java などの言語を使用してスクリプトを使用して接続することもできます。インストールおよび設定の詳細については、「Amazon Redshift での JDBC ドライバーバージョン 2.x による接続の構成」を参照してください。

  たとえば、SQLWorkbench/J をクライアントとして使用できます。SQLWorkbench/J を設定する場合、データベースの URL は次の形式を使用します。

  ```
  jdbc:redshift:iam://cluster-identifier:us-west-1/dev
  ```

  SQLWorkbench/J をクライアントとして使用する場合は、次の手順を実行します。

  a. [Select Connection Profile] ページで、[Profile Group] を追加します (例: **Okta**)。
  b. [接続プロファイル] に、***your-connection-profile-name*** と入力します (例: **Okta**)。
  c. [Manage Drivers (ドライバーの管理)] を選択し、[Amazon Redshift] を選択します。[フォルダを開く] アイコン、[ライブラリ] を選択し、適切な JDBC .jar ファイルを選択します。
  d. 接続プロファイルの選択ページで、次のように接続プロファイルに情報を追加します。

- ユーザーには、Okta ユーザー名を入力します。これは、使用認証しようとしているクラスターに対するアクセス許可を持つ、シングルサインオンに使用している Okta アカウントのユーザー名です。
- パスワードには、Okta パスワードを入力します。
- [ドライバ] では、[Amazon Redshift (com.amazon.redshift.jdbc.Driver)] を選択します。
- [URL] には、**jdbc:redshift:iam://*your-cluster-identifier*:*your-cluster-region*/*your-database-name*** と入力します。

e. [拡張プロパティ] を選択し、次のいずれかの操作を行います。

- [login_url] には、***your-okta-sso-login-url*** と入力します。この値は、Okta に ログインするための認証としてシングルサインオンを使用する URL を指定します。
- Okta シングルサインオンの場合は、plugin_name に **com.amazon.redshift.plugin.OktaCredentialsProvider** と入力します。この値は、認証方法として Okta シングルサインオンを使用するドライバーに指定します。
- MFA を使用した Okta シングルサインオンの場合は、plugin_nameに **com.amazon.redshift.plugin.BrowserSamlCredentialsProvider**と入力します。これにより、認証方法として MFA での Okta シングルサインオンを使用するようドライバーに指定します。

ODBC

Okta への認証用に ODBC を設定するには

- Okta シングルサインオンを使用して ODBC 経由でクラスターに接続するようにデータベースクライアントを構成します。

  Amazon Redshift には Linux、Windows、macOS オペレーティングシステム用の ODBC ドライバが用意されています。ODBC ドライバをインストールする前に、SQL クライアントツールが 32 ビットであるか 64 ビットであるかを調べます。SQL クライアントツールの要件と一致する ODBC ドライバをインストールします。

  Windows の場合、Amazon Redshift ODBC ドライバ DSN セットアップ ページの [接続設定] で、次の情報を入力します。

- [データソース名] には、***your-DSN*** と入力します。ODBC プロファイル名として使用するデータソース名を指定します。
- Auth type で、以下のいずれかを実行します。
  - Okta シングルサインオン設定の場合は、**Identity Provider: Okta** を選択します。これは、ODBC ドライバーが ODBC シングルサインオンを使用した認証に使用する認証方法です。
  - MFA 設定を使用した Okta シングルサインオンの場合は、**Identity Provider: Browser SAML** を選択します。これは、ODBC ドライバーが MFA での Okta シングルサインオンを使用した認証に使用する認証方法です。
- クラスター ID には、***your-cluster-identifier*** と入力します。
- [リージョン] には、***your-cluster-region*** と入力します。
- [データベース] には、***your-database-name*** と入力します。
- [ユーザー] には、***your-okta-username*** と入力します。これは、使用認証しようとしているクラスターに対するアクセス許可を持つ、シングルサインオンに使用しているマイクロソフトの Okta アカウントのユーザー名です。[Auth type] が Identity Provider: Okta である場合にのみ使用します。
- [パスワード] には、***your-okta-password*** と入力します。[Auth type] が Identity Provider: Okta である場合にのみ使用します。

macOS と Linux では、次のように `odbc.ini` ファイルを編集します。

> **Note**
>
> すべてのエントリでは、大文字と小文字は区別されません。

- [clusterid] には、***your-cluster-identifier*** と入力します。これは、作成された Amazon Redshift クラスターの名前です。
- [リージョン] には、***your-cluster-region*** と入力します。これは、作成された Amazon Redshift クラスターの AWS リージョンです。
- [データベース] には、***your-database-name*** と入力します。これは、Amazon Redshift クラスター上でアクセスしようとしているデータベースの名前です。

- [ロケール] には、**en-us** と入力します。これは、エラーメッセージが表示される言語です。
- [iam] には、**1** と入力します。この値は、IAM 認証情報を使用して認証するためにドライバに指定します。
- plugin_name には、次のいずれかの操作を行います。
  - MFA 設定を使用した Okta シングルサインオンの場合は、**BrowserSAML** と入力します。これは、ODBC ドライバーが MFA での Azure シングルサインオンを使用した認証に使用する認証方法です。
  - Okta シングルサインオン設定の場合は、**Okta** と入力します。これは、ODBC ドライバーが ODBC シングルサインオンを使用した認証に使用する認証方法です。
- [uid] には、***your-okta-username*** と入力します。これは、認証しようとしているクラスターに対するアクセス許可を持つ、シングルサインオンに使用している Okta アカウントのユーザー名です。これは、[plugin_name] が Oktaである場合にのみ使用します。
- [pwd] には、***your-okta-password*** と入力します。これは、[plugin_name] が Oktaである場合にのみ使用します。
- [login_url] には、***your-login-url*** と入力します。これは、SAML 応答を返すシングルサインオンの開始 URL です。これは、ブラウザの SAML プラグインにのみ適用されます。
- [idp_response_timeout] には、***the-number-of-seconds*** と入力します。これは、PingOne からの応答を待機する秒単位の指定された時間です。これは、ブラウザの SAML プラグインにのみ適用されます。
- [listen_port] には、***your-listen-port***と入力します。これは、ローカルサーバーがリッスンしているポートです。デフォルト値は 7890 です。これは、ブラウザの SAML プラグインにのみ適用されます。

macOS および Linux では、プロファイル設定を編集して、次のエクスポートも追加できます。

```
export ODBCINI=/opt/amazon/redshift/Setup/odbc.ini
```

```
export ODBCINSTINI=/opt/amazon/redshift/Setup/odbcinst.ini
```

ユーザーに代わって Amazon Redshift が他の AWS のサービスにアクセスすることを許可する

一部の Amazon Redshift 機能では、ユーザーに代わって Amazon Redshift が他の AWS のサービスにアクセスする必要があります。たとえば、COPY と UNLOAD コマンドでは、Amazon S3 バケットを使用してデータを Amazon Redshift クラスターにロードまたはアンロードできます。CREATE EXTERNAL FUNCTION コマンドは、スカラーの Lambda ユーザー定義関数 (UDF) を使用して AWS Lambda 関数を呼び出すことができます。Amazon Redshift Spectrum では、Amazon Athena または AWS Glue のデータカタログを使用できます。Amazon Redshift クラスターがユーザーに代わってタスクを実行するには、クラスターにセキュリティ認証情報を指定します。セキュリティ認証情報を指定する望ましい方法は、AWS Identity and Access Management (IAM) ロールを指定することです。COPY と UNLOAD では、一時的な認証情報を指定できます。

AWS マネジメントコンソール の外部で AWS を操作するには、プログラマチックアクセス権が必要です。プログラムによるアクセス権を付与する方法は、AWS にアクセスしているユーザーのタイプによって異なります。

ユーザーにプログラムによるアクセス権を付与するには、以下のいずれかのオプションを選択します。

プログラムによるアクセス権を必要とするユーザー	目的	方法
IAM	(推奨) 一時的な認証情報としてコンソール認証情報を使用して、AWS CLI、AWS SDK、または AWS API へのプログラマチックリクエストに署名します。	使用するインターフェイスの指示に従ってください。 • AWS CLI については、「AWS Command Line Interface ユーザーガイド」の「AWS ローカル開発用のログイン」を参照してください。 • AWS SDK については、「AWS SDK およびツールリファレンスガイド」の「AWS ローカル開発用の口

プログラムによるアクセス権を必要とするユーザー	目的	方法
		グイン」を参照してください。
ワークフォースアイデンティティ (IAM アイデンティティセンターで管理されているユーザー)	一時的な認証情報を使用して、AWS CLI、AWS SDK、または AWS API へのプログラムによるリクエストに署名します。	使用するインターフェイスの指示に従ってください。 • AWS CLI については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM アイデンティティセンター を使用するための AWS CLI の設定」を参照してください。 • AWS SDK、ツール、および AWS API については、AWS SDK とツールリファレンスガイドの「IAM アイデンティティセンター認証」を参照してください。
IAM	一時的な認証情報を使用して、AWS CLI、AWS SDK、または AWS API へのプログラムによるリクエストに署名します。	IAM ユーザーガイドの「AWS リソースでの一時的な認証情報の使用」の指示に従ってください。

プログラムによるアクセス権を必要とするユーザー	目的	方法
IAM	(非推奨) 長期的な認証情報を使用して、AWS CLI、AWS SDK、または AWS API へのプログラムによるリクエストに署名します。	使用するインターフェイスの指示に従ってください。 • AWS CLI については、「AWS Command Line Interface ユーザーガイド」の「IAM ユーザー認証情報を使用した認証」を参照してください。 • AWS SDK とツールについては、「AWS SDK とツールリファレンスガイド」の「長期認証情報を使用して認証する」を参照してください。 • AWS API については、「IAM ユーザーガイド」の「IAM ユーザーのアクセスキーの管理」を参照してください。

次に、他の AWS サービスにアクセスするための適切なアクセス許可を持つ IAM ロールを作成する方法を説明します。Amazon Redshift コマンドの実行時に、ロールをクラスターに関連付けてから、そのロールの Amazon リソースネーム (ARN) を指定する必要があります。詳細については、「IAM ロールを使用して COPY、UNLOAD、CREATE EXTERNAL FUNCTION、CREATE EXTERNAL SCHEMA オペレーションを承認する」を参照してください。

さらに、スーパーユーザーは、COPY および UNLOAD 操作のロールへのアクセスを提供するために、特定のユーザーおよびグループに AssumEROLE 権限を付与できます。詳細については、Amazon Redshift データベースデベロッパーガイドから GRANT を参照してください。

Amazon Redshift クラスターが AWS サービスにアクセスすることを許可する IAM ロールの作成

Amazon Redshift クラスターが AWS サービスにアクセスすることを許可する IAM ロールの作成

ユーザーに代わって Amazon Redshift クラスターが他の AWS サービスと通信することを許可する IAM ロールを作成するには、次の手順を実行します。このセクションで使用する値は例であり、必要に応じて値を選択できます。

Amazon Redshift が AWS サービスにアクセスすることを許可する IAM ロールを作成するには

1. IAM コンソールを開きます。
2. ナビゲーションペインで [ロール] を選択します。
3. ロールの作成を選択します。
4. AWS サービス、Redshiftの順にクリックします。
5. ユースケースの選択で、Redshift - カスタマイズ可能 、次へ : 許可の順に選択します。アクセス権限ポリシーをアタッチする ページが表示されます。
6. COPY を使用して Amazon S3 にアクセスするには、例として、 **AmazonS3ReadOnlyAccess** と入力して追加します。COPY または UNLOAD を使用して Amazon S3 にアクセスする場合は、適切なバケットとプレフィックスへのアクセスを制限する管理ポリシーを作成することをお勧めします。読み取り操作と書き込み操作の両方で、最小限の権限を適用し、Amazon Redshift に必要な Amazon S3 バケットとキープレフィックスのみに制限することをお勧めします。

   CREATE EXTERNAL FUNCTION コマンドの Lambda 関数を呼び出すためのアクセスには、**AWSLambdaRole**を追加します。

   Redshift Spectrum の場合は、Amazon S3 アクセスに加えて、 **AWSGlueConsoleFullAccess** または **AmazonAthenaFullAccess**を追加します。

   次へ: タグを選択します。
7. タグの追加 ページが表示されます。必要に応じてタグを追加できます。次へ: レビューを選択します。
8. ロール名で、 **RedshiftCopyUnload**などのロール名を入力します。ロールの作成を選択します。
9. 新しいロールは、そのロールを使用するクラスターのすべてのユーザーが利用できます。特定のクラスターの特定のユーザーのみ、または特定のリージョンのクラスターのみにアクセスを制限

するには、ロールの信頼関係を編集します。詳細については、「IAM ロールへのアクセスの制限」を参照してください。

10. ロールとクラスターを関連付けます。クラスターの作成時に IAM ロールをクラスターに関連付けるか、既存クラスターにロールを追加できます。詳細については、「IAM ロールをクラスターに関連付ける」を参照してください。

> **Note**
>
> 特定のデータへのアクセスを制限するには、必要な最小限の権限を付与する IAM ロールを使用します。

IAM ロールへのアクセスの制限

デフォルトでは、Amazon Redshift クラスターに使用できる IAM ロールはそのクラスターですべてのユーザーが利用できます。特定のクラスターの特定の Amazon Redshift データベースユーザーまたは特定のリージョンに IAM ロールを制限することもできます。

IAM ロールの使用を特定のデータベースユーザーのみに許可するには、次の手順に従います。

IAM ロールへのアクセス権を持つ特定のデータベースユーザーを特定する

1. Amazon Redshift クラスターでデータベースユーザーの Amazon リソースネーム (ARN) を特定します。データベースユーザーの ARN は形式: `arn:aws:redshift:`*`region`*`:`*`account-id`*`:dbuser:`*`cluster-name`*`/`*`user-name`*になります。

   Amazon Redshift Serverless の場合は、次の ARN 形式を使用します。`arn:aws:redshift:`*`region`*`:`*`account-id`*`:dbuser:serverless-`*`account-id`*`-`*`workgroup-id`*`/`*`user-name`*

2. IAM コンソールを開きます。
3. ナビゲーションペインで [ロール] を選択します。
4. 特定の Amazon Redshift データベースユーザーに制限する IAM ロールを選択します。
5. 信頼関係 タブを選択し、 信頼関係の編集を選択します。ユーザーに代わって Amazon Redshift が他の AWS サービスにアクセスすることを許可する新しい IAM ロールには、次のような信頼関係があります。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "redshift.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

6. 指定した値に sts:AssumeRole フィールドを制限する信頼関係のセクションの sts:ExternalId アクションに条件を追加します。ロールへのアクセスを許可する各データベースユーザーの ARN を含めます。外部 ID には、任意の一意の文字列を指定できます。

   たとえば、次の信頼関係では、 user1 リージョンの user2 クラスターで my-cluster と us-west-2 データベースユーザーのみがこの IAM ロールを使用するアクセス許可を持つよう指定されます。

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
  {
    "Effect": "Allow",
    "Principal": {
      "Service": "redshift.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringEquals": {
        "sts:ExternalId": [
          "arn:aws:redshift:us-west-2:123456789012:dbuser:my-cluster/user1",
          "arn:aws:redshift:us-west-2:123456789012:dbuser:my-cluster/user2"
        ]
```

```
      }
    }
  }]
}
```

7. 信頼ポリシーの更新を選択します。

AWS リージョンへの IAM ロールの制限

特定の AWS リージョンにのみアクセスできるように IAM ロールを制限できます。デフォルトでは、Amazon Redshift の IAM ロールは、1 つのリージョンに限定されません。

IAM ロールの使用をリージョンごとに制限するには、次の手順に従います。

IAM ロールの許可されたリージョンを特定する

1. IAM コンソール (https://console.aws.amazon.com/) を開きます。
2. ナビゲーションペインで [ロール] を選択します。
3. 特定のリージョンで変更するロールを選択します。
4. 信頼関係 タブを選択し、 信頼関係の編集を選択します。ユーザーに代わって Amazon Redshift が他の AWS サービスにアクセスすることを許可する新しい IAM ロールには、次のような信頼関係があります。

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "redshift.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

5. ロールの使用を許可する特定のリージョンのリストで、 Service の Principal リストを変更します。Service リストの各リージョンは、次の形式 : redshift.*region*.amazonaws.comである必要があります。

   たとえば、次の編集された信頼関係では、 us-east-1 と us-west-2 リージョンでのみ IAM ロールの使用が許可されます。

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "redshift.us-east-1.amazonaws.com",
          "redshift.us-west-2.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

6. 信頼ポリシーの更新を選択

Amazon Redshift で IAM ロールを連鎖する

クラスターにロールをアタッチすると、クラスターはそのロールを引き受け、ユーザーに代わって Amazon S3、Amazon Athena、 AWS Glue、と AWS Lambda にアクセスできます。クラスターにアタッチされたロールに必要なリソースへのアクセスがない場合、他のアカウントに属している可能性がある別のロールを連鎖することができます。クラスターは、このデータにアクセスするための連鎖ロールを一時的に引き受けます。また、ロールを連鎖してクロスアカウントアクセスを付与することもできます。連鎖における各ロールは、クラスターが連鎖の末尾のロールを引き受けるまで、連鎖の次のロールを引き受けます。関連付けることができる IAM ロールの最大数は、クォータの対象となります。詳細については、「Amazon Redshift オブジェクトのクォータ」のクォータの「Cluster IAM roles for Amazon Redshift to access other AWS services」 (他の サービスにアクセスするための Amazon Redshift のクラスター IAM ロール) を参照してください。

> **Note**
>
> チェーンが正しく動作するためには、IAM ロールを指定する必要があります。

たとえば、企業 A が企業 B に属する Amazon S3 バケットのデータにアクセスするとします。その場合、企業 A は RoleA という名前の Amazon Redshift 用の AWS サービスロールを作成し、クラスターにアタッチします。企業 B は RoleB という名前のロールを作成します。このロールは企業 B のバケットのデータへのアクセスを承認されています。企業 B のバケットのデータにアクセスするには、A が iam_role と RoleA を連鎖させる RoleB パラメータを使用して COPY コマンドを実行します。COPY オペレーションの継続期間中は、 RoleA は一時的に RoleB を引き受け Amazon S3 バケットにアクセスします。

ロールを連鎖するには、ロール間で信頼関係を確立します。別のロール (例: RoleA) を引き受けるロールには、次の連鎖したロール (例: RoleB) を引き受けることを許可するアクセス権限ポリシーがある必要があります。また、アクセス権限を渡すロール (RoleB) は、前に連鎖したロール (RoleA) にアクセス権限を渡すことができる信頼ポリシーを保持している必要があります。詳細については、IAM ユーザーガイドの IAM ロールの使用 を参照してください。

連鎖の最初のロールは、クラスターにアタッチアタッチされたロールである必要があります。最初のロールと、次のロールを引き受ける後続の各ロールには、特定のステートメントを含むポリシーがあることが必要です。このステートメントには、 Allow アクションに sts:AssumeRole　効果、そして Resource 要素に次のロールの Amazon リソースネーム (ARN) があります。この例では、RoleA に AWS アカウント 210987654321 が所有する RoleB を引き受けることを許可する、次のアクセス許可ポリシーがあります。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt1487639602000",
            "Effect": "Allow",
            "Action": [
                "sts:AssumeRole"
            ],
            "Resource": "arn:aws:iam::111122223333:role/RoleB"
       }
```

```
    ]
}
```

別のロールに渡されるロールは、そのロールを引き受けるロール、またはそのロールを所有する AWS アカウントとの信頼関係を確立する必要があります。この例では、 RoleB に RoleA との信頼関係を確立する次の信頼ポリシーがあります。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/RoleA"
      }
    }
  ]
}
```

次の信頼ポリシーにより、 RoleA の所有者である AWS アカウント 123456789012 との信頼関係が確立されます。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:root"
      }
    }
  ]
```

```
}
```

> **Note**
>
> ロールチェーン認可を特定のユーザーに制限するには、条件を定義します。詳細については、「IAM ロールへのアクセスの制限」を参照してください。

UNLOAD、COPY、または CREATE EXTERNAL SCHEMA コマンドを実行すると、 iam_role パラメータにロールの ARN のカンマ区切りのリストを含めることで、ロールを連鎖することができます。以下に iam_role パラメータ内の連鎖されたロールの構文を示します。

```
unload ('select * from venue limit 10')
to 's3://acmedata/redshift/venue_pipe_'
IAM_ROLE 'arn:aws:iam::<aws-account-id-1>:role/<role-name-1>[,arn:aws:iam::<aws-account-id-2>:role/<role-name-2>][,...]';
```

> **Note**
>
> ロールチェーン全体は一重引用符で囲まれ、スペースが含まれてはいけません。

次の例では、 RoleA は AWS アカウント 123456789012 に属するクラスターにアタッチされます。 RoleB はアカウント 210987654321 に属しており、 s3://companyb/redshift/というバケットにアクセスするアクセス許可を持っています。次の例では、RoleA と RoleB を連鎖させてデータを s3://companyb/redshift/ バケットに UNLOAD します。

```
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';
```

次の例では、COPY コマンドを使用して、前の例でアンロードしたデータをロードします。

```
copy venue
from 's3://companyb/redshift/venue_pipe_'
iam_role 'arn:aws:iam::123456789012:role/RoleA,arn:aws:iam::210987654321:role/RoleB';
```

次の例では、CREATE EXTERNAL SCHEMA は連鎖されたロールを使用してロール RoleBを引き受けます。

```
create external schema spectrumexample from data catalog
database 'exampledb' region 'us-west-2'
iam_role 'arn:aws:iam::123456789012:role/RoleA,arn:aws:iam::210987654321:role/RoleB';
```

次の例では、CREATE EXTERNAL SCHEMA は連鎖されたロールを使用してロール RoleBを引き受けます。

```
create external function lambda_example(varchar)
returns varchar
volatile
lambda 'exampleLambdaFunction'
iam_role 'arn:aws:iam::123456789012:role/RoleA,arn:aws:iam::210987654321:role/RoleB';
```

IAM ロールを使用して COPY、UNLOAD、CREATE EXTERNAL FUNCTION、CREATE EXTERNAL SCHEMA オペレーションを承認する

COPY コマンドを使用してデータを Amazon Redshift にロード (またはインポート) および UNLOAD コマンドを使用してデータを Amazon Redshift からアンロード (またはエクスポート) できます。CREATE EXTERAL FUNCTION コマンドを使用して、 AWS Lambdaから関数を呼び出すユーザー定義関数を作成できます。

Amazon Redshift Spectrum を使用する場合、 CREATE EXTERNAL SCHEMA コマンドを使用してデータを含む Amazon S3 バケットの場所を指定します。COPY、UNLOAD、CREATE EXTERNAL SCHEMA コマンドを実行するときは、セキュリティ認証情報を指定する必要があります。これらの認証情報により、Amazon S3 バケットなどのターゲット宛先との間でデータを読み書きする権限が Amazon Redshift クラスターに付与されます。

CREATE EXTERNAL FUNCTION を実行するときは、IAM ロールパラメータを使用してセキュリティ認証情報を指定します。これらの認証情報により、Amazon Redshift クラスターがAWS Lambdaから Lambda 関数を呼び出すことが許可されます。セキュリティ認証情報を指定する望ましい方法は、AWS Identity and Access Management (IAM) ロールを指定することです。COPY と UNLOAD では、一時的な認証情報を指定できます。IAM ロールの作成の詳細については、 ユーザーに代わって Amazon Redshift が他の AWS のサービスにアクセスすることを許可するを参照してください。

AWS マネジメントコンソール の外部で AWS を操作するには、プログラマチックアクセス権が必要です。プログラムによるアクセス権を付与する方法は、AWS にアクセスしているユーザーのタイプによって異なります。

ユーザーにプログラムによるアクセス権を付与するには、以下のいずれかのオプションを選択します。

プログラムによるアクセス権を必要とするユーザー	目的	方法
IAM	(推奨) 一時的な認証情報としてコンソール認証情報を使用して、AWS CLI、AWS SDK、または AWS API へのプログラマチックリクエストに署名します。	使用するインターフェイスの指示に従ってください。 • AWS CLI については、「AWS Command Line Interface ユーザーガイド」の「AWS ローカル開発用のログイン」を参照してください。 • AWS SDK については、「AWS SDK およびツールリファレンスガイド」の「AWS ローカル開発用のログイン」を参照してください。
ワークフォースアイデンティティ (IAM アイデンティティセンターで管理されているユーザー)	一時的な認証情報を使用して、AWS CLI、AWS SDK、または AWS API へのプログラムによるリクエストに署名します。	使用するインターフェイスの指示に従ってください。 • AWS CLI については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM アイデンティティセンター を使用するための AWS CLI の設定」を参照してください。 • AWS SDK、ツール、および AWS API について

プログラムによるアクセス権を必要とするユーザー	目的	方法
		は、AWS SDK とツールリファレンスガイドの「IAM アイデンティティセンター認証」を参照してください。
IAM	一時的な認証情報を使用して、AWS CLI、AWS SDK、または AWS API へのプログラムによるリクエストに署名します。	IAM ユーザーガイドの「AWS リソースでの一時的な認証情報の使用」の指示に従ってください。
IAM	(非推奨) 長期的な認証情報を使用して、AWS CLI、AWS SDK、または AWS API へのプログラムによるリクエストに署名します。	使用するインターフェイスの指示に従ってください。 • AWS CLI については、「AWS Command Line Interface ユーザーガイド」の「IAM ユーザー認証情報を使用した認証」を参照してください。 • AWS SDK とツールについては、「AWS SDK とツールリファレンスガイド」の「長期認証情報を使用して認証する」を参照してください。 • AWS API については、「IAM ユーザーガイド」の「IAM ユーザーのアクセスキーの管理」を参照してください。

IAM ロールを使用する手順は次のとおりです。

- Amazon Redshift クラスターで使用する IAM ロールを作成します。
- IAM ロールをクラスターに関連付けます。
- COPY、UNLOAD、CREATE EXTERNAL SCHEMA または CREATE EXTERNAL FUNCTION コマンドを呼び出す場合、IAM ロールの ARN を含めます。

IAM ロールをクラスターに関連付ける

Amazon Redshift が他の AWS のサービスにアクセスすることを許可する IAM ロールを作成した後、そのロールを Amazon Redshift クラスターに関連付ける必要があります。ロールを使用してデータをロードまたはアンロードする前に、これを行う必要があります。

IAM ロールをクラスターに関連付けるために必要なアクセス許可

IAM ロールをクラスターに関連付ける場合、ユーザーには、その IAM ロールの `iam:PassRole` アクセス許可が必要です。このアクセス許可により、ユーザーが IAM ロールを Amazon Redshift クラスターに関連付けることを管理者が制限できます。ベストプラクティスとして、アクセス許可ポリシーを IAM ロールにアタッチし、それを必要に応じてユーザーやグループに割り当てることをお勧めします。詳細については、「Amazon Redshift での Identity and Access Management」を参照してください。

次の例に示す IAM ポリシーは、以下のアクションの実行を許可するユーザーにアタッチできます。

- そのユーザーアカウントによって所有されるすべての Amazon Redshift クラスターの詳細を取得します。
- 3 つの IAM ロールのいずれかを 2 つの Amazon Redshift クラスターのいずれかに関連付けます。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "redshift:DescribeClusters",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
```

```
            "Action": [
                 "redshift:ModifyClusterIamRoles",
                 "redshift:CreateCluster"
            ],
            "Resource": [
                 "arn:aws:redshift:us-east-1:123456789012:cluster:my-redshift-
cluster",
                 "arn:aws:redshift:us-east-1:123456789012:cluster:my-second-
redshift-cluster"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [
                "arn:aws:iam::123456789012:role/MyRedshiftRole",
                "arn:aws:iam::123456789012:role/SecondRedshiftRole",
                "arn:aws:iam::123456789012:role/ThirdRedshiftRole"
             ]
        }
    ]
}
```

ユーザーは、適切なアクセス許可を取得すると、IAM ロールを Amazon Redshift クラスターに関連付けることができます。これで、IAM ロールは COPY や UNLOAD などの Amazon Redshift コマンドで使用できるようになります。

IAM ポリシーの詳細については、 IAM ユーザーガイドの IAM ポリシーの概要 を参照してください。

クラスターに関連付けられる IAM ロールの管理

IAM ロールは、Amazon Redshift クラスターの作成時にクラスターに関連付けることができます。または、既存のクラスターを変更し、1 つまたは複数の IAM ロールの関連付けを追加または削除できます。

以下の点に注意してください。

- 関連付けることができる IAM ロールの最大数は、クォータの対象となります。
- IAM ロールは複数の Amazon Redshift クラスターと関連付けることができます。
- IAM ロールとクラスターの両方が同じ AWS アカウントによって所有されている場合にのみ、IAM ロールを Amazon Redshift クラスターに関連付けることができます。

次の手順を使用して、コンソールでクラスターに対する IAM ロールの関連付けを管理できます。

IAM ロールの関連付けを管理するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [Clusters] (クラスター) を選択し、更新するクラスターを選択します。
3. アクションで IAM ロールの管理 を選択し、クラスターに現在関連付けられている IAM ロールのリストを表示します。
4. IAM ロールの管理 ページで、追加する IAMロールを使用可能なオプションから選択した後、IAM ロールの追加を選択します。
5. 完了 を選択して変更を保存します。

次のアプローチを使用して、 AWS CLI でクラスターに対する IAM ロールの関連付けを管理できます。

クラスターの作成時に IAM ロールをクラスターに関連付けるには、 `--iam-role-arns` コマンドの `create-cluster` パラメータで IAM ロールの Amazon リソースネーム (ARN) を指定します。`create-cluster` コマンドを呼び出すときに追加できる IAM ロールの最大数は、クォータの対象となります。

IAM ロールの Amazon Redshift クラスターへの関連付けおよび関連付けの解除は非同期処理です。`describe-clusters` コマンドを呼び出して、すべての IAM ロールのクラスター関連付けのステータスを取得できます。

次の例では、2 つの IAM ロールを `my-redshift-cluster` という名前の新しく作成されたクラスターに関連付けます。

```
aws redshift create-cluster \
    --cluster-identifier "my-redshift-cluster" \
    --node-type "ra3.4xlarge" \
    --number-of-nodes 16 \
    --iam-role-arns "arn:aws:iam::123456789012:role/RedshiftCopyUnload" \
                    "arn:aws:iam::123456789012:role/SecondRedshiftRole"
```

既存の Amazon Redshift クラスターに IAM ロールを関連付けるには、 `modify-cluster-iam-roles` コマンドの `--add-iam-roles` パラメータで IAM ロールの Amazon リソースネーム (ARN)

を指定します。modify-cluster-iam-roles コマンドを呼び出すときに追加できる IAM ロールの最大数は、クォータの対象となります。

次の例では、IAM ロールを my-redshift-clusterという名前の既存のクラスターに関連付けます。

```
aws redshift modify-cluster-iam-roles \
    --cluster-identifier "my-redshift-cluster" \
    --add-iam-roles "arn:aws:iam::123456789012:role/RedshiftCopyUnload"
```

クラスターから IAM ロールの関連付けを解除するには、modify-cluster-iam-roles コマンドの --remove-iam-roles パラメータで IAM ロールの ARN を指定します。modify-cluster-iam-rolesmodify-cluster-iam-roles コマンドを呼び出すときに削除できる IAM ロールの最大数は、クォータの対象となります。

次の例では、 my-redshift-clusterという名前のクラスターから 123456789012 AWS アカウントの IAM ロールの関連付けを削除します。

```
aws redshift modify-cluster-iam-roles \
    --cluster-identifier "my-redshift-cluster" \
    --remove-iam-roles "arn:aws:iam::123456789012:role/RedshiftCopyUnload"
```

AWS CLIを使用してクラスターの IAM ロールの関連付けをリスト化する

Amazon Redshift クラスターに関連付けられている IAM ロールおよび IAM ロールの関連付けのステータスをリスト化するには、 describe-clusters コマンドを呼び出します。クラスターに関連付けられた各 IAM ロールの ARN は、次の出力例のように、 IamRoles リストで返されます。

クラスターに関連付けられているロールは、in-sync のステータスを示します。クラスターとの関連付けの途中であるロールは、adding のステータスを示します。クラスターから関連付けを解除されるロールは、removing のステータスを示します。

```
{
    "Clusters": [
        {
            "ClusterIdentifier": "my-redshift-cluster",
            "NodeType": "ra3.4xlarge",
            "NumberOfNodes": 16,
            "IamRoles": [
```

```
                {
                    "IamRoleArn": "arn:aws:iam::123456789012:role/MyRedshiftRole",
                    "IamRoleApplyStatus": "in-sync"
                },
                {
                    "IamRoleArn": "arn:aws:iam::123456789012:role/SecondRedshiftRole",
                    "IamRoleApplyStatus": "in-sync"
                }
            ],
            ...
        },
        {
            "ClusterIdentifier": "my-second-redshift-cluster",
            "NodeType": "ra3.4xlarge",
            "NumberOfNodes": 10,
            "IamRoles": [
                {
                    "IamRoleArn": "arn:aws:iam::123456789012:role/MyRedshiftRole",
                    "IamRoleApplyStatus": "in-sync"
                },
                {
                    "IamRoleArn": "arn:aws:iam::123456789012:role/SecondRedshiftRole",
                    "IamRoleApplyStatus": "in-sync"
                },
                {
                    "IamRoleArn": "arn:aws:iam::123456789012:role/ThirdRedshiftRole",
                    "IamRoleApplyStatus": "in-sync"
                }
            ],
            ...
        }
    ]
}
```

AWS CLIの使用の詳細については、 AWS CLI ユーザーガイドを参照してください。

Amazon Redshift 用にデフォルトの IAM ロールを作成する

Redshift コンソールから IAM ロールを作成すると、Amazon Redshift が AWS アカウント 内でプログラム的にロールを作成します。その後、作成したロールに対し、既存の AWS 管理ポリシーを自動的にアタッチします。このアプローチにより、ユーザーは Redshift コンソール内に留まることができ、IAM コンソールに切り替えることなくロールを作成できます Amazon Redshift コンソールで作

成された既存の IAM ロールにおいて、アクセス許可をより詳細に制御する場合には、カスタマイズされた管理ポリシーを IAM ロールにアタッチします。

コンソールで作成された IAM ロール

Amazon Redshift コンソールを使用して作成されたすべての IAM ロールは、Amazon Redshift により追跡されます。Amazon Redshift は、新しいすべてのクラスターの作成と、スナップショットからのクラスター復元のために、最新のデフォルトの IAM ロールを事前に選択します。

IAM ロールは、SQL コマンドを実行することが許可されたポリシーを持つコンソールから作成できます。ここで実行されるコマンドには、COPY、UNLOAD、CREATE EXTERNAL FUNCTION、CREATE EXTERNAL TABLE、CREATE EXTERNAL SCHEMA、CREATE MODEL、および CREATE LIBRARY などがあります オプションで、カスタムポリシーを作成し IAM ロールにアタッチすることで、AWS リソースへのユーザーアクセスをより詳細に制御できます。

コンソールを使用して IAM ロールを作成し、クラスターのデフォルトとして設定した場合には、認証と認可を実行するために IAM ロールの Amazon リソースネーム (ARN) を指定する必要はありません。

クラスター用としてコンソールから作成する IAM ロールには、`AmazonRedshiftAllCommandsFullAccess` マネージドのポリシーが自動的にアタッチされます。この IAM ロールにより Amazon Redshift は、IAM アカウントにある AWS リソースにおいて、データのコピー、解放、クエリ、および分析を行えるようになります。このマネージドポリシーは、COPY、UNLOAD、CREATE EXTERNAL FUNCTION、CREATE EXTERNAL SCHEMA、CREATE MODEL、CREATE LIBRARY の各オペレーションへのアクセスを提供します。また、このポリシーでは、関連する AWS サービス (Amazon S3、Amazon CloudWatch Logs、Amazon SageMaker AI、AWS Glue など) のために SELECT ステートメントを実行するアクセス許可も付与します。

CREATE EXTERNAL FUNCTION、CREATE EXTERNAL SCHEMA、CREATE MODEL、および CREATE LIBRARY コマンドには、`default` キーワードが付加されています。これらのコマンドにこのキーワードがある場合、Amazon Redshift は、コマンドの実行時にデフォルトとして設定され、クラスターに関連付けられた IAM ロールを使用します。DEFAULT_IAM_ROLE コマンドを実行すると、クラスターにアタッチされている現在のデフォルトの IAM ロールを確認できます。

Redshift クラスターのデフォルトとして作成ならびに設定された IAM ロールのアクセス権限を制御する場合は、ASSUMEROLE 権限を使用します。このアクセス制御は、先に一覧で示したコマンドを実行する際に、データベースユーザーおよびグループに対し適用されます。IAM ロールのユーザーまたはグループに ASSUMEROLE 権限を付与すると、そのユーザーまたはグループは、コマン

ドの実行時にそのロールを引き受けることができるようになります。ASSUMEROLE 権限を使用することにより、必要に応じて適切なコマンドにアクセス許可を付与することができます。

Amazon Redshift コンソールを使用すると、以下の操作を実行できます。

- IAM ロールをデフォルトとして作成する
- クラスターかのら IAM ロールの削除
- クラスターへの IAM ロールの関連付け
- IAM ロールをデフォルトとして設定する
- クラスターでの IAM ロールのデフォルト解除

AmazonRedshiftAllCommandsFullAccess 管理ポリシーによるアクセスの許可

次に、クラスターのデフォルトとして設定された IAM ロールに特定のアクションを許可する、AmazonRedshiftAllCommandsFullAccess 管理ポリシーでのアクセス許可の例を示します。許可ポリシーがアタッチされた IAM ロールにより、ユーザーまたはグループで実行可能なことと不可能なことが制御されます。これらのアクセス許可が付与されることで、Amazon S3 からの COPY コマンドや、UNLOAD、CREATE MODEL コマンドの実行が可能になります。

```
{
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetBucketAcl",
                "s3:GetBucketCors",
                "s3:GetEncryptionConfiguration",
                "s3:GetBucketLocation",
                "s3:ListBucket",
                "s3:ListAllMyBuckets",
                "s3:ListMultipartUploadParts",
                "s3:ListBucketMultipartUploads",
                "s3:PutObject",
                "s3:PutBucketAcl",
                "s3:PutBucketCors",
                "s3:DeleteObject",
                "s3:AbortMultipartUpload",
                "s3:CreateBucket"
            ],
            "Resource": [
                "arn:aws:s3:::redshift-downloads",
```

```
                "arn:aws:s3:::redshift-downloads/*",
                "arn:aws:s3:::*redshift*",
                "arn:aws:s3:::*redshift*/*"
            ]
}
```

次の例では、クラスターのデフォルトとして設定された IAM ロールのために特定のアクションを許可する、AmazonRedshiftAllCommandsFullAccess 管理ポリシーにより付与されるアクセス許可を表示します。許可ポリシーがアタッチされた IAM ロールにより、ユーザーまたはグループで実行可能なことと不可能なことが制御されます。次のアクセス許可が付与されている場合には、CREATE EXTERNAL FUNCTION コマンドを実行できます。

```
{
    "Action": [
        "lambda:InvokeFunction"
    ],
    "Resource": "arn:aws:lambda:*:*:function:*redshift*"
}
```

次の例では、クラスターのデフォルトとして設定された IAM ロールのために特定のアクションを許可する、AmazonRedshiftAllCommandsFullAccess 管理ポリシーにより付与されるアクセス許可を表示します。許可ポリシーがアタッチされた IAM ロールにより、ユーザーまたはグループで実行可能なことと不可能なことが制御されます。次のアクセス許可では、Amazon Redshift Spectrum に必要な、CREATE EXTERNAL SCHEMA コマンドおよび CREATE EXTERNAL コマンドを実行できます。

```
{
            "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": [
                "arn:aws:glue:*:*:table/*redshift*/*",
                "arn:aws:glue:*:*:catalog",
                "arn:aws:glue:*:*:database/*redshift*"
            ]
}
```

次に、クラスターのデフォルトとして設定された IAM ロールに対する特定のアクションを許可する、AmazonRedshiftAllCommandsFullAccess 管理ポリシーでのアクセス許可の例を示します。許可ポリシーがアタッチされた IAM ロールにより、ユーザーまたはグループで実行可能なことと不可能なことが制御されます。次の権限が付与されている場合は、横串検索を使用する CREATE EXTERNAL SCHEMA コマンドを実行できます。

```
{
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetResourcePolicy",
                "secretsmanager:GetSecretValue",
                "secretsmanager:DescribeSecret",
                "secretsmanager:ListSecretVersionIds"
            ],
            "Resource": [
                "arn:aws:secretsmanager:*:*:secret:*Redshift*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword",
                "secretsmanager:ListSecrets"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "secretsmanager:ResourceTag/Redshift": "true"
```

```
                }
            }
},
```

コンソールを使用してクラスター用に作成された IAM ロールの管理

Amazon Redshift コンソールから IAM ロールを作成し、変更、および削除するには、コンソールの [Clusters] (クラスター) セクションを使用します。

IAM ロールをデフォルトとして作成する

コンソールでは、AmazonRedshiftAllCommandsFullAccess ポリシーを自動的にアタッチしながら、クラスター用の IAM ロールを作成できます。作成した新しい IAM ロールにより、IAM アカウントの Amazon リソースからデータをコピー、ロード、クエリ、分析することが、Amazon Redshift に許可されます。

クラスターのデフォルトとしては、IAM ロールを 1 つだけ設定できます。既存の IAM ロールがデフォルトとして既に割り当てられている状態で、別の IAM ロールをクラスターのデフォルトとして作成すると、新しい IAM ロールがデフォルトとしてもう一方の IAM ロールを置き換えます。

新しいクラスターと IAM ロールセットを新しいクラスターのデフォルトとして作成するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [クラスター] を選択します。現在の AWS リージョン にあるアカウントのクラスターがリストされています。各クラスターのプロパティのサブセットが、リストの列に表示されます。
3. [クラスターを作成] を選択して、クラスターを作成します。
4. コンソールページの指示に従って [クラスター設定] のプロパティを入力します。この手順の詳細については、「クラスターの作成」を参照してください。
5. (オプション) [Load sample data] (サンプルデータをロード) をクリックし Amazon Redshift クラスターにサンプルのデータセットをロードして、データをクエリするためのクエリエディタの使用を開始します。

   ファイアウォールの内側にいる場合、データベースポートはインバウンド接続を受け入れる、開いているポートである必要があります。
6. コンソールページの指示に従って [Database configurations] (データベース設定) にプロパティを入力します。

7. [Cluster permissions] (クラスターのアクセス許可) で、[Manage IAM roles] (IAM ロールの管理) から [Create IAM role] (IAM ロールの作成) を選択します。
8. 以下のいずれかの方法を選択して、IAM ロールがアクセスする Amazon S3 バケットを指定します。

   - [No additional Amazon S3 bucket] (追加の Amazon S3 バケットはありません) を選択し、特定の Amazon S3 バケットを指定せずに IAM ロールを作成します。
   - Amazon Redshift クラスターへのアクセス権を持つユーザーが、AWS アカウント 内の任意の Amazon S3 バケットとそのコンテンツにもアクセスできるようにするには、[Any Amazon S3 bucket] (すべての Amazon S3 バケット) を選択します。
   - IAM ロールを作成中の (1 つ以上の) Amazon S3 バケットに、ユーザーがアクセスできるようにするには、[Specific Amazon S3 buckets] (特定の Amazon S3 バケット) を選択します。次に、テーブルから Amazon S3 バケットを 1 つ以上選択します。

9. [Create IAM role as default] (デフォルトとして IAM ロールを作成する) をクリックします。Amazon Redshift は、クラスター用に IAM ロールを自動的に作成し、デフォルトとして設定します。
10. [クラスターを作成] を選択して、クラスターを作成します。クラスターの使用準備ができるまで、数分かかることがあります。

クラスターかのら IAM ロールの削除

クラスターからは、1 つ以上の IAM ロールを削除できます。

クラスターから IAM ロールを削除するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [クラスター] を選択します。現在の AWS リージョン にあるアカウントのクラスターがリストされています。各クラスターのプロパティのサブセットが、リストの列に表示されます。
3. IAM ロールを削除するクラスターを選択します。
4. [Cluster permissions] (クラスターのアクセス許可) で、クラスターから削除する 1 つ以上の IAM ロールを選択します。
5. [Manage IAM roles] (IAM ロールの管理) で、[Remove IAM roles] (IAM ロールを削除) をクリックします。

クラスターへの IAM ロールの関連付け

クラスターには、1 つ以上の IAM ロールを関連付けることができます。

IAM ロールをクラスターに関連付けるには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [クラスター] を選択します。現在の AWS リージョン にあるアカウントのクラスターがリストされています。各クラスターのプロパティのサブセットが、リストの列に表示されます。
3. IAM ロールを関連付けるクラスターを選択します。
4. [Cluster permissions] (クラスターのアクセス許可) で、クラスターに関連付ける IAM ロールを 1 つ以上選択します。
5. [Manage IAM roles] (IAM ロールの管理) で、[Associate IAM roles] (IAM ロールを関連付ける) を選択します。
6. クラスターに関連付ける IAM ロールを 1 つ以上選択します。
7. その後、[Associate IAM roles] (IAM ロールの関連付け) をクリックします。

IAM ロールをデフォルトとして設定する

IAM ロールをクラスターのデフォルトに設定することができます。

IAM ロールをクラスターのデフォルトにするには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [クラスター] を選択します。現在の AWS リージョン にあるアカウントのクラスターがリストされています。各クラスターのプロパティのサブセットが、リストの列に表示されます。
3. IAM ロールをデフォルトに設定するクラスターを選択します。
4. [Cluster permissions] (クラスターのアクセス許可) にある [Associated IAM roles] (関連付けられた IAM ロール) から、クラスターのデフォルトとして設定する IAM ロールを選択します。
5. [Set default] (デフォルトの設定) で、[Make default] (デフォルトにする) をクリックします。
6. プロンプトが表示されたら、[Set default] (デフォルトに設定) をクリックして、指定した IAM ロールをデフォルトにすることを確認します。

クラスターでの IAM ロールのデフォルト解除

クラスターのデフォルトとしての IAM ロールの使用を終了することができます。

クラスターでの IAM ロールのデフォルト使用をクリアするには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [クラスター] を選択します。現在の AWS リージョン にあるアカウントのクラスターがリストされています。各クラスターのプロパティのサブセットが、リストの列に表示されます。
3. IAM ロールを関連付けるクラスターを選択します。
4. [Cluster permissions] (クラスターのアクセス許可) にある [Associated IAM roles] (関連付けられた IAM ロール) から、デフォルトの IAM ロールを選択します。
5. [Set default] (デフォルトの設定) で、[Clear default] (デフォルトをクリアする) をクリックします。
6. プロンプトが表示されたら、[Clear default] (デフォルトをクリア) をクリックして、指定された IAM ロールのデフォルト使用を解除することを確認します。

クラスターで作成された IAM ロールの AWS CLI による管理

クラスターで作成された IAM ロールを、AWS CLI を使用して管理することができます。

IAM ロールをデフォルトとして設定しながら Amazon Redshift クラスターを作成するには

IAM ロールをデフォルトとして設定しながら、Amazon Redshift クラスターを作成するには、AWS CLI の `aws redshift create-cluster` コマンドを使用します。

以下の AWS CLI コマンドは、Amazon Redshift クラスターと myrole1 という名前の IAM ロールを作成します。この AWS CLI コマンドでは、同時に myrole1 をクラスターのデフォルトとして設定します。

```
aws redshift create-cluster \
    --node-type dc2.large \
    --number-of-nodes 2 \
    --master-username adminuser \
    --master-user-password TopSecret1 \
    --cluster-identifier mycluster \
```

```
    --iam-roles 'arn:aws:iam::012345678910:role/myrole1'
 'arn:aws:iam::012345678910:role/myrole2' \
    --default-iam-role-arn 'arn:aws:iam::012345678910:role/myrole1'
```

以下のスニペットに、この応答例を示します。

```
{
    "Cluster": {
        "ClusterIdentifier": "mycluster",
        "NodeType": "dc2.large",
        "MasterUsername": "adminuser",
        "DefaultIamRoleArn": "arn:aws:iam::012345678910:role/myrole1",
        "IamRoles": [
            {
                "IamRoleArn": "arn:aws:iam::012345678910:role/myrole1",
                "ApplyStatus": "adding"
            },
            {
                "IamRoleArn": "arn:aws:iam::012345678910:role/myrole2",
                "ApplyStatus": "adding"
            }
        ]
        ...
    }
}
```

Amazon Redshift クラスターに IAM ロールを 1 つ以上追加するには

クラスターに 1 つ以上の IAM ロールを追加し関連付けを行うには、AWS CLI の `aws redshift modify-cluster-iam-roles` コマンドを使用します。

以下の AWS CLI コマンドでは、クラスターに `myrole3` および `myrole4` が追加されます。

```
aws redshift modify-cluster-iam-roles \
    --cluster-identifier mycluster \
    --add-iam-roles 'arn:aws:iam::012345678910:role/myrole3'
 'arn:aws:iam::012345678910:role/myrole4'
```

以下のスニペットに、この応答例を示します。

```
{
```

```
    "Cluster": {
        "ClusterIdentifier": "mycluster",
        "NodeType": "dc2.large",
        "MasterUsername": "adminuser",
        "DefaultIamRoleArn": "arn:aws:iam::012345678910:role/myrole1",
        "IamRoles": [
            {
                "IamRoleArn": "arn:aws:iam::012345678910:role/myrole1",
                "ApplyStatus": "in-sync"
            },
            {
                "IamRoleArn": "arn:aws:iam::012345678910:role/myrole2",
                "ApplyStatus": "in-sync"
            },
            {
                "IamRoleArn": "arn:aws:iam::012345678910:role/myrole3",
                "ApplyStatus": "adding"
            },
            {
                "IamRoleArn": "arn:aws:iam::012345678910:role/myrole4",
                "ApplyStatus": "adding"
            }
        ],
        ...
    }
}
```

Amazon Redshift クラスターから 1 つ以上の IAM ロールを削除するには

クラスターに関連付けられた 1 つ以上の IAM ロールを削除するには、AWS CLI の `aws redshift modify-cluster-iam-roles` コマンドを使用します。

以下の AWS CLI コマンドにより、クラスターから `myrole3` および `myrole4` が削除されます。

```
aws redshift modify-cluster-iam-roles \
    --cluster-identifier mycluster \
    --remove-iam-roles 'arn:aws:iam::012345678910:role/myrole3'
 'arn:aws:iam::012345678910:role/myrole4'
```

以下のスニペットに、この応答例を示します。

```
{
```

```
    "Cluster": {
        "ClusterIdentifier": "mycluster",
        "NodeType": "dc2.large",
        "MasterUsername": "adminuser",
        "DefaultIamRoleArn": "arn:aws:iam::012345678910:role/myrole1",
        "IamRoles": [
            {
                "IamRoleArn": "arn:aws:iam::012345678910:role/myrole1",
                "ApplyStatus": "in-sync"
            },
            {
                "IamRoleArn": "arn:aws:iam::012345678910:role/myrole2",
                "ApplyStatus": "in-sync"
            },
            {
                "IamRoleArn": "arn:aws:iam::012345678910:role/myrole3",
                "ApplyStatus": "removing"
            },
            {
                "IamRoleArn": "arn:aws:iam::012345678910:role/myrole4",
                "ApplyStatus": "removing"
            }
        ],
        ...
    }
}
```

関連付けられた IAM ロールをクラスターのデフォルトとして設定するには

関連付けられた IAM ロールをクラスターのデフォルトとして設定するには、AWS CLI の `aws redshift modify-cluster-iam-roles` コマンドを使用します。

以下の AWS CLI コマンドでは、myrole2 がクラスターのデフォルトとして指定されます。

```
aws redshift modify-cluster-iam-roles \
    --cluster-identifier mycluster \
    --default-iam-role-arn 'arn:aws:iam::012345678910:role/myrole2'
```

以下のスニペットに、この応答例を示します。

```
{
    "Cluster": {
```

```
        "ClusterIdentifier": "mycluster",
        "NodeType": "dc2.large",
        "MasterUsername": "adminuser",
        "DefaultIamRoleArn": "arn:aws:iam::012345678910:role/myrole2",
        "IamRoles": [
            {
                "IamRoleArn": "arn:aws:iam::012345678910:role/myrole1",
                "ApplyStatus": "in-sync"
            },
            {
                "IamRoleArn": "arn:aws:iam::012345678910:role/myrole2",
                "ApplyStatus": "in-sync"
            }
        ],
        ...
    }
}
```

関連付けられていない IAM ロールをクラスターのデフォルトに設定するには

関連付けられていない IAM ロールをクラスターのデフォルトとして設定する場合は、AWS CLI の `aws redshift modify-cluster-iam-roles` コマンドを使用します。

以下の AWS CLI コマンドは、`myrole2` を Amazon Redshift クラスターに追加した上で、そのクラスターのデフォルトとして設定します。

```
aws redshift modify-cluster-iam-roles \
    --cluster-identifier mycluster \
    --add-iam-roles 'arn:aws:iam::012345678910:role/myrole3' \
    --default-iam-role-arn 'arn:aws:iam::012345678910:role/myrole3'
```

以下のスニペットに、この応答例を示します。

```
{
    "Cluster": {
        "ClusterIdentifier": "mycluster",
        "NodeType": "dc2.large",
        "MasterUsername": "adminuser",
        "DefaultIamRoleArn": "arn:aws:iam::012345678910:role/myrole3",
        "IamRoles": [
            {
                "IamRoleArn": "arn:aws:iam::012345678910:role/myrole1",
```

```
                "ApplyStatus": "in-sync"
            },
            {
                "IamRoleArn": "arn:aws:iam::012345678910:role/myrole2",
                "ApplyStatus": "in-sync"
            },
            {
                "IamRoleArn": "arn:aws:iam::012345678910:role/myrole3",
                "ApplyStatus": "adding"
            }
        ],
        ...
    }
}
```

スナップショットからクラスターを復元し、そこに IAM ロールをデフォルトとして設定するには

スナップショットからクラスターを復元する際、既存の IAM ロールを関連付けるか、新しい IAM ロールを作成しクラスターのデフォルトとして設定することができます。

スナップショットから復元した Amazon Redshift クラスターに、IAM ロールをデフォルトとして設定するには、AWS CLI の `aws redshift restore-from-cluster-snapshot` コマンドを使用します。

以下の AWS CLI コマンドは、スナップショットからクラスターを復元した上で、そのクラスターのデフォルトとして `myrole2` を設定します。

```
aws redshift restore-from-cluster-snapshot \
    --cluster-identifier mycluster-clone \
    --snapshot-identifier my-snapshot-id
    --iam-roles 'arn:aws:iam::012345678910:role/myrole1'
 'arn:aws:iam::012345678910:role/myrole2' \
    --default-iam-role-arn 'arn:aws:iam::012345678910:role/myrole1'
```

以下のスニペットに、この応答例を示します。

```
{
    "Cluster": {
        "ClusterIdentifier": "mycluster-clone",
        "NodeType": "dc2.large",
        "MasterUsername": "adminuser",
```

```
        "DefaultIamRoleArn": "arn:aws:iam::012345678910:role/myrole1",
        "IamRoles": [
            {
                "IamRoleArn": "arn:aws:iam::012345678910:role/myrole1",
                "ApplyStatus": "adding"
            },
            {
                "IamRoleArn": "arn:aws:iam::012345678910:role/myrole2",
                "ApplyStatus": "adding"
            }
        ],
        ...
    }
}
```

フェデレーション ID を使用して、ローカルリソースと Amazon Redshift Spectrum の外部テーブルへの Amazon Redshift アクセスを管理する

GetDatabaseCredentials から提供された認証情報を使用して AWS の ID フェデレーションを使用すると、ローカルデータや外部データへの認可とアクセスを簡素化できます。このチュートリアルでは、特定の IAM ロールを使用する代わりに、AWS ID フェデレーションを使用してリソースへのアクセスを提供する方法を説明します。

現在、Amazon S3 にある外部データへのアクセスをユーザーに許可するには、アクセス権限ポリシーで定義されたアクセス権限を持つ IAM ロールを作成します。これにより、ロールが割り当てられたユーザーは、外部データにアクセスできます。これは有効ですが、特定のユーザーに特定の列を使用できないようにするなど、きめ細かいルールを提供したい場合は、外部スキーマで追加の設定が必要な場合があります。

ID フェデレーションでは、GetDatabaseCredentials から提供された認証情報を使用することで、指定や変更が簡単にできる詳細な IAM ルールを介して AWS Glue や Redshift Spectrum リソースにアクセスできます。これにより、ビジネスルールに適合したアクセスを簡単に適用できます。

フェデレーション認証情報を使用すると、次のような利点があります。

- Redshift Spectrum では、クラスターにアタッチされた IAM ロールを管理する必要はありません。
- クラスター管理者は、異なる IAM コンテキストを持ち、コンシューマーがアクセス可能な外部スキーマを作成できます。これは、例えば、異なるコンシューマーが同じ外部スキーマをクエリして、返されたレコードに複数のフィールドを取得する場合、テーブルで列のフィルタリングを実行する場合に便利です。

- Amazon Redshift へのクエリは、ロールを使用するだけでなく、IAM アクセス許可を持つユーザーを使用して実行できます。

フェデレーション ID でログインする ID の準備

フェデレーション ID でログインする前に、準備のための手順をいくつか実行する必要があります。これらの手順は、Amazon S3 バケットに保存されたデータファイルを参照する既存の Redshift Spectrum 外部スキーマがあり、そのバケットが Amazon Redshift クラスターまたは Amazon Redshift サーバーレスデータウェアハウスと同じアカウントにあることを前提としています。

1. IAM ID を作成する。これはユーザーまたは IAM ロールのどちらでも構いません。IAM がサポートする任意の名前を使用してください。
2. ID に許可ポリシーをアタッチします。次のいずれかを指定します。

   - `redshift:GetClusterCredentialsWithIAM` (Amazon Redshift でプロビジョニングされたクラスターの場合)
   - `redshift-serverless:GetCredentials` (Amazon Redshift サーバーレスの場合)

   IAM コンソールを使用して、ポリシーエディタで権限を追加できます。

   IAM ID には、外部データにアクセスするための権限も必要です。以下の AWS マネージドポリシーを直接追加して、Amazon S3 へのアクセスを許可します。

   - `AmazonS3ReadOnlyAccess`
   - `AWSGlueConsoleFullAccess`

   AWS Glue を使用して外部データを準備する場合は、最後のマネージドポリシーが必要です。Amazon Redshift Spectrum へのアクセスを許可する手順の詳細については、Amazon Redshift と Redshift Spectrum の入門ガイドの一部である「Amazon Redshift 用の IAM ロールの作成」を参照してください。これは Redshift Spectrum にアクセスするための IAM ポリシーを追加する手順を示しています。
3. Amazon Redshift に接続する SQL クライアントの設定。Amazon Redshift JDBC ドライバーを使用して、ユーザーの認証情報をツールの認証情報プロパティに追加します。SQL Workbench/J のようなクライアントはこれに適しています。次のクライアント接続拡張プロパティを設定します。

   - AccessKeyID — アクセスキーの ID。

- SecretAccessKey – シークレットアクセスキー (暗号化を使用しない場合は、シークレットキーを送信することによるセキュリティ上のリスクに注意が必要です)。
- SessionToken – IAM ロールに必要な一連の認証情報。
- groupFederation — プロビジョニングされたクラスターにフェデレーション ID を設定する場合は、`true` に設定します。Amazon Redshift サーバーレスを使用している場合は、このパラメータを設定しないでください。
- LogLevel — 整数のログレベルの値。これはオプションです。

4. Amazon Redshift または Amazon Redshift サーバーレスコンソールにある JDBC エンドポイントの URL を設定します。URL スキーマを jdbc:redshift:iam: に置き換え、次の書式を使用してください。

   - Amazon Redshift でプロビジョニングされたクラスターの書式: `jdbc:redshift:iam://<cluster_id>.<unique_suffix>.<region>.redshift.amazonaws.com:<port>/<database_name>`

     例:`jdbc:redshift:iam://test1.12345abcdefg.us-east-1.redshift.amazonaws.com:5439/dev`
   - Amazon Redshift サーバーレスの書式: `jdbc:redshift:iam://<workgroup-name>.<account-number>.<aws-region>.redshift-serverless.amazonaws.com:5439:<port>/<database_name>`

     例:`jdbc:redshift:iam://default.123456789012.us-east-1.redshift-serverless.amazonaws.com:5439/dev`

   IAM ID を使用して初めてデータベースに接続すると、Amazon Redshift は自動的に同じ名前の Amazon Redshift ID を作成し、プレフィックスとして IAM ユーザーの場合は `IAM:`、IAM ロールの場合は `IAMR:` を付けます。このトピックの残りのステップでは、ユーザー向けの例を示します。

   Redshift ユーザーが自動的に作成されない場合は、管理者アカウントを使用して `CREATE USER` ステートメントを実行し、`IAM:<user name>` の形式でユーザー名を指定することで作成できます。

5. Amazon Redshift クラスター管理者として、Redshift ユーザーに外部スキーマへのアクセスに必要なアクセス権限を付与します。

```
GRANT ALL ON SCHEMA my_schema to "IAM:my_user";
```

Redshift ユーザーに外部スキーマにテーブルを作成する権限を付与するには、そのユーザーがスキーマの所有者である必要があります。例えば、次のようになります。

```
ALTER SCHEMA my_schema owner to "IAM:my_user";
```

6. 設定を検証するには、アクセス許可を付与した後で、SQL クライアントを使用してユーザーとしてクエリを実行します。このクエリサンプルは、外部テーブルからデータを取得します。

```
SELECT * FROM my_schema.my_table;
```

Redshift Spectrum への ID と権限の伝達を始める

フェデレーション ID を外部テーブルのクエリに渡すには、CREATE EXTERNAL SCHEMA の IAM_ROLE クエリパラメータの値として SESSION を設定します。次のステップでは、外部スキーマにクエリを許可する SESSION を設定し、活用する方法を説明します。

1. ローカルテーブルと外部テーブルを作成します。AWS Glue でカタログ化された外部テーブルがこれに該当します。
2. IAM ID を使用して Amazon Redshift に接続します。前のセクションで説明したように、Amazon Redshift に ID が接続されると、Redshift データベースユーザーが作成されます。ユーザーが以前には存在しなかった場合は作成されます。新規ユーザーの場合、管理者はユーザーにクエリやテーブルの作成など、Amazon Redshift でタスクを実行する権限を付与する必要があります。
3. 管理者アカウントで Redshift に接続します。SESSION 値を使用して、外部スキーマを作成するコマンドを実行します。

```
create external schema spectrum_schema from data catalog
database '<my_external_database>'
region '<my_region>'
iam_role 'SESSION'
catalog_id '<my_catalog_id>';
```

この場合、catalog_id が設定されていることに注意してください。これは、SESSION によって特定のロールを置き換えるため、この機能で追加された新しい設定になります。

この例では、クエリ内の値は実際の値がどのように表示されているかを示しています。

```
create external schema spectrum_schema from data catalog
database 'spectrum_db'
region 'us-east-1'
iam_role 'SESSION'
catalog_id '123456789012'
```

この場合の catalog_id の値は、AWS アカウント ID です。

4. ステップ 2 で接続した IAM ID を使用して、クエリを実行して外部データにアクセスします。例:

```
select * from spectrum_schema.table1;
```

この場合、例えば table1 は Amazon S3 バケット内のファイル内の JSON 形式のデータである可能性があります。

5. クラスターにアタッチされた IAM ロールを使用する外部スキーマが既に存在し、外部データベースまたはスキーマを参照している場合は、既存のスキーマを置き換えて、このステップで説明しているフェデレーション ID を使用するか、新しいスキーマを作成できます。

SESSION はフェデレーション ID 認証情報が外部スキーマのクエリに使用されることを示します。SESSION クエリパラメータを使用する場合は、必ず catalog_id を設定してください。これは、スキーマに使用されるデータカタログを指しているため必須です。以前は、catalog_id は iam_role に割り当てられた値から取得していました。このように、ID と認可の伝達を設定すると、例えば Redshift Spectrum にフェデレーション認証情報を使用して外部スキーマをクエリする場合、IAM ロールによる認可は必要ありません。

使用に関する注意事項

よくある接続エラーは、「IAM error retrieving temp credentials: Unable to unmarshall exception response with the unmarshallers provided」です。このエラーは、レガシー JDBC ドライバーを使用していることが原因です。フェデレーテッド ID に必要な最小ドライバーバージョンは 2.1.0.9 です。「Amazon Redshift JDBC ドライバーのバージョン 2.x をダウンロードする」から JDBC ドライバーを取得できます。

その他のリソース

これらのリンクは、外部データへのアクセスを管理するための追加情報を提供します。

- IAM ロールを使用して Redshift Spectrum データに引き続きアクセスできます。詳細については、「ユーザーに代わって Amazon Redshift が他の AWS のサービスにアクセスすることを許可する」を参照してください。
- AWS Lake Formation で外部テーブルへのアクセスを管理する場合、フェデレーション IAM ID で Redshift Spectrum を使用して外部テーブルにクエリを実行できます。Redshift Spectrum が AWS Lake Formation に登録されたデータをクエリするために、クラスターにアタッチされた IAM ロールを管理する必要がなくなりました。詳細については、「Amazon Redshift Spectrum での AWS Lake Formation の使用」を参照してください。

AWS Secrets Manager を使用した Amazon Redshift 管理者パスワードの管理

Amazon Redshift は、AWS Secrets Manager と統合して、暗号化されたシークレット内で管理者認証情報を生成して管理できます。AWS Secrets Manager を使用して、管理者パスワードを API コールに置き換えて、必要な際にシークレットをプログラムで取得できます。ハードコードされた認証情報の代わりにシークレットを使用すると、認証情報が公開されたり侵害されたりするリスクを軽減できます。AWS Secrets Manager については AWS Secrets Managerユーザーガイドを参照してください。

次のいずれかのオペレーションを実行する場合、Amazon Redshift が AWS Secrets Manager を使用して管理者パスワードを管理するように指定できます。

- プロビジョニングされたクラスターまたはサーバーレス名前空間を作成する
- プロビジョニングされたクラスターまたはサーバーレス名前空間の管理者認証情報を編集、更新、または変更する
- スナップショットからのクラスターまたはサーバーレス名前空間を復元する

AWS Secrets Manager で管理者パスワードを Amazon Redshift で管理するように指定すると、Amazon Redshift はパスワードを生成して Secrets Manager に保存します。このシークレットを AWS Secrets Manager で直接アクセスして、管理者ユーザーの認証情報を取得できます。別の AWS アカウントからシークレットにアクセスする必要がある場合、オプションでカスタマーマネージドキーを指定してシークレットを暗号化できます。AWS Secrets Manager が提供する KMS キーを使用することもできます。

Amazon Redshift はシークレットの設定を管理し、デフォルトでシークレットを 30 日ごとにローテーションします。シークレットはいつでもを手動でローテーションできます。AWS Secrets

Manager でシークレットを管理しているプロビジョンされたクラスターまたはサーバーレス名前空間を削除すると、シークレットとそれに関連するメタデータも削除されます。

シークレットで管理された認証情報を使用してクラスターまたはサーバーレス名前空間に接続するには、Secrets Manager コンソールまたは `GetSecretValue` Secrets Manager API コールを使用して AWS Secrets Manager からシークレットを取得できます。詳細については、AWS Secrets Manager ユーザーガイドの「AWS Secrets Manager からのシークレットの取得、AWS Secrets Manager シークレットの認証情報を使用して SQL データベースに接続する」を参照してください。

AWS Secrets Manager の統合に必要となるアクセス許可

AWS Secrets Manager の統合に関連するオペレーションを実行するには、ユーザーは必要なアクセス許可を持っている必要があります。必要となる特定のリソースの API オペレーションを実行するためのアクセス許可を付与する IAM ポリシーを作成します。その後、このようなポリシーを、アクセス許可を必要とする IAM アクセス許可セットまたはロールにアタッチします。詳細については、「Amazon Redshift での Identity and Access Management」を参照してください。

Amazon Redshift が AWS Secrets Manager を使用して管理者パスワードを管理するように指定するユーザーには、次のオペレーションを実行するアクセス許可が付与されている必要があります。

- `secretsmanager:CreateSecret`
- `secretsmanager:RotateSecret`
- `secretsmanager:DescribeSecret`
- `secretsmanager:UpdateSecret`
- `secretsmanager:DeleteSecret`
- `secretsmanager:GetRandomPassword`
- `secretsmanager:TagResource`

プロビジョン済みのクラスターの `MasterPasswordSecretKmsKeyId` パラメータ、またはサーバーレス名前空間の `AdminPasswordSecretKmsKeyId` パラメータで KMS キーを渡すユーザーの場合は、上記のアクセス許可に加えて、次のアクセス許可が必要です。

- `kms:Decrypt`
- `kms:GenerateDataKey`
- `kms:CreateGrant`
- `kms:RetireGrant`

管理者パスワードシークレットのローテーション

デフォルトでは、Amazon Redshift は 30 日ごとにシークレットを自動的にローテーションして、同じ認証情報が長期間使用されることを回避できます。Amazon Redshift が管理者パスワードのシークレットをローテーションすると、AWS Secrets Manager は既存のシークレットが新しい管理者パスワードを含むように更新します。Amazon Redshift は、更新されたシークレットのパスワードと一致するようにクラスターの管理者パスワードを変更します。

AWS Secrets Manager を使用して、スケジュールされたローテーションを待つ代わりに、シークレットを直ちにローテーションできます。シークレットのローテーションの詳細については、「AWS Secrets Manager ユーザーガイド」の「AWS Secrets Manager シークレットのローテーション」を参照してください。

Amazon Redshift で AWS Secrets Manager を使用する場合の考慮事項

AWS Secrets Manager を使用してプロビジョニングされたクラスターまたはサーバーレス名前空間の管理者認証情報を管理する場合は、次の点を考慮する必要があります。

- 管理者認証情報が AWS Secrets Manager で管理されているクラスターを一時停止しても、クラスターのシークレットは削除されず、引き続きシークレットに対して料金が発生します。シークレットは、クラスターを削除したときにのみ削除されます。
- Amazon Redshift がアタッチされたシークレットのローテーションを試行する際にクラスターが一時停止していると、ローテーションは失敗します。この場合、Amazon Redshift は自動ローテーションを停止して、クラスターを再開した後でも自動ローテーションを再度試行することはありません。シークレットの自動ローテーションを継続するには、`secretsmanager:RotateSecret` API コールを使用して AWS Secrets Manager の自動ローテーションのスケジュールを再開する必要があります。
- Amazon Redshift がアタッチされたシークレットのローテーションを試行する際に、サーバーレス名前空間にワークグループが関連付けられていない場合、ローテーションは失敗し、ワークグループをアタッチした後でも自動ローテーションを再度試行することはありません。シークレットの自動ローテーションを継続するには、`secretsmanager:RotateSecret` API コールを使用して AWS Secrets Manager の自動ローテーションのスケジュールを再開する必要があります。

Amazon Redshift でのシークレットの Amazon リソースネーム (ARN) の取得

Amazon Redshift コンソールを使用して、AWS Secrets Manager が管理するシークレットの Amazon リソースネーム (ARN) を確認できます。シークレットの ARN を取得したら、AWS Secrets Manager を使用してシークレットの詳細とシークレットで暗号化されたデータの詳細を表示できます。ARN を使用したシークレットの取得の詳細については、「AWS Secrets Manager ユーザーガイド」の「シークレットの取得」を参照してください。

Amazon Redshift でプロビジョンされたクラスターのシークレットに関する詳細の表示

次の手順に従って、Amazon Redshift コンソールを使用してクラスターのシークレットの Amazon リソースネーム (ARN) を表示します。

1. AWS マネジメントコンソール にサインインして、Amazon Redshift コンソールを開きます。
2. [クラスターの概要] ペインで、シークレットを確認するクラスターを選択します。
3. [プロパティ] タブを選択します。
4. [管理者認証情報 ARN] の下で、シークレットの ARN を確認します。この ARN はシークレットの識別子です。これを使用して AWS Secrets Manager でシークレットの詳細を表示できます。

Amazon Redshift Serverless 名前空間のシークレットに関する詳細の表示

次の手順に従って、Amazon Redshift コンソールを使用してサーバーレス名前空間のシークレットの Amazon リソースネーム (ARN) を表示します。

1. AWS マネジメントコンソール にサインインして、Amazon Redshift コンソールを開きます。
2. [プロビジョニングされたクラスターダッシュボード] ダッシュボードで、ページの右上にある [サーバーレスに移動] をクリックします。
3. [サーバーレスダッシュボード] で [Namespaces / Workgroups] ペインまでスクロールして、シークレットを確認する名前空間を選択します。
4. [一般的な情報] ペインの [管理者認証情報 ARN] の下で、シークレットの ARN を確認します。この ARN はシークレットの識別子です。これを使用して AWS Secrets Manager でシークレットの詳細を表示できます。

データベース接続認証情報のシークレットの作成

Secrets Manager のシークレットを作成して、Amazon Redshift でプロビジョニングされたクラスターまたは Redshift Serverless 名前空間とワークグループへの接続に使用される認証情報を保存で

きます。Amazon Redshift クエリエディタ v2 でクエリをスケジュールするときにも、このシークレットを使用できます。

Secrets Manager コンソールを使用して Amazon Redshift でプロビジョニングされたクラスター内のデータベースのシークレットを作成するには

1. Secrets Manager のコンソール (https://console.aws.amazon.com/secretsmanager/) を開きます。
2. [シークレット] のリストに移動し、[新しいシークレットを保存] を選択します。
3. [Amazon Redshift データウェアハウスの認証情報] を選択します。シークレットを作成するには、次の手順に従って情報を入力します。
   - [ユーザー名] の [認証情報] に、データウェアハウスの管理ユーザーの名前を入力します。
   - [パスワード] の [認証情報] に、[ユーザー名] のパスワードを入力します。
   - [暗号化キー] で、暗号化キーを選択します。
   - [データウェアハウス] で、データが含まれている、Amazon Redshift でプロビジョニングされたクラスターを選択します。
   - [シークレット名] にシークレットの名前を入力します。
   - [説明] に、シークレットの説明を入力します。
   - [タグ] に、**`Redshift`** という単語を含むタグキーを入力します。このタグキーは、Amazon Redshift クエリエディタ v2 を使用してデータウェアハウスに接続する際にシークレットを一覧表示するために必要です。シークレットをマネジメントコンソールの AWS Secrets Manager の下に一覧表示するには、シークレットに文字列 **`Redshift`** で始まるタグキーが必要です。
4. シークレットに関する情報を引き続きいくつかのステップで入力してから、[レビュー] ステップで変更を保存します。

   認証情報、エンジン、ホスト、ポート、クラスター識別子の特定の値は、シークレットに保存されます。また、シークレットにはタグキー `Redshift` がタグ付けされます。

Redshift Serverless コンソールを使用して Redshift Serverless 名前空間にデータベースのシークレットを作成するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. [Redshift サーバーレス] を選択し、[名前空間の設定] に移動します。

3. シークレット認証情報を作成する名前空間を選択します。
4. [アクション] を開き、[管理者認証情報の編集] で編集を行います。
5. [管理者パスワード] で、[AWS Secrets Manager の管理者認証情報の管理] を選択します。
6. [変更を保存] を選択して、変更を保存します。

パスワードが正常に変更されたことを示すメッセージが表示されることを確認します。また、Secrets Manager コンソールでシークレットを表示することもできます。このシークレットと AWS Secrets Manager 接続方法を使用して、Redshift Serverless コンソールと Amazon Redshift クエリエディタ v2 でワークグループ内のデータベースに接続できます。クエリエディタ v2 ウェブアプリケーションでシークレットを一覧表示するには、文字列「Redshift」で始まるタグキーをシークレットに追加する必要があります。シークレットをマネジメントコンソールの AWS Secrets Manager の下に一覧表示するには、シークレットに文字列 **`Redshift`** で始まるタグキーが必要です。

Secrets Manager コンソールを使用して Redshift Serverless 名前空間にデータベースのシークレットを作成するには

1. Secrets Manager のコンソール (https://console.aws.amazon.com/secretsmanager/) を開きます。
2. [シークレット] のリストに移動し、[新しいシークレットを保存] を選択します。
3. [Amazon Redshift データウェアハウスの認証情報] を選択します。シークレットを作成するには、次の手順に従って情報を入力します。
   - [ユーザー名] の [認証情報] に、データウェアハウスの管理ユーザーの名前を入力します。
   - [パスワード] の [認証情報] に、[ユーザー名] のパスワードを入力します。
   - [暗号化キー] で、暗号化キーを選択します。
   - [データウェアハウス] で、データが含まれている Redshift Serverless 名前空間を選択します。
   - [シークレット名] にシークレットの名前を入力します。
   - [説明] に、シークレットの説明を入力します。
   - [タグ] に、**`Redshift`** という単語を含むタグキーを入力します。このタグキーは、Amazon Redshift クエリエディタ v2 を使用してデータウェアハウスに接続する際にシークレットを一覧表示するために必要です。シークレットをマネジメントコンソールの AWS Secrets Manager の下に一覧表示するには、シークレットに文字列 **`Redshift`** で始まるタグキーが必要です。

4. シークレットに関する情報を引き続きいくつかのステップで入力してから、[レビュー] ステップで変更を保存します。

   認証情報、データベース名、ホスト、ポート、名前空間、エンジンの特定の値は、シークレットに保存されます。また、シークレットにはタグキー Redshift がタグ付けされます。

AWS CLI を使用して Redshift Serverless 名前空間にデータベースのシークレットを作成するには

シークレットは、AWS CLI を使用して作成できます。1 つの方法は、次のように AWS CloudShell を使用して Secrets Manager AWS CLI コマンドを実行することです。次の手順に示す、AWS CLI コマンドを実行するための適切なアクセス許可を持っている必要があります。

1. AWS コンソールで、AWS CloudShell コマンドプロンプトを開きます。AWS CloudShell の詳細については、「AWS CloudShell ユーザーガイド」の「AWS CloudShell とは」を参照してください。
2. 例えば、MyTestSecret シークレットの場合、Secrets Manager コマンドを入力して、データベースへの接続や Amazon Redshift クエリエディタ v2 クエリのスケジュール設定に使用するシークレットを保存します。次のコマンド内の値を環境の値に置き換えます。

   - *admin* は、データウェアハウスの管理者ユーザー名です。
   - *passw0rd* は、管理者のパスワードです。
   - *dev* は、データウェアハウス内の初期データベース名です。
   - *region* は、データウェアハウスを含む AWS リージョンです。例: us-east-1。
   - *123456789012* は AWS アカウントです。
   - *namespace-id* は、c3928f0e-c889-4d2b-97a5-5738324d5d3e に似た名前空間 ID です。この ID は、Amazon Redshift コンソールの詳細ページのサーバーレス名前空間で確認できます。

```
aws secretsmanager create-secret \
--name MyTestSecret \
--description "My test secret created with the CLI." \
--secret-string "{\"username\":\"admin\",\"password\":\"passw0rd\",\"dbname\":
\"dev\",\"engine\":\"redshift\"}" \
--tags "[{\"Key\":\"redshift-serverless:namespaceArn\",\"Value\":
\"arn:aws:redshift-serverless:region:123456789012:namespace/namespace-id\"}]"
```

Amazon Redshift でのログ作成とモニタリング

モニタリングは、Amazon Redshift と AWS ソリューションの信頼性、可用性、パフォーマンスを維持する上で重要なパートです。マルチポイント障害が発生した場合は、その障害をより簡単にデバッグできるように、AWS ソリューションのすべてのパートからモニタリングデータを収集できます。AWS には、Amazon Redshift リソースをモニタリングし、潜在的なインシデントに対応するための複数のツールが用意されています。

Amazon CloudWatch アラーム

Amazon CloudWatch アラームを使用して、指定した期間中、1 つのメトリクスをモニタリングします。メトリクスが特定の閾値を超えると、Amazon SNS トピックまたは AWS Auto Scaling ポリシーに通知が送信されます。CloudWatch アラームは、特定の状態にあるという理由ではアクションを呼び出しません。状態が変わり、それが指定した期間だけ維持される必要があります。詳細については、アラームを作成する を参照してください。メトリクスの一覧は、Amazon Redshift のパフォーマンスデータをご覧ください。

AWS CloudTrail ログ

CloudTrail では、Amazon Redshift でユーザー、IAM ロール、または AWS サービスにより実行された API オペレーションのレコードを確認できます。CloudTrail で収集された情報を使用して、Amazon Redshift に対する要求、要求元の IP アドレス、要求者、要求日時などの詳細を確認できます。詳細については、「CloudTrail による口グ記録」を参照してください。

データベース監査口グ作成

Amazon Redshift は、データベースの接続とユーザーアクティビティに関する情報を記録します。これらの口グは、セキュリティとトラブルシューティング目的でのデータベースのモニタリングに役立ちます。このプロセスは、データベース監査と呼ばれます。口グは次の場所に保存できます。

- Amazon S3 バケット – これによりデータベースでのモニタリング作業を担当するユーザーはデータセキュリティ機能にアクセスできます。
- Amazon CloudWatch – 可視化機能やアクションの設定など、CloudWatch に組み込まれている機能を使用して、監査口グデータを表示できます。

> **Note**
>
> SYS_CONNECTION_LOG は、Amazon Redshift Serverless の接続ログデータを収集します。Amazon Redshift Serverless の監査ログデータを収集した場合、データはログファイルに送信できず、CloudWatch にのみ送信できることに注意してください。

トピック

- Amazon Redshift ログ
- 監査ログと Amazon CloudWatch
- 監査ログ記録の有効化
- セキュアなログ記録

Amazon Redshift ログ

Amazon Redshift は、次のログファイルに情報を記録します。

- 接続ログ – 認証試行、接続、切断をログに記録します。
- ユーザーログ – データベースのユーザー定義の変更に関する情報をログに記録します。
- ユーザーアクティビティログ – 各クエリをデータベースで実行される前にログに記録します。

接続ログとユーザーログは、主にセキュリティに役立ちます。接続ログを使用して、データベースに接続するユーザーや関連する接続情報についての情報をモニタリングできます。この情報には、ユーザーの IP アドレス、要求日時、使用した認証の種類などが含まれます。ユーザーログを使用して、データベースユーザーの定義の変更を監視できます。

ユーザーアクティビティログは、主にトラブルシューティングに役立ちます。ユーザーとシステムによってデータベースで実行されるクエリの種類についての情報を追跡します。

接続ログとユーザーログは、データベースのシステムテーブルに保存されている情報と一致します。システムテーブルを使用して同じ情報を取得できますが、ログファイルのほうがより簡単に検索と参照ができます。ログファイルは、テーブルに対してクエリを実行するために、データベースのアクセス許可ではなく Amazon S3 のアクセス許可に依存しています。また、システムテーブルに対してクエリを実行するのではなくログファイルの情報を参照するので、データベースとのやりとりによる影響が軽減されます。

> **Note**
>
> ログファイルは、システムログテーブル (STL_USERLOG と STL_CONNECTION_LOG) ほど最新ではありません。最新のレコードを含まない、それより古いレコードはログファイルにコピーされます。

> **Note**
>
> Amazon Redshift Serverless の場合、SYS_CONNECTION_LOG は接続ログデータを収集します。Amazon Redshift Serverless の監査ログデータを収集した場合、データはログファイルには送信されず、CloudWatch にのみ送信できます。

接続ログ

認証の試行、接続と切断を記録します。次の表は、接続ログの情報を示しています。これらのフィールドの詳細については、Amazon Redshift データベースデベロッパーガイドの「STL_CONNECTION_LOG」を参照してください。Amazon Redshift Serverless で収集した接続ログデータの詳細については、「SYS_CONNECTION_LOG」を参照してください。

列名	説明
event	接続または認証イベント。
recordtime	イベントが発生した時刻。
remotehost	リモートホストの名前または IP アドレス。
remoteport	リモートホストのポート番号。
pid	ステートメントに関連付けられるプロセス ID。
dbname	データベース名。
username	ユーザー名。
authmethod	認証方法。

列名	説明
duration	接続時間 (マイクロ秒)。
sslversion	Secure Sockets Layer (SSL) バージョン。
sslcipher	SSL 暗号。
mtu	最大送信単位 (MTU)。
sslcompression	SSL 圧縮タイプ。
sslexpansion	SSL 拡張タイプ。
iamauthguid	AWS CloudTrail リクエストの AWS Identity and Access Management (IAM) 認証 ID。これは、特定の接続に使用される認証情報を作成するための GetClusterCredentials API コールの識別子です。
application_name	セッションのアプリケーションの初期名または更新名。
os_version	Amazon Redshift クラスターに接続するクライアントマシン上にあるオペレーティングシステムのバージョン。
driver_version	サードパーティーの SQL クライアントツールから Amazon Redshift クラスターに接続する ODBC または JDBC ドライバーのバージョン。
plugin_name	Amazon Redshift クラスターへの接続に使用されるプラグインの名前。
protocol_version	Amazon Redshift ドライバーが、サーバーとの接続を確立する際に使用する内部プロトコルのバージョン。
sessionid	現在のセッションのグローバル一意識別子。
compression	接続に使用されている圧縮アルゴリズム。

ユーザーログ

データベースユーザーに対する次の変更の詳細のレコード。

- ユーザーの作成

- ユーザーの削除
- ユーザーの変更 (名前の変更)
- ユーザーの変更 (プロパティの変更)

列名	説明
userid	変更の影響を受けるユーザーの ID。
username	変更の影響を受けるユーザーのユーザー名。
oldusername	名前の変更アクションの場合、以前のユーザー名。その他のアクションの場合、このフィールドは空欄です。
action	実行されたアクション。有効な値: • 変更 • 作成 • 削除 • 名前の変更
usecreatedb	true (1) の場合、ユーザーにデータベースを作成する許可があることを示しています。
usesuper	true (1) の場合、ユーザーがスーパーユーザーであることを示しています。
usecatupd	true (1) の場合、ユーザーはシステムカタログを更新できることを示します。
valuntil	パスワードの有効期限。
pid	プロセス ID。
xid	トランザクション ID。
recordtime	UTC でのクエリの開始時間。

ユーザーの変更に関する追加情報を確認するには、SYS_USERLOG システムビューにクエリを実行します。このビューには、Amazon Redshift Serverless のログデータが含まれています。

ユーザーアクティビティログ

データベースで実行される前に記録した各クエリのログ。

列名	説明
recordtime	イベントが発生した時刻。
db	データベース名。
user	ユーザー名。
pid	ステートメントに関連付けられるプロセス ID。
userid	ユーザー ID
xid	トランザクション ID。
query	プレフィックス LOG の後に、改行を含むクエリのテキストが続きます。

監査ログと Amazon CloudWatch

Amazon Redshift の監査ログ作成はデフォルトではオンになっていません。クラスターでログ作成をオンにすると、Amazon Redshift は、監査ログが有効になった時点から現在までのデータをキャプチャするログを作成して Amazon CloudWatch にエクスポートするか、Amazon S3 にアップロードします。各ログの更新は、以前のログの続きとなります。

CloudWatch または Amazon S3 の監査ログ作成は、任意で、手動のプロセスです。システムテーブルへのログ作成は任意ではなく、自動的に作成されます。システムテーブルのログ作成の詳細については、Amazon Redshift データベースデベロッパーガイドのシステムテーブルのリファレンスを参照してください。

接続ログ、ユーザーログ、ユーザーアクティビティログを同時に有効にするには、AWS マネジメントコンソール、Amazon Redshift API リファレンス、AWS Command Line Interface (AWS CLI) のいずれかを使用します。ユーザーアクティビティログについては、`enable_user_activity_logging` データベースパラメータも有効にする必要があります。監査ログ作成機能のみを有効にし、関連するパラメータを有効にしない場合、データベース監査ログは接続ログとユーザーログの情報のみを記録し、ユーザーアクティビティログの情報は記録しません。この `enable_user_activity_logging` パラメータはデフォルトでは有効になっていません

(false)。ユーザーアクティビティログを有効にするには、このパラメータを true に設定します。詳細については、「Amazon Redshift パラメータグループを作成します。」を参照してください。

CloudWatch へのログ作成を有効にすると、Amazon Redshift はクラスター接続、ユーザー、およびユーザーアクティビティに関するログデータを、Amazon CloudWatch Logs のロググループにエクスポートします。ログデータは、スキーマ的には変更されません。CloudWatch はアプリケーションをモニタリングするために構築されており、リアルタイム分析を実行したり、アクションを実行するように設定したりできます。Amazon CloudWatch Logs を使用して、非常に耐久性が高いストレージにログレコードを保存できます。

CloudWatch を使用したログの表示は、Amazon S3 にログファイルを保存する代わりに推奨される代替手段です。多くの設定を必要とせず、特に他のサービスやアプリケーションのモニタリングにすでに使用している場合は、監視要件に適している可能性があります。

Amazon CloudWatch でのロググループとログイベント

エクスポートする Amazon Redshift ログを選択すると、Amazon CloudWatch Logs でログイベントをモニタリングできるようになります。Amazon Redshift Serverless のための新しいロググループは、次の (log_type がログタイプを表す) プレフィックスを使用して自動的に作成されます。

```
/aws/redshift/cluster/<cluster_name>/<log_type>
```

例えば、接続ログをエクスポートする場合、そのログデータは次のロググループに保存されます。

```
/aws/redshift/cluster/cluster1/connectionlog
```

ログイベントをロググループに対しエクスポートする際には、ログストリームが使用されます。サーバーレスエンドポイントのログイベント内で情報を検索するには、Amazon CloudWatch Logs コンソール、AWS CLI、または Amazon CloudWatch Logs API を使用します。ログデータの検索およびフィルタ処理の詳細については、「フィルターを使用したログイベントからのメトリクスの作成」を参照してください。

CloudWatch では、粒度と柔軟性を提供するクエリ構文を使用してログデータを検索できます。詳細については、「CloudWatch Logs Insights クエリ構文」を参照してください。

Amazon CloudWatch 監査ログ作成に移行する

ログを Amazon S3 に送信している場合に、CloudWatch にログを送信するなど、設定を変更しても、Amazon S3 に残っているログは影響を受けません。そのデータが格納されている Amazon S3 バケット内で引き続きログデータをクエリすることができます。

Amazon S3 でのログファイル

Amazon S3 の Amazon Redshift のログファイルの数とサイズは、クラスターのアクティビティによって大きく異なります。大量のログを生成しているアクティブなクラスターがある場合、Amazon Redshift はより頻繁にログファイルを生成することがあります。同じ時間に複数の接続ログがあるなど、同じタイプのアクティビティに対して一連のログファイルが存在する場合があります。

Amazon Redshift が Amazon S3 を使用してログを保存する場合、Amazon S3 で使用するストレージの料金が発生します。Amazon S3 にログ作成の設定を行う前に、ログファイルをどのくらいの期間保存する必要があるかのプランを必ず作成してください。この作業の一環として、監査の必要性に応じてログファイルをいつ削除またはアーカイブできるかを決定します。作成するプランは、コンプライアンス要件または規制要件に従ったデータなど、保存するデータの種類によって大きく異なります。Amazon S3 料金の詳細については、Amazon Simple Storage Service (S3) の料金を参照してください。

Amazon S3 へのログ記録を有効にする場合の制限事項

監査ログ記録には以下の制約があります。

- Amazon S3 マネージドキー (SSE-S3) 暗号化 (AES-256) のみを使用できます。
- Amazon S3 バケットでは、S3 オブジェクトロック機能をオフにする必要があります。

Amazon Redshift 監査ログ作成のためのバケットのアクセス許可

Amazon S3 へのログ作成をオンにすると、Amazon Redshift はログ作成情報を収集し、Amazon S3 に保存されたログファイルにアップロードします。新しいバケットを作成することも、既存のバケットを使用することもできます。Amazon Redshift には、バケットに対して以下の IAM アクセス許可が必要です。

- `s3:GetBucketAcl` このサービスは、Amazon S3 バケットに対して読み取りのアクセス許可が必要です。これにより、バケット所有者を識別できます。
- `s3:PutObject` このサービスは、ログをアップロードするため、put object のアクセス許可が必要です。また、ユーザーまたは IAM ロールがログ記録を有効にする場合は、Amazon S3 バケットへの `s3:PutObject` アクセス許可が必要です。ログがアップロードされるたびに、サービスは現在のバケット所有者のログ作成が有効になったときのバケット所有者と一致するかどうかを判定します。これらの所有者が一致しない場合は、エラーが発生します。

監査ログ作成を有効にするときにに新規バケットを作成するオプションを選択すると、正確なアクセス許可がバケットに適用されます。ただし、Amazon S3 で独自にバケットを作成する、または既存のバケットを使用する場合、必ずバケット名を含むバケットポリシーを追加してください。ログは、サービスプリンシパルの認証情報を使用して配信されます。ほとんどの場合のAWS リージョンでは、Redshift サービスプリンシパル名、 *redshift.amazonaws.com*です。

このバケットポリシーでは、次の形式を使用します。*ServiceName* と *BucketName*は独自の値のプレースホルダーです。バケットポリシーで、関連付けられたアクションとリソースも指定します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Put bucket policy needed for audit logging",
            "Effect": "Allow",
            "Principal": {
                "Service": "ServiceName"
            },
            "Action": [
                "s3:PutObject",
                "s3:GetBucketAcl"
            ],
            "Resource": [
                "arn:aws:s3:::BucketName",
                "arn:aws:s3:::BucketName/*"
            ]
        }
    ]
}
```

次の例は、米国東部 (バージニア北部) リージョン、および AuditLogsという名前のバケットのバケットポリシーです。

JSON

```
{
    "Version":"2012-10-17",
```

```
    "Statement": [
        {
            "Sid": "Put bucket policy needed for audit logging",
            "Effect": "Allow",
            "Principal": {
                "Service": "redshift.amazonaws.com"
            },
            "Action": [
                "s3:PutObject",
                "s3:GetBucketAcl"
            ],
            "Resource": [
                "arn:aws:s3:::AuditLogs",
                "arn:aws:s3:::AuditLogs/*"
            ]
        }
    ]
}
```

デフォルトで有効になっていないリージョン (「オプトイン」リージョンとも呼ばれます) には、リージョン固有のサービスプリンシパル名が必要です。これらの場合、サービスプリンシパル名には、`redshift.region.amazonaws.com`という形式でリージョンが含まれます。たとえば、*redshift.ap-east-1.amazonaws.com* は、アジアパシフィック (香港) リージョンの 1 つです。デフォルトで有効になっていないリージョンの一覧については、AWS 全般のリファレンスの「AWS リージョンの管理」を参照してください。

> **Note**
>
> リージョン固有のサービスプリンシパル名は、クラスターがあるリージョンに対応します。

ログファイルのベストプラクティス

Redshift が Amazon S3 にログファイルをアップロードする場合、大きなファイルを部分的にアップロードできます。マルチパートアップロードが成功しなかった場合、ファイルの一部が Amazon S3 バケットに残っている可能性があります。これにより、追加のストレージコストが発生する可能性があるため、マルチパートアップロードが失敗した場合に何が起きるかを理解することが重要です。監査ログのマルチパートアップロードの詳細については、マルチパートアップロードを使用したオブジェクトのアップロードとコピー と マルチパートアップロードの中止を参照してください。

Amazon S3 バケットの作成とバケットポリシー追加の詳細については、「Amazon Simple Storage Service ユーザーガイド」の「汎用バケットの作成」と「Amazon S3 のバケットポリシー」を参照してください。

Amazon Redshift 監査ログ作成のバケットの構造

デフォルトでは、Amazon Redshift は Amazon S3 バケット内のログファイルの整理に以下のバケットおよびオブジェクト構造 を使用します。

```
AWSLogs/AccountID/ServiceName/Region/Year/Month/Day/AccountID_ServiceName_Region
```

例: `AWSLogs/123456789012/redshift/us-east-1/2013/10/29/123456789012_redshift_us-east-1_mycluster_userlog_2013-10-29T18:01.gz`

Amazon S3 のキープレフィックスを指定すると、キーの冒頭にプレフィックスが挿入されます。

たとえば、myprefix のプレフィックスを指定する場合: `myprefix/AWSLogs/123456789012/redshift/us-east-1/2013/10/29/123456789012_redshift_us-east-1_mycluster_userlog_2013-10-29T18:01.gz`

Amazon S3 のキープレフィックスは 512 文字を超えることはできません。スペース ( )、二重引用符 (“)、一重引用符 (‘)、バックスラッシュ (\) を含めることはできません。また、許可されない特殊文字、および制御文字もいくつかあります。これらの文字の 16 進コードは次のとおりです。

- x00 から x20
- x 22
- x 27
- x5c
- x7f 以上

Amazon S3 での監査ログ記録に関する考慮事項

Amazon Redshift 監査ログ作成は、以下の理由で中断されることがあります。

- Amazon Redshift には、Amazon S3 バケットにログをアップロードするアクセス許可がありません。バケットに正しい IAM ポリシーが設定されていることを確認します。詳細については、Amazon Redshift 監査ログ作成のためのバケットのアクセス許可を参照してください。

- バケット所有者が変更されました。Amazon Redshift がログをアップロードするとき、バケット所有者がログが有効になったときと同じであることを確認します。バケット所有者を変更した場合、Amazon Redshift は、監査ログ作成に使用する別のバケットを設定するまでログをアップロードできません。
- バケットが見つかりません。Amazon S3 でバケットが削除された場合、Amazon Redshift はログをアップロードできません。バケットを再作成するか、別のバケットにログをアップロードするように Amazon Redshift を設定する必要があります。

AWS CloudTrail を使用した API コール

Amazon Redshiftは、Amazon Redshift 内のユーザー、ロール、または AWS サービスによって実行されたアクションのレコードを提供するサービスである AWS CloudTrailと統合されています。CloudTrail のすべての API コールをイベントとして Amazon Redshift にキャプチャします。Amazon Redshift と AWS CloudTrail の統合の詳細については、「CloudTrail による口グ記録」を参照してください。

CloudTrail は、Amazon Redshift データベース監査ログ作成から独立して、または追加して使用できます。

CloudTrail の詳細については、 AWS CloudTrail ユーザーガイドを参照してください。

監査ログ記録の有効化

Amazon Redshift を設定することで、監査ログデータをエクスポートできます。ログは CloudWatch にエクスポートすることも、ファイルとして Amazon S3 バケットにエクスポートすることもできます。

コンソールを使用して監査ログ作成を有効にする

コンソールの手順

クラスターに対して監査ログを有効にする

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [Clusters] (クラスター) を選択し、更新するクラスターを選択します。
3. プロパティ タブを選択します。[Database configurations] (データベース設定) パネルで、[Edit] (編集)、[Edit audit logging] (監査ログ作成を編集する) を選択します。

4. [Edit audit logging] (監査ログ作成を編集する) ページで、[Turn on] (有効化)、{S3 bucket} (S3 バケット) または [CloudWatch] を選択します。管理が簡単で、データの視覚化に役立つ機能を備えているため、CloudWatch の使用をお勧めします。
5. エクスポートするログを選択します。
6. [Save changes] (変更の保存) を選択して、変更を保存します。

セキュアなログ記録

Amazon Redshift が 1 つ以上の AWS Glue Data Catalog ビューを参照するクエリを記録すると、Amazon Redshift は特定のシステムテーブルとビュー列のフィールドを自動的にマスクし、そのクエリに関するメタデータをログに記録します。

セキュアログマスキングは、マスキング条件に合ったクエリの実行中に Amazon Redshift が生成するすべてのシステムテーブルとビューエントリに適用されます。次の表は、セキュアなログ記録が適用され、テキストは ****** で、数値は -1 でマスクされているシステムビューと列を示しています。テキストをマスクするために使用されるアスタリスクの数は、元のテキストの文字数と一致し、最大 6 文字です。6 文字を超える文字列は、6 つのアスタリスクとして表示されます。

システムテーブル	機密列
SYS_EXTERNAL_QUERY_DETAIL	列: source_type、total_partitions、qualified_partitions、scanned_files、returned_rows、returned_bytes、file_format、file_location、external_query_text、warning_message。
SYS_EXTERNAL_QUERY_ERROR	列: file_location、rowid、column_name、original_value、modified_value、trigger、action、action_value、error_code。
SYS_QUERY_DETAIL	列: step_id、step_name、table_id、table_name、input_bytes、input_rows、output_bytes、output_rows、blocks_read、blocks_write、local_read_IO、remote_read_IO、spilled_block_local_disk、spilled_block_remote_disk、step_attribute。
SYS_QUERY_HISTORY	列: returned_rows、returned_bytes。
STL_AGGR	列: rows、bytes、tbl、type。
STL_BCAST	列: rows、bytes、packets。

システムテーブル	機密列
STL_DDLTEXT	列: label、text。
STL_DELETE	列: rows、tbl。
STL_DIST	列: rows、bytes、packets。
STL_ERROR	列: file、linenum、context、error。
STL_EXPLAIN	列: plannode、info。
STL_FILE_SCAN	列: name、line、bytes。
STL_HASH	列: rows、bytes、tbl、est_rows。
STL_HASHJOIN	列: rows、tbl、num_parts、join_type。
STL_INSERT	列: rows、tbl。
STL_LIMIT	列: rows。
STL_MERGE	列: rows。
STL_MERGEJOIN	列: rows、tbl。
STL_NESTLOOP	列: rows、tbl。
STL_PARSE	列: rows。
STL_PLAN_INFO	列: startupcost、totalcost、rows、bytes。
STL_PROJECT	列: rows、tbl。
STL_QUERY	列: querytxt。
STL_QUERY_METRICS	列: max_rows、rows、max_blocks_read、blocks_read、max_blocks_to_disk、blocks_to_disk、max_query_scan_size、query_scan_size。
STL_QUERYTEXT	列: text。
STL_RETURN	列: rows、bytes。

システムテーブル	機密列
STL_S3CLIENT	列: bucket、key、transfer_size、data_size。
STL_S3CLIENT_ERROR	列: bucket、key、error、transfer_size。
STL_SAVE	列: rows、bytes、tbl。
STL_SCAN	列: rows、bytes、fetches、type、tbl、rows_pre_filter、rows_pre_user_filter、perm_table_name、scanned_mega_value。
STL_SORT	列: rows、bytes、tbl。
STL_SSHCLIENT_ERROR	列: ssh_username、endpoint、command、error。
STL_TR_CONFLICT	列: table_id。
STL_UNDONE	列: table_id。
STL_UNIQUE	列: rows、type、bytes。
STL_UTILITYTEXT	列: label、text。
STL_WINDOW	列: rows。
STV_BLOCKLIST	列: col、tbl、num_values、minvalue、maxvalue。
STV_EXEC_STATE	列: rows、bytes、label。
STV_INFLIGHT	列: label、text。
STV_LOCKS	列: table_id。
STV_QUERY_METRICS	列: rows、rowsmax_rows、blocks_read、max_blocks_read、max_blocks_to_disk、blocks_to_disk、max_query_scan_size、query_scan_size。

システムテーブル	機密列
STV_START_UP_RECOVERY_STATE	列: table_id、table_name。
STV_TBL_PERM	列: ID、name、rows、sorted_rows、temp、block_count、query_scan_size。
STV_TBL_TRANS	列: id、rows、size。
SVCS_EXPLAIN	列: plannode、info。
SVCS_PLAN_INFO	列: rows、bytes。
SVCS_QUERY_SUMMARY	列: step、rows、bytes、rate_row、rate_byte、label、rows_pre_filter。
SVCS_S3LIST	列: bucket、prefix、retrieved_files、max_file_size、avg_file_size。
SVCS_S3LOG	列: message。
SVCS_S3PARTITION_SUMMARY	列: total_partitions、qualified_partitions、min_assigned_partitions、max_assigned_partitions、avg_assigned_partitions。
SVCS_S3QUERY_SUMMARY	列: external_table_name、file_format、s3_scanned_rows、s3_scanned_bytes、s3query_returned_rows、s3query_returned_bytes。
SVL_QUERY_METRICS	列: step_label、Scan_row_count、join_row_count、nested_loop_join_row_count、return_row_count、spectal_scan_row_count、spectal_scan_size_mb。
SVL_QUERY_METRICS_SUMMARY	列: step_label、Scan_row_count、join_row_count、nested_loop_join_row_count、return_row_count、spectal_scan_row_count、spectal_scan_size_mb。
SVL_QUERY_REPORT	列: rows、bytes、label、rows_pre_filter。
SVL_QUERY_SUMMARY	列: rows、bytes、rows_pre_filter。

システムテーブル	機密列
SVL_S3LIST	列: bucket、prefix、retrieved_files、max_file_size、avg_file_size。
SVL_S3LOG	列: message。
SVL_S3PARTITION	列: rows、bytes、label、rows_pre_filter。
SVL_S3PARTITION_SUMMARY	列: total_partitions、qualified_partitions、min_assigned_partitions、max_assigned_partitions、avg_assigned_partitions。
SVL_S3QUERY	列: external_table_name、file_format、s3_scanned_rows、s3_scanned_bytes、s3query_returned_rows、s3query_returned_bytes、files。
SVL_S3QUERY_SUMMARY	列: external_table_name、file_format、s3_scanned_rows、s3_scanned_bytes、s3query_returned_rows、s3query_returned_bytes。
SVL_S3RETRIES	列: file_size、location、message。
SVL_SPECTRUM_SCAN_ERROR	列: location、rowid、colname、original_value、modified_value、trigger、action、action_value、error_code。
SVL_STATEMENTTEXT	列: type、text。
SVL_STORED_PROC_CALL	列: querytxt。
SVL_STORED_PROC_MESSAGES	列: message、linenum、querytext。
SVL_UDF_LOG	列: message、funcname。
SVV_DISKUSAGE	列: name、col、tbl、blocknum、num_values、minvalue、maxvalue。
SVV_QUERY_STATE	列: rows、bytes、label。
SVV_TABLE_INFO	列: table_id、table。
SVV_TRANSACTIONS	列: relation。

システムテーブルとビューの詳細については、「Amazon Redshift データベース管理者ガイド」の「システムテーブルとビューのリファレンス」を参照してください。Amazon Redshift がクエリ結果を動的にマスクする機能の詳細については、「Amazon Redshift データベース開発者ガイド」の「動的データマスキング」を参照してください。AWS Glue Data Catalog で Amazon Redshift を使用してビューを作成する方法については、「Amazon Redshift データベース開発者ガイド」の「AWS Glue Data Catalog のビュー」を参照してください。

CloudTrail によるログ記録

Amazon Redshift、データ共有、Amazon Redshift サーバーレス、Amazon Redshift Data API、クエリエディタ v2 のすべてが AWS CloudTrail と統合されています。CloudTrail は、Amazon Redshift のユーザー、ロール、または AWS サービスによって実行されたアクションを記録するサービスです。CloudTrail のすべての API コールをイベントとして Amazon Redshift にキャプチャします。キャプチャされたコールには、Amazon Redshift コンソールからのコールと、Redshift オペレーションへのコードコールが含まれます。

CloudTrail 証跡を作成する場合は、Amazon S3 バケットに対する CloudTrail イベント (Redshift のイベントなど) の継続的デリバリーを使用できます。証跡を設定しない場合でも、CloudTrail コンソールの [Event history (イベント履歴)] で最新のイベントを表示できます。CloudTrail によって収集されたデータを使用して、多くの情報を判断できます。これには、Redshift に対するリクエスト、リクエスト元の IP アドレス、リクエスト者、リクエスト日時などの詳細を含みます。

CloudTrail は、Amazon Redshift データベース監査ログ作成から独立して、または追加して使用できます。

CloudTrail の詳細については、 AWS CloudTrail ユーザーガイドを参照してください。

CloudTrail での情報

CloudTrail は、アカウント作成時に AWS で有効になります。アクティビティが発生すると、そのアクティビティは、[Event history] (イベント履歴) の他の AWS サービスイベントとともに、CloudTrail イベントに記録されます。最近のイベントは、AWSアカウントで表示、検索、ダウンロードできます。詳細については、AWS CloudTrailユーザーガイドの「CloudTrail イベント履歴でのイベントの表示」を参照してください。

AWS アカウントのイベント (Redshift のイベントなど) を継続的に記録するには、証跡を作成します。CloudTrail は、証跡を使用してログファイルを Amazon S3 バケットに配信します。デフォルトでは、コンソールで証跡を作成すると、すべての AWS リージョンに証跡が適用されます。証跡は、AWSパーティションのすべてのリージョンからのイベントをログに記録し、指定した Amazon S3 バケットにログファイルを配信します。さらに、その他の AWS サービスを設定して、CloudTrail

ログで収集したデータをより詳細に分析し、それに基づく対応を行うことができます。詳細については、AWS CloudTrailユーザーガイドで次を参照してください。

- 証跡の作成のための概要
- CloudTrail がサポートするサービスと統合
- CloudTrail 用 Amazon SNS 通知の構成
- 複数のリージョンから CloudTrail ログファイルを受け取るおよび複数のアカウントから CloudTrail ログファイルを受け取る

Amazon Redshift、Amazon Redshift サーバーレス、Data API、データ共有、クエリエディタ v2 のすべてのアクションは、CloudTrail によってログに記録されます。例えば、`AuthorizeDatashare`、`CreateNamespace`、`ExecuteStatement`、`CreateConnection` の各アクションを呼び出すと、CloudTrail ログファイルにエントリが生成されます。

各イベントまたはログエントリには、リクエストの生成者に関する情報が含まれます。ID 情報は次の判断に役立ちます。

- リクエストが、ルートとユーザー認証情報のどちらを使用して送信されたか。
- リクエストが、ロールとフェデレーションユーザーの一時的なセキュリティ認証情報のどちらを使用して送信されたか。
- リクエストが、別の AWS サービスによって送信されたかどうか。

詳細については、「AWS CloudTrail ユーザーガイド」の「CloudTrail userIdentity Element」を参照してください。

ログファイルエントリ

証跡は、指定した Amazon S3 バケットに対して、イベントをログファイルとして配信できるようにする設定です。CloudTrail のログファイルは、単一か複数のログエントリを含みます。イベントは、任意の出典からの単一のリクエストを表し、リクエストされたアクション、アクションの日時、リクエストパラメータなどに関する情報が含まれます。CloudTrail ログファイルは、パブリック API 呼び出しの順序付けられたスタックトレースではないため、特定の順序では表示されません。

Amazon Redshift データ共有の例

以下の例は、 `AuthorizeDataShare` オペレーションを説明する CloudTrail ログエントリを示しています。

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AKIAIOSFODNN7EXAMPLE:janedoe",
        "arn": "arn:aws:sts::111122223333:user/janedoe",
        "accountId": "111122223333",
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AKIAIOSFODNN7EXAMPLE:janedoe",
                "arn": "arn:aws:sts::111122223333:user/janedoe",
                "accountId": "111122223333",
                "userName": "janedoe"
            },
            "attributes": {
                "creationDate": "2021-08-02T23:40:45Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2021-08-02T23:40:58Z",
    "eventSource": "redshift.amazonaws.com",
    "eventName": "AuthorizeDataShare",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "3.227.36.75",
    "userAgent":"aws-cli/1.18.118 Python/3.6.10
 Linux/4.9.217-0.1.ac.205.84.332.metal1.x86_64 botocore/1.17.41",
    "requestParameters": {
        "dataShareArn": "arn:aws:redshift:us-
east-1:111122223333:datashare:4c64c6ec-73d5-42be-869b-b7f7c43c7a53/testshare",
        "consumerIdentifier": "555555555555"
    },
    "responseElements": {
        "dataShareArn": "arn:aws:redshift:us-
east-1:111122223333:datashare:4c64c6ec-73d5-42be-869b-b7f7c43c7a53/testshare",
        "producerNamespaceArn": "arn:aws:redshift:us-
east-1:123456789012:namespace:4c64c6ec-73d5-42be-869b-b7f7c43c7a53",
        "producerArn": "arn:aws:redshift:us-
east-1:111122223333:namespace:4c64c6ec-73d5-42be-869b-b7f7c43c7a53",
        "allowPubliclyAccessibleConsumers": true,
        "dataShareAssociations": [
```

```
            {
                "consumerIdentifier": "555555555555",
                "status": "AUTHORIZED",
                "createdDate": "Aug 2, 2021 11:40:56 PM",
                "statusChangeDate": "Aug 2, 2021 11:40:57 PM"
            }
        ]
    },
    "requestID": "87ee1c99-9e41-42be-a5c4-00495f928422",
    "eventID": "03a3d818-37c8-46a6-aad5-0151803bdb09",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Amazon Redshift サーバーレスの例

Amazon Redshift サーバーレスは、AWS CloudTrail との統合により、Amazon Redshift サーバーレスで実行されたアクションの記録を提供します。CloudTrail のすべての API コールをイベントとして Amazon Redshift Serverless にキャプチャします。Amazon Redshift サーバーレスの機能の詳細については、「Amazon Redshift サーバーレス機能の概要」を参照してください。

以下の例は、`CreateNamespace` アクションを示す CloudTrail ログエントリです。

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AAKEOFPINEXAMPLE:admin",
        "arn": "arn:aws:sts::111111111111:assumed-role/admin/admin",
        "accountId": "111111111111",
        "accessKeyId": "AAKEOFPINEXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AAKEOFPINEXAMPLE",
                "arn": "arn:aws:iam::111111111111:role/admin",
                "accountId": "111111111111",
                "userName": "admin"
            },
            "webIdFederationData": {},
```

```
            "attributes": {
                "creationDate": "2022-03-21T20:51:58Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2022-03-21T23:15:40Z",
    "eventSource": "redshift-serverless.amazonaws.com",
    "eventName": "CreateNamespace",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "56.23.155.33",
    "userAgent": "aws-cli/2.4.14 Python/3.8.8 Linux/5.4.181-109.354.amzn2int.x86_64
 exe/x86_64.amzn.2 prompt/off command/redshift-serverless.create-namespace",
    "requestParameters": {
        "adminUserPassword": "HIDDEN_DUE_TO_SECURITY_REASONS",
        "adminUsername": "HIDDEN_DUE_TO_SECURITY_REASONS",
        "dbName": "dev",
        "namespaceName": "testnamespace"
    },
    "responseElements": {
        "namespace": {
            "adminUsername": "HIDDEN_DUE_TO_SECURITY_REASONS",
            "creationDate": "Mar 21, 2022 11:15:40 PM",
            "defaultIamRoleArn": "",
            "iamRoles": [],
            "logExports": [],
            "namespaceArn": "arn:aws:redshift-serverless:us-
east-1:111111111111:namespace/befa5123-16c2-4449-afca-1d27cb40fc99",
            "namespaceId": "8b726a0c-16ca-4799-acca-1d27cb403599",
            "namespaceName": "testnamespace",
            "status": "AVAILABLE"
        }
    },
    "requestID": "ed4bb777-8127-4dae-aea3-bac009999163",
    "eventID": "1dbee944-f889-4beb-b228-7ad0f312464",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111111111111",
    "eventCategory": "Management",
}
```

Amazon Redshift Data API の例

以下の例は、ExecuteStatement アクションを示す CloudTrail ログエントリです。


```
{
    "eventVersion":"1.05",
    "userIdentity":{
        "type":"IAMUser",
        "principalId":"AKIAIOSFODNN7EXAMPLE:janedoe",
        "arn":"arn:aws:sts::123456789012:user/janedoe",
        "accountId":"123456789012",
        "accessKeyId":"AKIAI44QH8DHBEXAMPLE",
        "userName": "janedoe"
    },
    "eventTime":"2020-08-19T17:55:59Z",
    "eventSource":"redshift-data.amazonaws.com",
    "eventName":"ExecuteStatement",
    "awsRegion":"us-east-1",
    "sourceIPAddress":"192.0.2.0",
    "userAgent":"aws-cli/1.18.118 Python/3.6.10
 Linux/4.9.217-0.1.ac.205.84.332.metal1.x86_64 botocore/1.17.41",
    "requestParameters":{
        "clusterIdentifier":"example-cluster-identifier",
        "database":"example-database-name",
        "dbUser":"example_db_user_name",
        "sql":"***OMITTED***"
    },
    "responseElements":{
        "clusterIdentifier":"example-cluster-identifier",
        "createdAt":"Aug 19, 2020 5:55:58 PM",
        "database":"example-database-name",
        "dbUser":"example_db_user_name",
        "id":"5c52b37b-9e07-40c1-98de-12ccd1419be7"
    },
    "requestID":"00c924d3-652e-4939-8a7a-cd0612eeb8ac",
    "eventID":"c1fb7076-102f-43e5-9ec9-40820bcc1175",
    "readOnly":false,
    "eventType":"AwsApiCall",
    "recipientAccountId":"123456789012"
}
```


べき等性に使用される clientToken を示す ExecuteStatement アクションを示した CloudTrail ログエントリの例を次に示します。

```
{
    "eventVersion":"1.05",
    "userIdentity":{
        "type":"IAMUser",
        "principalId":"AKIAIOSFODNN7EXAMPLE:janedoe",
        "arn":"arn:aws:sts::123456789012:user/janedoe",
        "accountId":"123456789012",
        "accessKeyId":"AKIAI44QH8DHBEXAMPLE",
        "userName": "janedoe"
    },
    "eventTime":"2020-08-19T17:55:59Z",
    "eventSource":"redshift-data.amazonaws.com",
    "eventName":"ExecuteStatement",
    "awsRegion":"us-east-1",
    "sourceIPAddress":"192.0.2.0",
    "userAgent":"aws-cli/1.18.118 Python/3.6.10
 Linux/4.9.217-0.1.ac.205.84.332.metal1.x86_64 botocore/1.17.41",
    "requestParameters":{
        "clusterIdentifier":"example-cluster-identifier",
        "database":"example-database-name",
        "dbUser":"example_db_user_name",
        "sql":"***OMITTED***",
        "clientToken":"32db2e10-69ac-4534-b3fc-a191052616ce"
    },
    "responseElements":{
        "clusterIdentifier":"example-cluster-identifier",
        "createdAt":"Aug 19, 2020 5:55:58 PM",
        "database":"example-database-name",
        "dbUser":"example_db_user_name",
        "id":"5c52b37b-9e07-40c1-98de-12ccd1419be7"
    },
    "requestID":"00c924d3-652e-4939-8a7a-cd0612eeb8ac",
    "eventID":"c1fb7076-102f-43e5-9ec9-40820bcc1175",
    "readOnly":false,
    "eventType":"AwsApiCall",
    "recipientAccountId":"123456789012"
}
```


Amazon Redshift クエリエディタ v2 の例

以下の例は、CreateConnection アクションを示す CloudTrail ログエントリです。


```
{
```

```
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AAKEOFPINEXAMPLE:session",
        "arn": "arn:aws:sts::123456789012:assumed-role/MyRole/session",
        "accountId": "123456789012",
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AAKEOFPINEXAMPLE",
                "arn": "arn:aws:iam::123456789012:role/MyRole",
                "accountId": "123456789012",
                "userName": "MyRole"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-09-21T17:19:02Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2022-09-21T22:22:05Z",
    "eventSource": "sqlworkbench.amazonaws.com",
    "eventName": "CreateConnection",
    "awsRegion": "ca-central-1",
    "sourceIPAddress": "192.2.0.2",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:102.0)
 Gecko/20100101 Firefox/102.0",
    "requestParameters": {
        "password": "***",
        "databaseName": "***",
        "isServerless": false,
        "name": "***",
        "host": "redshift-cluster-2.c8robpbxvbf9.ca-central-1.redshift.amazonaws.com",
        "authenticationType": "***",
        "clusterId": "redshift-cluster-2",
        "username": "***",
        "tags": {
            "sqlworkbench-resource-owner": "AAKEOFPINEXAMPLE:session"
        }
    },
    "responseElements": {
        "result": true,
```

```
        "code": "",
        "data": {
            "id": "arn:aws:sqlworkbench:ca-central-1:123456789012:connection/ce56b1be-
dd65-4bfb-8b17-12345123456",
            "name": "***",
            "authenticationType": "***",
            "databaseName": "***",
            "secretArn": "arn:aws:secretsmanager:ca-
central-1:123456789012:secret:sqlworkbench!7da333b4-9a07-4917-b1dc-12345123456-qTCoFm",
            "clusterId": "redshift-cluster-2",
            "dbUser": "***",
            "userSettings": "***",
            "recordDate": "2022-09-21 22:22:05",
            "updatedDate": "2022-09-21 22:22:05",
            "accountId": "123456789012",
            "tags": {
                "sqlworkbench-resource-owner": "AAKEOFPINEXAMPLE:session"
            },
            "isServerless": false
        }
    },
    "requestID": "9b82f483-9c03-4cdd-bb49-a7009e7da714",
    "eventID": "a7cdd442-e92f-46a2-bc82-2325588d41c3",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management"
}
```


AWS CloudTrail ログの Amazon Redshift アカウント ID

Amazon Redshift が別の AWS のサービスを呼び出す際、その呼び出しは、Amazon Redshift に属するアカウント ID で記録されます。自分のアカウント ID では記録されません。たとえば、Amazon Redshift が、クラスターでの暗号化を管理するために `CreateGrant`、 `Decrypt`、 `Encrypt`、 `RetireGrant` などの AWS Key Management Service (AWS KMS) オペレーションを呼び出すとします。この場合、この呼び出しは AWS CloudTrail により、Amazon Redshift アカウント ID を使用して記録されます。

Amazon Redshift が他の AWS サービスを呼び出す際、次の表にあるアカウント ID が使用されます。

リージョン	リージョン	アカウント ID
米国東部 (バージニア州北部) リージョン	us-east-1	368064434614
米国東部(オハイオ州)リージョン	us-east-2	790247189693
米国西部 (北カリフォルニア) リージョン	us-west-1	703715109447
米国西部 (オレゴン州) リージョン	us-west-2	473191095985
アフリカ (ケープタウン) リージョン	af-south-1	420376844563
アジアパシフィック (香港) リージョン	ap-east-1	651179539253
アジアパシフィック (ハイデラバード) リージョン	ap-south-2	297058826802
アジアパシフィック (ジャカルタ) リージョン	ap-southeast-3	623197973179
アジアパシフィック (マレーシア) リージョン	ap-southeast-5	590184011157
アジアパシフィック (メルボルン) リージョン	ap-southeast-4	945512339897
アジアパシフィック (ムンバイ) リージョン	ap-south-1	408097707231
アジアパシフィック (ニュージーランド)	ap-southeast-6	730335362089

リージョン	リージョン	アカウント ID
アジアパシフィック (大阪) リージョン	ap-northeast-3	398671365691
アジアパシフィック (ソウル) リージョン	ap-northeast-2	713597048934
アジアパシフィック (シンガポール) リージョン	ap-southeast-1	960118270566
アジアパシフィック (シドニー) リージョン	ap-southeast-2	485979073181
アジアパシフィック (台北)	ap-east-2	211125526206
アジアパシフィック (タイ) リージョン	ap-southeast-7	767397930036
アジアパシフィック (東京) リージョン	ap-northeast-1	615915377779
カナダ (中部) リージョン	ca-central-1	764870610256
カナダ西部 (カルガリー)	ca-west-1	830903446466
欧州 (フランクフルト) リージョン	eu-central-1	434091160558
欧州 (アイルランド) リージョン	eu-west-1	246478207311
欧州 (ロンドン) リージョン	eu-west-2	885798887673
欧州 (ミラノ) リージョン	eu-south-1	041313461515
欧州 (パリ) リージョン	eu-west-3	694668203235
欧州 (スペイン) リージョン	eu-south-2	028811157404

リージョン	リージョン	アカウント ID
欧州 (ストックホルム) リージョン	eu-north-1	553461782468
欧州 (チューリッヒ) リージョン	eu-central-2	668912161003
イスラエル (テルアビブ) リージョン	il-central-1	901883065212
メキシコ (中部) リージョン	mx-central-1	058264411980
中東 (バーレーン) リージョン	me-south-1	051362938876
中東 (アラブ首長国連邦) リージョン	me-central-1	595013617770
南米 (サンパウロ) リージョン	sa-east-1	392442076723

次の例は、Amazon Redshift から呼び出された AWS KMS Decrypt オペレーションの CloudTrail ログエントリを示しています。

```
{

    "eventVersion": "1.05",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAI5QPCMKLTL4VHFCYY:i-0f53e22dbe5df8a89",
        "arn": "arn:aws:sts::790247189693:assumed-role/prod-23264-role-wp/
i-0f53e22dbe5df8a89",
        "accountId": "790247189693",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2017-03-03T16:24:54Z"
            },
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAI5QPCMKLTL4VHFCYY",
                "arn": "arn:aws:iam::790247189693:role/prod-23264-role-wp",
```

```
                "accountId": "790247189693",
                "userName": "prod-23264-role-wp"
            }
        }
    },
    "eventTime": "2017-03-03T17:16:51Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "52.14.143.61",
    "userAgent": "aws-internal/3",
    "requestParameters": {
        "encryptionContext": {
            "aws:redshift:createtime": "20170303T1710Z",
            "aws:redshift:arn": "arn:aws:redshift:us-east-2:123456789012:cluster:my-dw-
instance-2"
        }
    },
    "responseElements": null,
    "requestID": "30d2fe51-0035-11e7-ab67-17595a8411c8",
    "eventID": "619bad54-1764-4de4-a786-8898b0a7f40c",
    "readOnly": true,
    "resources": [
        {
            "ARN": "arn:aws:kms:us-east-2:123456789012:key/f8f4f94f-e588-4254-
b7e8-078b99270be7",
            "accountId": "123456789012",
            "type": "AWS::KMS::Key"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012",
    "sharedEventID": "c1daefea-a5c2-4fab-b6f4-d8eaa1e522dc"

}
```


S3 Tables とのシステムテーブル統合

Amazon S3 Tables とのシステムテーブル統合を使用して、Amazon Redshift システムテーブルデータを 7 日間のクラスター内保持期間を超えて保持します。これにより、コンプライアンス、監査、オブザーバビリティの要件を満たすことができます。この機能を有効にすると、Redshift はシステムテーブルログを Apache Iceberg 形式で S3 Tables に書き込み、パーティショニング、圧縮、ス

ナップショットのメンテナンスを管理します。このデータは、分析と監査に必要な期間保持できます。データはオープンな Iceberg 形式で保存されるため、Redshift、Amazon Athena、またはその他の Iceberg 互換エンジンを使用してクエリを実行できます。また、自然言語クエリに AI エージェントスキルを使用することもできます (GitHub ウェブサイトの「System table skills for AWS Agent Toolkit」を参照)。

コンソール、Redshift CLI、または SDK を使用してこの機能を有効にすると、Redshift はその後すぐに、選択したシステムテーブルから S3 Tables への新しい行の書き込みを一定の頻度で開始します。発行するシステムテーブルはユーザーが選択します。S3 Tables は、ユーザーが設定した保持期間を適用します。

以下のユースケースがサポートされます。

- コンプライアンスおよび監査要件を満たす。クエリ、接続、変更履歴を数か月または数年間保持します。
- フリートを 1 か所からモニタリングする。アカウントとリージョンにおける複数のデータウェアハウスのモニタリングデータを統合し、フリート全体のアクティビティを分析します。
- カスタムの抽出、変換、ロード (ETL) パイプラインを不要にする。システムテーブルデータを S3 にエクスポートして長期保存するためのジョブの構築と保守を停止します。
- 過去のインシデントを調査する。根本原因分析、インシデント後のレビュー、ワークロードの変化とパフォーマンス傾向の経時的なモニタリングのために、クラスター内の 7 日間のウィンドウを超えて履歴データをクエリします。

この機能は、Redshift および S3 Tables がサポートされているすべての AWS 商用リージョンにおいて、Amazon Redshift Provisioned RA3 および RG インスタンスと Amazon Redshift Serverless で使用できます。このトピックにおいて、データウェアハウスとは、プロビジョニングされたクラスターまたは Redshift Serverless ワークグループのいずれかを指します。

トピック

- 仕組み
- S3 Tables とのシステムテーブル統合の有効化
- サポートされているシステムテーブル
- 各テーブルに追加されたメタデータ列
- 保持の設定
- データガバナンス
- 配信とライフサイクルの動作

- ベストプラクティス
- 料金
- 考慮事項と制限事項

仕組み

機能を有効にし、サポートされている 1 つ以上のシステムテーブルを選択すると、Redshift はそれらのシステムテーブルから新しく完了したレコードを S3 Tables に書き込みます。データは固定頻度でバッチ配信されます。

システムテーブルは AWS アカウントに存在しますが、サービスマネージド型です。AWS は、以下を含むデータ配信とテーブル管理を所有および運用します。

- アカウント内に S3 テーブルバケット、名前空間、S3 テーブルを作成します。
- テーブルスキーマを定義および更新します。
- Apache Iceberg 形式のデータを S3 テーブルに書き込みます。
- データを圧縮し、スナップショットを維持します (S3 Tables によって自動的に処理されます)。

構築または保守するインフラストラクチャはなく、ワークロードへの影響もありません。

S3 Tables のレコードの有効期限を通じて、データの保持期間を設定します。詳細については、「the section called “保持の設定”」を参照してください。

テーブルは AWS によって管理されるため、Redshift が配信する行を変更または削除することはできません。配信されるデータはイミュータブルです。テーブルへの読み取りアクセスのみを制御します。詳細については、「the section called “データガバナンス”」を参照してください。

完了したアクティビティのみが S3 Tables に配信されます。データウェアハウスのシステムテーブルには処理中のアクティビティが表示されることがありますが、最終状態に達したレコードのみがコピーされます。例えば、完了、中止、またはキャンセルされたクエリは配信されますが、まだ実行中のクエリは最終状態になるまで配信されません。

S3 Tables とのシステムテーブル統合の有効化

S3 Tables とのシステムテーブル統合を有効にするには、新しいログ送信先タイプ `s3table` を指定して既存の Redshift API を使用します。有効にすると、Redshift はアカウントに `aws-redshift` という名前の S3 テーブルバケットを作成します。この機能は、他のログ送信先に使用するの

と同じオペレーションで管理します。プロビジョニングされたクラスターの場合は enable-logging、disable-logging、describe-logging-status を使用し、Redshift Serverless の場合は update-namespace と get-namespace を使用します。

アクセス許可

機能を有効化、変更、または無効化するプリンシパルには、以下のアクセス許可が必要です。

- redshift:EnableLogging (プロビジョニングされたクラスターの場合) または redshift-serverless:UpdateNamespace (Redshift Serverless の場合)
- S3 テーブルバケットを作成および設定するためのアクセス許可。有効にするプリンシパルには、以下の S3 Tables アクセス許可が必要です。
  - s3tables:CreateTableBucket
  - s3tables:PutTableBucketEncryption
  - s3tables:PutTableBucketPolicy

  この機能を有効にすると、Redshift はオペレーションをトリガーするプリンシパルの ID を使用して S3 テーブルバケットを作成および設定します。

バケットが作成されると、Redshift は Redshift と S3 Tables 間のサービス信頼関係を使用して、バケット内に名前空間とテーブルを作成します。有効にするプリンシパルには、名前空間やテーブルを作成するためのアクセス許可は不要です。

デプロイオプション

配信は、2 つのパターンのいずれかで設定できます。特定のデータウェアハウスは、一度に 1 つのパターンを使用します。

デプロイオプション	説明
ウェアハウスごと	Redshift は、各データウェアハウスのシステムテーブルデータを、それぞれ専用の S3 テーブルセットに書き込みます。テーブル内のすべての行は、単一のデータウェアハウスから取得されます。これは、ウェアハウス間の物理的な分離に使用します。
統合	Redshift は、同じアカウントおよびリージョン内の複数のデータウェアハウスからシステムテーブルデータを S3 テーブルの共有セットに書き込みます。異なるウェアハウスからの行は、warehouse_namespac

デプロイオプション	説明
	e_arn 列と warehouse_name 列で区別されます。これは、一元化されたクロスウェアハウス分析に使用します。

デプロイモデルは、S3 Tables でのデータの構成方法を決定します。ウェアハウスごとのモデルでは、S3 Tables 名前空間名にデータウェアハウスの一意の識別子が含まれているため、各ウェアハウスのデータは物理的に独自の名前空間に分割されます。統合モデルでは、S3 Tables の名前空間名に AWS アカウント番号が含まれるため、そのアカウントおよびリージョン内におけるすべてのウェアハウスのデータは同じ名前空間とテーブルにまとめて保存されます。

どちらのオプションも、単一の AWS アカウントおよび単一の AWS リージョンでサポートされます。リージョン間またはアカウント間をまたいでデータを分析するには、クエリ実行時に各リージョンまたはアカウントのテーブルの結果を結合します。クロスアカウントアクセスでは、AWS Glue データカタログ共有を使用できます。詳細については、「AWS Glue デベロッパーガイド」の「クロスアカウントアクセス許可の付与」を参照してください。

> **Note**
>
> データウェアハウスを、あるデプロイパターンから別のデプロイパターンに切り替えると、以前に配信されたデータは既存のテーブルに残り、設定した保持期間にわたって保持されます。Redshift は新しいターゲットに新しいデータの配信を開始し、履歴データはバックフィルしません。

コンソールを使用する

S3 Tables とのシステムテーブル統合を設定するには

1. Redshift コンソールを開き、ナビゲーションペインの [Amazon Redshift] または [Amazon Redshift Serverless] で、[System table integrations] を選択します。クラスターまたは名前空間の詳細ページから、[統合] タブを選択するか、[アクション]、[統合]、[Configure system table integration] の順に選択して設定することもできます。
2. [Create System table integration] を選択します。
3. プロビジョニングされたクラスターまたは Redshift Serverless ワークグループを選択します。
4. 発行するシステムテーブルを選択します。個々の SYS_* ビューを選択するか、[Select all supported system tables] を選択して、現在および将来のすべてのサポート対象ビューを発行し

ます。すべてを選択すると、将来追加される新しいビューが、設定を変更することなく自動的に含まれます。

5. デプロイモデルを選択します。

   - データウェアハウスごとにシステムテーブルごとの個別の S3 テーブル。このウェアハウスのデータを専用のテーブルセットに保持します。
   - データウェアハウス間でシステムテーブルごとに共有 S3 テーブル。アカウント内の複数のウェアハウスのデータを共有テーブルセットに統合します。

6. 必要に応じて、暗号化キーを選択します。デフォルトでは、データは Amazon S3 マネージドキー (SSE-S3) で暗号化されます。独自の AWS KMS キーを使用するには、ここで選択します。独自の AWS KMS キーを使用する場合は、キーポリシーで Redshift と S3 Tables にそのキーへのアクセスを許可する必要があります。詳細については、「the section called “暗号化の設定”」を参照してください。
7. 変更内容を保存します。Redshift は、選択したシステムテーブルを S3 Tables に発行し始め、継続的に新しいレコードを追加していきます。

システムテーブルの発行を停止するには、設定に戻って削除します。既に発行されたデータは、S3 Tables で有効期限を設定するまで保持されます。

クラスターまたは名前空間の詳細ページから統合のステータスを確認できます。[View S3 table mapping] を展開すると、各システムテーブルおよび対応する S3 テーブルの名前および名前空間のマッピングを確認できます。S3 Tables コンソールで発行済みデータを確認することもできます。

AWS CLI の使用

以下のパラメータは、S3 Tables への発行を設定します。

パラメータ	適用対象	説明
`--log-destination-type`	両方	S3 Tables に発行するには、`s3table` に設定します。その他の値は `s3` と `cloudwatch` です。
`--log-exports`	プロビジョニング	発行するシステムテーブル。`SYS_*` ビュー名のリスト、または `all` として指定します。
`--s3-table-names`	Serverless	発行するシステムテーブル。`SYS_*` ビュー名のリスト、または `all` として指定します。

パラメータ	適用対象	説明
--s3-table-action	Serverless	S3 Tables への発行の Enable または Disable。
--s3-table-granularity	両方	テーブルスコープ。プロビジョニング: cluster (デフォルト) または account。Serverless: namespace (デフォルト) または account。
--s3-table-kms-key-id	両方	暗号化用の AWS KMS キーの ARN または ID (オプション)。デフォルトは Amazon S3 マネージドキー (SSE-S3) です。

プロビジョニングされたクラスター

配信を有効にします。--log-exports は all またはサポートされているシステムテーブルのスペース区切りリストを受け入れ、--s3-table-granularity は cluster (ウェアハウスごと) または account (統合) を受け入れます。

```
aws redshift enable-logging \
    --cluster-identifier my-redshift-cluster \
    --log-destination-type s3table \
    --log-exports all \
    --s3-table-granularity account
```

特定のシステムテーブルに対してのみ配信を有効にするには:

```
aws redshift enable-logging \
    --cluster-identifier my-redshift-cluster \
    --log-destination-type s3table \
    --log-exports sys_query_history sys_query_text sys_userlog \
    --s3-table-granularity cluster
```

配信ステータスを確認します。

```
aws redshift describe-logging-status \
    --cluster-identifier my-redshift-cluster
```

すべてのシステムテーブル、または --log-exports を持つ特定のテーブルに対する配信を無効にします。

```
aws redshift disable-logging \
    --cluster-identifier my-redshift-cluster \
    --log-destination-type s3table \
    --log-exports sys_stream_scan_states
```

Redshift Serverless

Redshift Serverless では、update-namespace を使用して名前空間ごとに機能を設定します。システムテーブル名は --s3-table-names に渡されます (--log-exports ではなく)。

配信を有効にします。

```
aws redshift-serverless update-namespace \
    --namespace-name my-namespace \
    --log-destination-type s3table \
    --s3-table-action Enable \
    --s3-table-names all \
    --s3-table-granularity namespace
```

特定のシステムテーブル (または all) に対する配信を無効にします。

```
aws redshift-serverless update-namespace \
    --namespace-name my-namespace \
    --log-destination-type s3table \
    --s3-table-action Disable \
    --s3-table-names sys_stream_scan_states
```

配信ステータスを確認します。

```
aws redshift-serverless get-namespace \
    --namespace-name my-namespace
```

S3 テーブルバケットまたはテーブルが削除された後に配信を再開するには、enable コマンドを再度実行します。

AWS Glue Data Catalogへの登録

Redshift や Athena などの分析サービスを使用して保持データをクエリする前に、サービスマネージド S3 テーブルバケットを AWS Glue Data Catalog と統合する必要があります。これは、アカウントおよびリージョンごとに 1 回だけ行う手順です。既に S3 Tables を AWS Glue Data Catalog と統合している場合は、追加のアクションは不要です。

統合手順については、「AWS Glue デベロッパーガイド」の「Amazon S3 Tables と AWS Glue Data Catalog の統合」を参照してください。

暗号化の設定

デフォルトでは、S3 Tables に配信されるデータは Amazon S3 マネージドキー (SSE-S3) で暗号化されます。代わりに AWS KMS カスタマーマネージドキーを使用するには、配信を有効にする際に `--s3-table-kms-key-id` で ARN を指定します。

```
aws redshift enable-logging \
    --cluster-identifier my-redshift-cluster \
    --log-destination-type s3table \
    --log-exports all \
    --s3-table-granularity account \
    --s3-table-kms-key-id arn:aws:kms:us-west-2:111122223333:key/key-id
```

カスタマーマネージドキーを使用する場合は、そのキーポリシーにより、Redshift システムテーブル統合サービスプリンシパルがシステムテーブルを書き込む際にデータキーを生成すること、および S3 Tables メンテナンスサービスプリンシパルがテーブルのメンテナンス (圧縮など) 中にキーを使用することを許可する必要があります。次のステートメントをキーポリシーに追加し、*REGION*、*ACCOUNT*、*KEY_ID* を実際の値に置き換えます。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "EnableRedshiftSystemTableKeyUsage",
      "Effect": "Allow",
      "Principal": {
        "Service": "systemtables.redshift.amazonaws.com"
      },
      "Action": [
        "kms:DescribeKey",
        "kms:GenerateDataKey",
        "kms:Decrypt"
      ],
      "Resource": "arn:aws:kms:REGION:ACCOUNT:key/KEY_ID",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "ACCOUNT"
        }
      }
```

```
    },
    {
      "Sid": "EnableS3TableMaintenanceKeyUsage",
      "Effect": "Allow",
      "Principal": {
        "Service": "maintenance.s3tables.amazonaws.com"
      },
      "Action": [
        "kms:GenerateDataKey",
        "kms:Decrypt"
      ],
      "Resource": "arn:aws:kms:REGION:ACCOUNT:key/KEY_ID",
      "Condition": {
        "StringLike": {
          "kms:EncryptionContext:aws:s3:arn": "arn:aws:s3tables:REGION:ACCOUNT:bucket/
aws-redshift/*"
        }
      }
    }
  ]
}
```

キーポリシーがこれらのアクセス許可を付与しない場合、Redshift は暗号化された S3 Tables に書き込むことができず、配信は失敗します。

サポートされているシステムテーブル

S3 Tables と統合するには、以下のいずれかの SYS_* モニタリングビューを選択できます。各ビューの列の説明については、ビュー名を選択してください。

- SYS_ANALYZE_COMPRESSION_HISTORY
- SYS_ANALYZE_HISTORY
- SYS_AUTOMATIC_OPTIMIZATION
- SYS_AUTO_TABLE_OPTIMIZATION
- SYS_CHILD_QUERY_TEXT
- SYS_CONNECTION_LOG
- SYS_COPY_JOB_INFO
- SYS_COPY_REPLACEMENTS
- SYS_DATASHARE_CHANGE_LOG

- SYS_DATASHARE_USAGE_CONSUMER
- SYS_DATASHARE_USAGE_PRODUCER
- SYS_DATASHARE_WRITE_HISTORY
- SYS_EXTERNAL_QUERY_ERROR
- SYS_INTEGRATION_ACTIVITY
- SYS_MV_STATE
- SYS_PROCEDURE_MESSAGES
- SYS_QUERY_DETAIL
- SYS_QUERY_EXPLAIN
- SYS_QUERY_HISTORY
- SYS_QUERY_TEXT
- SYS_SCHEMA_QUOTA_VIOLATIONS
- SYS_SESSION_HISTORY
- SYS_SPATIAL_SIMPLIFY
- SYS_STREAM_SCAN_ERRORS
- SYS_STREAM_SCAN_STATES
- SYS_UNLOAD_DETAIL
- SYS_USERLOG
- SYS_VACUUM_HISTORY

> **Note**
>
> 以下のビューには、パッチ P203 以降が必要です。SYS_CHILD_QUERY_TEXT、SYS_COPY_REPLACEMENTS、SYS_EXTERNAL_QUERY_ERRO より前のパッチを実行しているデータウェアハウスでこれらのビューを有効にすると、テーブルは正しいスキーマで作成されますが、データウェアハウスが P203 以降に更新されるまでデータは含まれません。

各テーブルに追加されたメタデータ列

各 S3 テーブルには、ソース SYS_* ビューのすべての列に加えて、Redshift が各行のソースと配信日時を識別するために追加する以下の 5 つのメタデータ列が含まれます。

列	型	説明
warehouse_account_id	string	ソースデータウェアハウスを所有する AWS アカウント。
warehouse_region_name	string	ソースデータウェアハウスが実行される AWS リージョン。
warehouse_namespace_arn	string	ソースデータウェアハウスの名前空間の ARN。これは安定した一意の識別子であり、名前変更や再作成をしても変わりません。
warehouse_name	string	ソースデータウェアハウスの人間が読み取れる名前 (クラスター名またはワークグループ名)。
s3_tables_ingestion_time	timestamp(6)、UTC	Redshift が行を S3 Tables にコミットした時刻。これは配信時刻であり、基となるイベントが発生した時刻ではありません。

保持の設定

Redshift は、S3 Tables に配信するデータの保持ポリシーを定義しません。S3 Tables でレコードの有効期限ポリシーを設定することで、データを保持する期間を設定します。日数 (5 日、100 日、365 日など) を指定すると、S3 Tables は、設定した期間を過ぎたデータを自動的に削除します。

有効期限ポリシーを設定しない場合、データは無期限に保持されます。

統合を無効にするか、システムテーブルを削除すると、Redshift は新しいデータの書き込みを停止しますが、以前に配信されたデータは S3 Tables に残り、設定した有効期限ポリシーの対象となり続けます。

詳細については、「Amazon S3 ユーザーガイド」の「S3 テーブルバケットのレコードの有効期限の設定」を参照してください。

データガバナンス

保持されているシステムテーブルデータへのアクセスはユーザーが管理します。Redshift はユーザーに代わって読み取りアクセス許可を適用したり、データをマスキングしたりしません。

- アクセス制御。S3 テーブルバケットを AWS Glue Data Catalog と統合したら、AWS Identity and Access Management (IAM) または AWS Lake Formation を使用して読み取りアクセスを管理できます。統合デプロイでは、ウェアハウスごとに読み取りアクセスの範囲を設定できます。
- カタログ化と検出。AWS Glue Data Catalog との統合により、保持されているテーブルを検出し、きめ細かなアクセス許可 (Lake Formation を使用したテーブルレベル、列レベル、行レベルのアクセスなど) を適用するための一元的な場所が提供されます。

配信とライフサイクルの動作

- 配信の頻度。データは固定頻度でバッチ配信されます。
- 正確に 1 回だけの配信。各レコードは 1 回のみ配信されます。システムテーブルを再有効化または再追加しても、重複は発生しません。
- 有効化、無効化、再有効化。機能 (または特定のシステムテーブル) が無効になっている間、その期間のデータはキャプチャされません。無効にすると書き込みは停止しますが、S3 Tables や以前に配信されたデータは削除されません。再度有効にすると、以降の配信が再開され、空白期間のデータは補完されません。
- システムテーブルを選択から削除。Redshift は、そのシステムテーブルへの新しいデータの配信を停止します。以前に配信されたデータは S3 Tables に残り、設定したレコードの有効期限ポリシーが適用されます。システムテーブルを再追加すると、データの重複なしに配信が再開されます。
- 処理中のデータを含むデータウェアハウスの削除。配信中にデータウェアハウスを削除し、そのデータウェアハウスの暗号化キーが S3 Tables で使用されているキーと異なる場合、Redshift は削除後も短期間、残りのデータの配信を完了するためにデータウェアハウスのキーに対する AWS KMS 許可を使用し続ける可能性があります。

配信ステータスの確認

現在の設定と最新の配信時間は、システムテーブルごとにいつでも確認できます。

- プロビジョニングされたクラスター。`describe-logging-status` (または `describe-cluster`) は、アクティブなシステムテーブル、S3 Tables の名前空間、粒度、および各システムテーブルの最終取り込み時間を返します。
- Redshift Serverless。`get-namespace` は、名前空間の S3 Tables 発行ステータスで同じ情報を返します。

Redshift Serverless

配信は、ワークグループをアクティブなままにしたり、コンピューティングリソースを消費したりすることはありません。ワークグループが一時停止する前に、Redshift は保留中のバッチがすべて配信されることを保証します。

ベストプラクティス

1. データの機密性を評価します。データウェアハウスに保存されているデータの種類と、システムテーブルに機密情報が含まれているかどうかを把握します。一部のシステムテーブル (SYS_QUERY_TEXT や SYS_PROCEDURE_MESSAGES など) では、クエリやストアドプロシージャからリテラル値をキャプチャできます。
2. デプロイモデルを選択します。データウェアハウスに機密情報が保存されている場合は、ウェアハウスごとのデプロイモデルを使用して、各データウェアハウスのシステムテーブルデータを物理的に分離することを検討してください。データが機密ではなく、複数のテーブルの結果を結合せずにクロスウェアハウスクエリを実行する場合は、統合デプロイモデルを使用します。
3. 必要なシステムテーブルを選択します。サポートされているシステムテーブルのリストを確認し、コンプライアンス、監査、またはオブザーバビリティの要件に合致するものを選択します。すべてのシステムテーブルを有効にする必要はありません。
4. この機能を有効にする前に暗号化を設定します。カスタマーマネージド AWS KMS キーを使用する場合は、機能を最初に有効にするときにそのキーを指定します。S3 テーブルの作成後にキーを変更することはできません。キーを変更するには、機能を無効にし、S3 Tables API を使用して S3 Tables を削除し (これにより保持データが完全に削除されます)、新しいキーで機能を再度有効にします。
5. ビジネス要件に基づいて保持期間を設定します。コンプライアンス、監査、または運用上のニーズに基づいて、S3 Tables でテーブルレベルのレコードの有効期限ポリシーを直接設定します。テーブルごとに保持期間を異なる設定にすることができます。有効期限ポリシーを設定しない場合、データは無期限に保持されます。システムテーブルのストレージ使用状況をモニタリングするには、「Amazon S3 ユーザーガイド」の「Amazon S3 Tables の CloudWatch メトリクス」を参照してください。

料金

S3 Tables へのシステムテーブルデータの書き込みは無料です。保持データの標準 S3 Tables ストレージとメンテナンス、およびデータの読み取りに使用するクエリエンジンには、そのエンジンの料金に従って課金されます。詳細については、「Amazon S3 Tables の料金」を参照してください。

考慮事項と制限事項

- 単一の AWS アカウントと単一の AWS リージョン内でサポートされます。アカウントやリージョン間をまたいでデータを分析するには、クエリ実行時に結果を結合します。
- データウェアハウスは、一度に 1 つのデプロイパターン (ウェアハウスごと、または統合) を使用します。
- デプロイパターンの切り替え、システムテーブルの無効化と再有効化、または削除と再追加を行っても、履歴データはバックフィルされません。
- 配信されたデータはイミュータブルです。Redshift を使用して個々の行を変更または削除することはできません。
- この機能で作成した S3 Tables は削除できますが、削除すると、それらのテーブルに保持されているすべてのデータが完全に削除され、配信が停止します。Redshift は、削除されたテーブルを自動的に再作成しません。配信を再開するには、この機能を再度有効にする必要があります。これにより、新しいテーブルが作成され、以降の新しいデータの配信が開始されます。以前に配信されたデータは復元されません。詳細については、「Amazon S3 ユーザーガイド」の「S3 テーブルの削除」を参照してください。
- 配信はバッチベースです (データは固定頻度で書き込まれます)。
- Redshift からクエリを実行するには、S3 テーブルバケットを AWS Glue Data Catalog と統合する必要があります。

Amazon Redshift のコンプライアンス検証

サードパーティーの監査者は、さまざまな AWS コンプライアンスプログラムの一環として Amazon Redshift のセキュリティとコンプライアンスを評価します。このプログラムには、SOC、PCI、FedRAMP、HIPAA などがあります。

特定のコンプライアンスプログラムの範囲内の AWS サービスのリストについては、「コンプライアンスプログラムによる AWS 対象範囲内のサービス」を参照してください。一般的な情報については、「AWS コンプライアンスプログラム」を参照してください。

AWS Artifact を使用して、サードパーティーの監査レポートをダウンロードできます。詳細については、「AWS Artifact のレポートのダウンロード」を参照してください。

Amazon Redshift を使用する際のお客様のコンプライアンス責任は、お客様のデータの機密性や組織のコンプライアンス目的、適用可能な法律および規制によって決定されます。Amazon Redshift の使

用が HIPAA、PCI、FedRAMP などの標準に準拠していることを前提とする場合、AWS で役立つリソースをご利用いただけます。

- セキュリティおよびコンプライアンスのクイックスタートガイドでは、AWS のデプロイメントセキュリティやコンプライアンスに重点を置いたベースライン環境におけるアーキテクチャ上の考慮事項や手順について説明しています。
- Architecting for HIPAA security and compliance ホワイトペーパー では、企業が AWS を使用して HIPAA 準拠のアプリケーションを作成する方法について説明しています。
- AWS コンプライアンスのリソースは、お客様の業界や場所に適用される場合があるワークブックおよびガイドです。
- AWS サービスの 1 つである AWS Config では、自社プラクティス、業界ガイドライン、規制に対するリソースの設定の準拠状態を評価できます。
- AWS サービスの 1 つである AWS Security Hub CSPM は、AWS 内のセキュリティ状態を総合的に把握するためのもので、セキュリティ業界における標準やベストプラクティスに準拠しているかを確認するのに役立ちます。Security Hub CSPM は、セキュリティコントロールを使用してリソース設定とセキュリティ標準を評価し、お客様がさまざまなコンプライアンスフレームワークに準拠できるようサポートします。Security Hub CSPM を使用して Amazon Redshift リソースを評価する方法の詳細については、「AWS Security Hub ユーザーガイド」の「Amazon Redshift コントロール」を参照してください。

次のコンプライアンスおよびセキュリティのドキュメントは Amazon Redshift を対象としており、AWS Artifact を通じてオンデマンドで入手できます。詳細については、「」を参照してください。AWS Artifact

- クラウドコンピューティングコンプライアンスコントロールカタログ (C5)
- ISO 27001:2013 適用宣言書 (SoA)
- ISO 27001:2013 認証
- ISO 27017:2015 適用宣言書 (SoA)
- ISO 27017:2015 認証
- ISO 27018:2015 適用宣言書 (SoA)
- ISO 27018:2014 認証
- ISO 9001:2015 認証
- PCI DSS Attestation of Compliance (AOC) と Responsibility Summary
- Service Organization Controls (SOC) 1 レポート

- Service Organization Controls (SOC) 2 レポート
- Service Organization Controls (SOC) 2 機密性に関するレポート

Amazon Redshift の復元力

AWS グローバルインフラストラクチャは、AWS リージョンとアベイラビリティーゾーン (AZ) を中心として構築されています。AWSリージョンでは、複数の物理的に独立し隔離されたアベイラビリティーゾーンが提供されており、それらは低レイテンシー、高スループット、高冗長性のネットワークにより接続されています。アベイラビリティーゾーンでは、アベイラビリティーゾーン間で中断することなく自動的にフェイルオーバーするアプリケーションとデータベースを設計および運用することができます。また、アベイラビリティーゾーンは、従来の単一データセンターのインフラストラクチャや複数データセンターのインフラストラクチャよりも可用性、耐障害性、および拡張性に優れています。

ほとんどの AWS リージョンには、複数のアベイラビリティーゾーンとデータセンターがあります。耐障害性と低レイテンシーのために、同じリージョン内の複数のアベイラビリティーゾーンにアプリケーションをデプロイできます。

データの損失やアプリケーションの変更なしに、クラスターを別のアベイラビリティーゾーンに移動するには、クラスターの再配置を設定できます。再配置により、クラスター上でサービスが中断された場合でも、影響を最小限に抑えて操作を続行できます。クラスターの再配置をオンにすると、Amazon Redshift は状況によってはクラスターの再配置を選択することがあります。Amazon Redshift の再配置の詳細については、「クラスターの再配置」を参照してください。

アベイラビリティーゾーンで予期しないイベントが発生するような障害シナリオでは、マルチアベイラビリティーゾーン (マルチ AZ) 配置を設定することで、Amazon Redshift データウェアハウスの継続的な稼働を確保できます。Amazon Redshift は 1 つのエンドポイントからアクセスできる 2 つのアベイラビリティーゾーンに同等のコンピューティングリソースをデプロイします。アベイラビリティーゾーン全体で障害が発生しても、2 つ目のアベイラビリティーゾーンの残りのコンピューティングリソースは引き続きワークロードの処理に使用できます。マルチ AZ 配置の詳細については、「マルチ AZ デプロイ」を参照してください。

AWS リージョンとアベイラビリティーゾーンの詳細については、「AWS グローバルインフラストラクチャ」を参照してください。Amazon Redshift を使用したディザスタリカバリの詳細については、「Amazon Redshift を使用したディザスタリカバリの実装」を参照してください。

.

Amazon Redshift のインフラストラクチャセキュリティ

マネージドサービスである Amazon Redshift は AWS グローバルネットワークセキュリティで保護されています。AWS セキュリティサービスと AWS がインフラストラクチャを保護する方法については「AWS クラウドセキュリティ」を参照してください。インフラストラクチャセキュリティのベストプラクティスを使用して AWS 環境を設計するには「セキュリティの柱 - AWS 適切なアーキテクチャを備えたフレームワーク」の「インフラストラクチャの保護」を参照してください。

AWS が公開した API コールを使用して、ネットワーク経由で Amazon Redshift にアクセスします。クライアントは以下をサポートする必要があります。

- Transport Layer Security (TLS)。TLS 1.2 が必須で、TLS 1.3 をお勧めします。
- DHE (楕円ディフィー・ヘルマン鍵共有) や ECDHE (楕円曲線ディフィー・ヘルマン鍵共有) などの完全前方秘匿性 (PFS) による暗号スイート。これらのモードは Java 7 以降など、ほとんどの最新システムでサポートされています。

ネットワークの隔離

Amazon VPC サービスに基づく 仮想プライベートクラウド (VPC) は、AWS クラウド内の論理的に隔離された領域にあるプライベートネットワークです。以下の手順を実行して、VPC 内に Amazon Redshift クラスターまたは Redshift Serverless ワークグループをデプロイできます。

- AWS リージョンに VPC を作成します。詳細については、Amazon VPC ユーザーガイドの Amazon VPC とは を参照してください。
- 2 つ以上のプライベート VPC サブネットを作成します。詳細については、Amazon VPC ユーザーガイドの VPC とサブネット を参照してください。
- Amazon Redshift クラスターまたは Redshift Serverless ワークグループをデプロイします。詳細については「Redshift リソースのサブネット」または「ワークグループと名前空間」を参照してください。

Amazon Redshift クラスターは、プロビジョニング時にデフォルトでロックされます。Amazon Redshift クライアントからのインバウンドネットワークトラフィックを許可するには、VPC セキュリティグループを Amazon Redshift クラスターに関連付けます。詳細については、「Redshift リソースのサブネット」を参照してください。

特定の IP アドレス範囲へのトラフィックまたは特定の IP アドレス範囲からのトラフィックのみを許可するには、VPC を使用してセキュリティグループを更新します。一例として、企業ネットワークからのトラフィックまたは企業ネットワークへのトラフィックのみを許可する場合があります。

Amazon Redshift クラスターがタグ付けされているサブネットに関連付けられたネットワークアクセスコントロールリストを設定する際には、それぞれの AWS リージョンの S3 CIDR 範囲が、イングレスルールとエグレスルールの両方で許可リストに追加されていることを確認します。これにより、Redshift Spectrum、COPY、UNLOAD などの S3 ベースの操作を中断なく実行できます。

以下のコマンド例では、us-east-1 リージョンの Amazon S3 で使用されるすべての IPv4 アドレスに対して JSON レスポンスを解析します。

```
curl https://ip-ranges.amazonaws.com/ip-ranges.json | jq -r '.prefixes[] |
 select(.region=="us-east-1") | select(.service=="S3") | .ip_prefix'

54.231.0.0/17

52.92.16.0/20

52.216.0.0/15
```

特定のリージョンの S3 IP 範囲を取得する方法については、「AWS IP アドレスの範囲」を参照してください。

Amazon Redshift は、専用テナンシー VPC へのクラスターのデプロイをサポートしています。詳細については、Amazon EC2 ユーザーガイドのハードウェア専有インスタンスを参照してください。

Amazon Redshift セキュリティグループ

Amazon Redshift クラスターをプロビジョニングすると、デフォルトではロックされているため、誰もアクセスすることはできません。他のユーザーに Amazon Redshift クラスターへのインバウンドアクセスを許可するには、クラスターをセキュリティグループに関連付けます。EC2-VPC プラットフォームを使用する場合は、既存の Amazon VPC セキュリティグループを使用するか、新しいグループを定義してクラスターに関連付けることができます。EC2-VPC プラットフォームでのクラスターの管理の詳細については、「VPC での Redshift リソース」を参照してください。

インターフェイス VPC エンドポイント

インターネット経由で接続するのではなく、仮想プライベートクラウド (VPC) のインターフェイス VPC エンドポイント (AWS PrivateLink) を通じて Amazon Redshift および Amazon Redshift

Serverless API のサービスに直接接続できます。Amazon Redshift API アクションの詳細については、Amazon Redshift API リファレンスからアクションを参照してください。Redshift Serverless API アクションの詳細については、「Amazon Redshift Serverless API リファレンス」の「アクション」を参照してください。AWS PrivateLink の詳細については、Amazon VPC ユーザーガイドの「インターフェイス VPC エンドポイント (AWS PrivateLink)」を参照してください。クラスターまたはワークスペースへの JDBC/ODBC 接続は、Amazon Redshift API サービスの一部ではないことに注意してください。

インターフェイス VPC エンドポイントを使用すると、VPC と Amazon Redshift または Redshift Serverlessの間の通信は完全に AWS ネットワーク内で実施されるため、セキュリティが強化されます。各 VPC エンドポイントは、VPC サブネット内の 1 つ以上の Elastic Network Interface とプライベート IP アドレスで表されます。Elastic Network Interface の詳細については、Amazon EC2 ユーザーガイドの「Elastic Network Interface」を参照してください。

インターフェイス VPC エンドポイントは、VPC を Amazon Redshift に直接接続します。インターネットゲートウェイ、ネットワークアドレス変換 (NAT) デバイス、仮想プライベートネットワーク (VPN) 接続、または Direct Connect 接続は使用されません。VPC のインスタンスは、パブリック IP アドレスがなくても Amazon Redshift API と通信できます。

VPC を介して Amazon Redshift または Redshift Serverless を使用するには、2 つの方法があります。1 つは、VPC 内のインスタンスから接続することです。もう 1 つの方法として、Site-to-Site VPN オプションまたは Direct Connect を使用して、プライベートネットワークを VPC に接続する方法があります。Site-to-Site VPN オプションの詳細については、Amazon VPC ユーザーガイドの「VPN 接続」を参照してください。Direct Connect の詳細については、Direct Connect ユーザーガイドの「コネクションの作成」を参照してください。

インターフェイス VPC エンドポイントを作成し、AWS マネジメントコンソール か AWS Command Line Interface (AWS CLI) コマンドのいずれかを使用して、Amazon Redshift に接続できます。詳細については、「インターフェイスエンドポイントの作成」を参照してください。

インターフェイス VPC エンドポイントを作成した後、エンドポイントのプライベート DNS ホスト名を有効にできます。その場合、デフォルトのエンドポイントは次のとおりです。

- Amazon Redshift でプロビジョニングされた場合: `https://redshift.Region.amazonaws.com`
- Amazon Redshift Serverless: `https://redshift-serverless.Region.amazonaws.com`

プライベート DNS ホスト名を有効にしない場合は、Amazon VPC が以下の形式で使用できる DNS エンドポイント名を提供します。

- Amazon Redshift でプロビジョニングされた場合: *VPC_endpoint_ID*.redshift.*Region*.vpce.amazonaws.com
- Amazon Redshift Serverless: *VPC_endpoint_ID*.redshift-serverless.*Region*.vpce.amazonaws.com

詳細については、Amazon VPC ユーザーガイドの「インターフェイス VPC エンドポイント (AWS PrivateLink) 」を参照してください。

Amazon Redshift および Redshift Serverless は、VPC 内のすべての Amazon Redshift API オペレーションと Redshift Serverless API オペレーションの呼び出しをサポートしています。

VPC エンドポイントポリシーを VPC エンドポイントにアタッチして、AWS Identity and Access Management (IAM) プリンシパルのアクセスを制御できます。また、セキュリティグループを VPC エンドポイントに関連付けて、ネットワークトラフィックの送信元と送信先に基づいてインバウンドとアウトバウンドのアクセスを制御することもできます。IP アドレスの例はです。詳細については、Amazon VPC ユーザーガイドの VPC エンドポイントによるサービスのアクセスコントロールを参照してください。

Amazon Redshift 向け VPC エンドポイントポリシー

Amazon Redshift VPC エンドポイントのポリシーを作成できます。このポリシーでは以下を指定します。

- アクションを実行できるプリンシパルまたは実行できないプリンシパル
- 実行可能なアクション
- アクションを実行できるリソース

詳細については、Amazon VPC ユーザーガイドの「VPC エンドポイントによるサービスのアクセスコントロール」を参照してください。

以下に、VPC エンドポイントポリシーの例を示します。

Amazon Redshift プロビジョンドエンドポイントポリシーの例

以下に、Amazon Redshift Provisioned の VPC エンドポイントポリシーの例を示します。

例: 指定した AWS アカウントからのすべてのアクセスを拒否する VPC エンドポイントポリシー

次の VPC エンドポイントポリシーは、AWS アカウント *123456789012* からリソースへのエンドポイントを使用したすべてのアクセスを拒否します。

```
{
    "Statement": [
        {
            "Action": "*",
            "Effect": "Allow",
            "Resource": "*",
            "Principal": "*"
        },
        {
            "Action": "*",
            "Effect": "Deny",
            "Resource": "*",
            "Principal": {
                "AWS": [
                    "123456789012"
                ]
            }
        }
    ]
}
```

例: 指定した IAM ロールにのみ VPC へのアクセスを許可する VPC エンドポイントポリシー

次の VPC エンドポイントポリシーは、AWS アカウント *123456789012* の IAM ロール *redshiftrole* にのみフルアクセスを許可します。他のすべての IAM プリンシパルは、エンドポイントを使用したアクセスを拒否されます。

```
   {
    "Statement": [
        {
            "Action": "*",
            "Effect": "Allow",
            "Resource": "*",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::123456789012:role/redshiftrole"
                ]
            }
        }]
```

```
}
```

これはサンプルのみです。ほとんどのユースケースでは、特定のアクションにアクセス許可を付与してアクセス許可の範囲を狭めることをお勧めします。

例: 指定した IAM プリンシパル (ユーザー) への VPC アクセスのみを許可する VPC エンドポイントポリシー

次の VPC エンドポイントポリシーは、AWS アカウント *123456789012* の IAM ユーザー *redshiftadmin* にのみフルアクセスを許可します。他のすべてのプリンシパルは、エンドポイントを使用したアクセスを拒否されます。

```
   {
    "Statement": [
        {
            "Action": "*",
            "Effect": "Allow",
            "Resource": "*",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::123456789012:user/redshiftadmin"
                ]
            }
        }]
}
```

これはサンプルのみです。ほとんどのユースケースでは、ユーザーにロールを割り当てる前にロールにアクセス許可をアタッチすることをお勧めします。また、特定のアクションを使用してアクセス許可の範囲を狭めることをお勧めします。

例: 読み取り専用の Amazon Redshift オペレーションを許可する VPC エンドポイントポリシー

次の VPC エンドポイントポリシーは、AWS アカウント *123456789012* にのみ、指定された Amazon Redshift アクションの実行を許可します。

指定されたアクションは、Amazon Redshift の読み込み専用アクセスに相当します。指定されたアカウントでは、VPC 上の他のすべてのアクションが拒否されます。他のすべてのアカウントは、すべてのアクセスを拒否されます。Amazon Redshift アクションのリストを表示するには、「IAM ユーザーガイド」の「Amazon Redshift のアクション、リソース、および条件キー」を参照してください。

```
{
  "Statement": [
      {
          "Action": [
              "redshift:DescribeAccountAttributes",
              "redshift:DescribeClusterParameterGroups",
              "redshift:DescribeClusterParameters",
              "redshift:DescribeClusterSecurityGroups",
              "redshift:DescribeClusterSnapshots",
              "redshift:DescribeClusterSubnetGroups",
              "redshift:DescribeClusterVersions",
              "redshift:DescribeDefaultClusterParameters",
              "redshift:DescribeEventCategories",
              "redshift:DescribeEventSubscriptions",
              "redshift:DescribeHsmClientCertificates",
              "redshift:DescribeHsmConfigurations",
              "redshift:DescribeLoggingStatus",
              "redshift:DescribeOrderableClusterOptions",
              "redshift:DescribeQuery",
              "redshift:DescribeReservedNodeOfferings",
              "redshift:DescribeReservedNodes",
              "redshift:DescribeResize",
              "redshift:DescribeSavedQueries",
              "redshift:DescribeScheduledActions",
              "redshift:DescribeSnapshotCopyGrants",
              "redshift:DescribeSnapshotSchedules",
              "redshift:DescribeStorage",
              "redshift:DescribeTable",
              "redshift:DescribeTableRestoreStatus",
              "redshift:DescribeTags",
              "redshift:FetchResults",
              "redshift:GetReservedNodeExchangeOfferings"
          ],
          "Effect": "Allow",
          "Resource": "*",
          "Principal": {
              "AWS": [
                  "123456789012"
              ]
          }
      }
  ]
```

```
}
```

例: 指定したクラスターへのアクセスを拒否する VPC エンドポイントポリシー

次の VPC エンドポイントポリシーは、すべてのアカウントとプリンシパルのフルアクセスを許可します。同時に、クラスター ID *my-redshift-cluster* の Amazon Redshift クラスターで実行されたアクションへの AWS アカウント *123456789012* のアクセスを拒否します。クラスターのリソースレベルのアクセス許可をサポートしないその他の Amazon Redshift アクションは、引き続き許可されます。Amazon Redshift アクションおよび対応するリソースタイプのリストについては、「IAM ユーザーガイド」の「Amazon Redshift のアクション、リソース、および条件キー」を参照してください。

```
 {
    "Statement": [
        {
            "Action": "*",
            "Effect": "Allow",
            "Resource": "*",
            "Principal": "*"
        },
        {
            "Action": "*",
            "Effect": "Deny",
            "Resource": "arn:aws:redshift:us-east-1:123456789012:cluster:my-redshift-
cluster",
            "Principal": {
                "AWS": [
                    "123456789012"
                ]
            }
        }
    ]
}
```

Amazon Redshift Serverless エンドポイントポリシーの例

以下に、Redshift Serverless の VPC エンドポイントポリシーの例を示します。

例: 読み取り専用の Redshift Serverless オペレーションを許可する VPC エンドポイントポリシー

次の VPC エンドポイントポリシーは、AWS アカウント *123456789012* にのみ、指定された Redshift Serverless アクションの実行を許可します。

指定されたアクションは、Redshift Serverless の読み込み専用アクセスに相当します。指定されたアカウントでは、VPC 上の他のすべてのアクションが拒否されます。他のすべてのアカウントは、すべてのアクセスを拒否されます。Redshift Serverless アクションのリストを表示するには、「IAM ユーザーガイド」の「Redshift Serverless のアクション、リソース、および条件キー」を参照してください。

```
{
  "Statement": [
      {
          "Action": [
              "redshift-serverless:DescribeOneTimeCredit",
              "redshift-serverless:GetCustomDomainAssociation",
              "redshift-serverless:GetEndpointAccess",
              "redshift-serverless:GetNamespace",
              "redshift-serverless:GetRecoveryPoint",
              "redshift-serverless:GetResourcePolicy",
              "redshift-serverless:GetScheduledAction",
              "redshift-serverless:GetSnapshot",
              "redshift-serverless:GetTableRestoreStatus",
              "redshift-serverless:GetUsageLimit",
              "redshift-serverless:GetWorkgroup"
          ],
          "Effect": "Allow",
          "Resource": "*",
          "Principal": {
              "AWS": [
                  "123456789012"
              ]
          }
      }
  ]
}
```

例: 指定したワークグループへのアクセスを拒否する VPC エンドポイントポリシー

次の VPC エンドポイントポリシーは、すべてのアカウントとプリンシパルのフルアクセスを許可します。同時に、ワークグループ ID *my-redshift-workgroup* の Amazon Redshift ワークグループで実行されたアクションへの AWS アカウント *123456789012* のアクセスを拒否します。ワークグループのリソースレベルのアクセス許可をサポートしないその他の Amazon Redshift アクションは、引き続き許可されます。Redshift Serverless アクションおよび対応するリソースタイプのリストについては、「IAM ユーザーガイド」の「Redshift Serverless のアクション、リソース、および条件キー」を参照してください。

```
 {
    "Statement": [
        {
            "Action": "*",
            "Effect": "Allow",
            "Resource": "*",
            "Principal": "*"
        },
        {
            "Action": "*",
            "Effect": "Deny",
            "Resource": "arn:aws:redshift-serverless:us-
east-1:123456789012:workgroup:my-redshift-workgroup",
            "Principal": {
                "AWS": [
                    "123456789012"
                ]
            }
        }
    ]
}
```

Amazon Redshift での設定と脆弱性の分析

AWS では、ゲストオペレーティングシステム (OS) やデータベースへのパッチ適用、ファイアウォール設定、災害対策 (DR) などのベーシックセキュリティタスクの処理を行います。これらの手順は認定サードパーティーによって確認されています。詳細については、「Amazon Redshift のコン

プライアンス検証」、「責任共有モデル」、および「セキュリティ、アイデンティティ、コンプライアンスのベストプラクティス」を参照してください。

Amazon Redshift によって自動的に更新やパッチがデータウェアハウスに適用されるため、お客様は管理作業ではなくアプリケーションそのものに集中することができます。パッチとアップグレードは、設定可能なメンテナンスウィンドウ中に適用されます。詳細については、「メンテナンスウィンドウ」を参照してください。

Amazon Redshift クエリエディタ v2 はAWSで管理されるアプリケーションです。すべてのパッチと更新は必要に応じて AWS によって適用されます。

ID が強化された IAM ロールセッションを使用して Redshift に接続する

IAM アイデンティティセンターを使用して、Amazon Redshift クラスターとサーバーレスワークグループへのフェデレーションアクセスを提供できます。このアプローチにより、ユーザーはアイデンティティセンターの認証情報を使用して認証できます。

Amazon Redshift は、ユーザー ID 情報を含む承認済みトークンを生成する `GetIdentityCenterAuthToken` API オペレーションを提供します。これらの API は、プロビジョニングされたクラスターとサーバーレスワークグループの両方で使用できます。トークンを使用すると、既存のアイデンティティセンターのセットアップを使用して、Amazon Redshift データベースへのシームレスなシングルサインオンアクセスが可能になります。

前提条件

Amazon Redshift で Identity Center 認証を使用する前に、次を確認してください。

- Identity Center のセットアップ: アカウントには、ユーザー ID と適切なアプリケーション割り当てで IAM アイデンティティセンターが設定されている必要があります。セットアップの手順については、「IAM アイデンティティセンターの設定」を参照してください。

  > **Important**
  >
  > Redshift に接続する場合、redshift:connect スコープを使用する必要があります。

- ID 拡張認証情報: アプリケーションは、埋め込みユーザー ID 情報を含む ID 拡張認証情報を使用する必要があります。詳細については、「ID 拡張 IAM ロールセッションの使用」を参照してください。
- IAM アクセス許可: IAM ロールまたはユーザーには、`GetIdentityCenterAuthToken` API を呼び出し、指定したクラスターまたはワークグループにアクセスするためのアクセス許可が必要です。必要なアクセス許可:
  - プロビジョニングされたクラスターの場合: クラスター ARN の `redshift:GetIdentityCenterAuthToken` (形式: `arn:aws:redshift:region:account:cluster:cluster-name`)

  - サーバーレスワークグループの場合: ワークグループ ARN の `redshift-serverless:GetIdentityCenterAuthToken` (形式: `arn:aws:redshift-serverless:region:account:workgroup/workgroup-name`)
- 互換性のあるドライバー: Identity Center 認証トークンをサポートする Amazon Redshift JDBC または ODBC ドライバーを使用します。
  - JDBC ドライバー: 「Amazon Redshift JDBC ドライバーバージョン 2.x のインストールと設定」を参照してください。
  - ODBC ドライバー: 「Amazon Redshift ODBC ドライバーバージョン 2.0 のインストールと設定」を参照してください。
- 拡張 VPC ルーティングのネットワーク設定: 拡張 VPC ルーティングを有効にする場合、VPC はアイデンティティセンターのサービスにアクセスできる必要があります。この接続を提供するには、インターフェイス VPC エンドポイントを使用することをお勧めします。ユーザーがサインインする前に、この接続を設定してください。アイデンティティセンターの拡張 VPC ルーティング要件の詳細については、「拡張 VPC ルーティングでの AWS IAM アイデンティティセンター認証の使用」を参照してください。

Identity Center 認証の仕組み

Amazon Redshift のアイデンティティセンター認証では、次のワークフローを使用します。

1. アプリケーションは、埋め込みユーザー ID 情報を含む ID 拡張認証情報を使用して `GetIdentityCenterAuthToken` API を呼び出します。
2. Amazon Redshift は Identity Center ID を検証し、特定のクラスターまたはワークグループを対象とする暗号化された認証トークンを生成します。IAM ポリシーの例を参照してください。
3. アプリケーションはこのトークンを使用して、指定された Amazon Redshift クラスターまたはワークグループに接続します。
4. Amazon Redshift データプレーンはトークンを検証し、Identity Center アプリケーション内の Identity Center ユーザーのアクセス許可に基づいてアクセスを許可します。

> **Important**
>
> この API には、アイデンティティが強化された認証情報が必要です。詳細については、「ID 拡張 IAM ロールセッションの使用」を参照してください。

ID 拡張認証情報なしで API を呼び出すと、UnsupportedOperationFault エラーを受け取ります。

GetIdentityCenterAuthToken API オペレーション

Amazon Redshift は 2 つの異なる GetIdentityCenterAuthToken API オペレーションを提供します。1 つはプロビジョニングされたクラスター用、もう 1 つはサーバーレスワークグループ用です。両方のオペレーションは同じ名前ですが、ターゲットリソースタイプに応じて異なるパラメータを受け入れます。

プロビジョニングされたクラスターの GetIdentityCenterAuthToken

プロビジョニングされた Amazon Redshift クラスターの場合、Amazon Redshift サービスの GetIdentityCenterAuthToken API を使用して、承認されたトークンを生成します。

リクエスト構文

```
{
   "ClusterIds": [ "string" ]
}
```

パラメータのリクエスト

ClusterIds

トークンがアクセスを許可される Amazon Redshift クラスター識別子のリスト。トークンは、このリストで指定されたクラスターでの認証にのみ使用できます。

型: 文字列の配列

長さの制限: 最小は 1 項目です。最大 20 項目。

必須: はい

CLI の例

例: 1 つのクラスターの承認されたトークンを取得する

```
aws redshift get-identity-center-auth-token \
    --cluster-ids my-redshift-cluster
```

例: 複数のクラスターの承認されたトークンを取得する

```
aws redshift get-identity-center-auth-token \
    --cluster-ids my-cluster-1 my-cluster-2
```

サーバーレスワークグループ用の GetIdentityCenterAuthToken

Amazon Redshift Serverless ワークグループの場合、Amazon Redshift Serverless サービスの `GetIdentityCenterAuthToken` API を使用して、承認されたトークンを生成します。

リクエスト構文

```
{
   "WorkgroupNames": [ "string" ]
}
```

パラメータのリクエスト

WorkgroupNames

トークンがアクセスを許可される Amazon Redshift Serverless ワークグループ名のリスト。トークンは、このリストで指定されたワークグループでの認証にのみ使用できます。

型: 文字列の配列

長さの制限: 最小は 1 項目です。最大 20 項目。

必須: はい

CLI の例

例: 1 つのワークグループの承認されたトークンを取得する

```
aws redshift-serverless get-identity-center-auth-token \
```

```
    --workgroup-names my-workgroup
```

例: 複数のワークグループの承認されたトークンを取得する

```
aws redshift-serverless get-identity-center-auth-token \
    --workgroup-names workgroup-1 workgroup-2
```

レスポンスの構文

両方の API は、同じレスポンス構造を返します。

```
{
   "AuthorizedToken": "string",
   "ExpirationTime": "timestamp"
}
```

レスポンスパラメータ

AuthorizedToken

ユーザー ID 情報と、承認されたクラスターまたはワークグループのリストを含む、暗号化および承認されたトークン。このトークンは機密データとして扱う必要があります。

タイプ: 文字列

ExpirationTime

トークンの有効期限が切れる UTC 形式の日時。トークンは、生成時から 1 時間有効です。

タイプ: タイムスタンプ

レスポンスの例

```
{
    "AuthorizedToken":
 "eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiJ1c2VyQGV4YW1wbGUuY29tIiwiaWF0IjoxNjQwOTk1MjAw
    "ExpirationTime": "2024-01-01T12:00:00Z"
}
```

ドライバーの統合

Amazon Redshift ドライバーは、トークンを直接使用してアイデンティティセンター認証をサポートします。

トークンの直接使用

`GetIdentityCenterAuthToken` API を呼び出してトークンを取得したら、`SUBJECT_TOKEN` トークンタイプで `IdpTokenAuthPlugin` を使用します。

接続設定:

```
plugin_name = com.amazon.redshift.plugin.IdpTokenAuthPlugin
token_type = SUBJECT_TOKEN
token = {encrypted_token_from_api_response}
```

Identity Center 認証プラグインとドライバー設定の詳細については、「Amazon Redshift クラスターへの接続」を参照してください。

Java コード例

アイデンティティセンター認証を使用して接続するための Java コードの例:

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.util.Properties;

// Get token from GetIdentityCenterAuthToken API
String token = "your_encrypted_token_from_api_response";

// Configure connection properties
Properties props = new Properties();
props.setProperty("user", "your_username");
props.setProperty("plugin_name", "com.amazon.redshift.plugin.IdpTokenAuthPlugin");
props.setProperty("token_type", "SUBJECT_TOKEN");
props.setProperty("token", token);
```

```
// Connect to Redshift
String url = "jdbc:redshift://your-cluster.region.redshift.amazonaws.com:5439/
your_database";
try (Connection conn = DriverManager.getConnection(url, props)) {
    // Use connection
    System.out.println("Connected successfully!");
} catch (SQLException e) {
    e.printStackTrace();
}
```

IAM ポリシーの要件

Amazon Redshift でアイデンティティセンター認証を使用するには、Amazon Redshift クラスターとワークグループへの接続に必要な標準のアクセス許可以外にも、特定の IAM アクセス許可が必要です。

API アクセス許可

プロビジョニングされたクラスターの場合、拡張 IAM ロールセッションに次のものが必要です。

- クラスター ARN の `redshift:GetIdentityCenterAuthToken` (形式: `arn:aws:redshift:region:account:cluster:cluster-name`)

サーバーレスワークグループの場合、拡張 IAM ロールセッションには次のものが必要です。

- ワークグループ ARN の `redshift-serverless:GetIdentityCenterAuthToken` (形式: `arn:aws:redshift-serverless:region:account:workgroup/workgroup-name`)

IAM ポリシーの例

プロビジョニングされたクラスターのポリシーの例:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "redshift:GetIdentityCenterAuthToken"
            ],
            "Resource": [
                "arn:aws:redshift:us-east-1:123456789012:cluster:my-cluster"
            ]
        }
    ]
}
```

サーバーレスワークグループのポリシーの例:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "redshift-serverless:GetIdentityCenterAuthToken"
            ],
            "Resource": [
                "arn:aws:redshift-serverless:us-east-1:123456789012:workgroup/my-
workgroup"
            ]
        }
    ]
}
```

複数のリソースのポリシーの例:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "redshift:GetIdentityCenterAuthToken"
            ],
            "Resource": [
                "arn:aws:redshift:*:123456789012:cluster/*"
            ]
        },
        {
            "Effect": "Allow",
```

```
            "Action": [
                "redshift-serverless:GetIdentityCenterAuthToken"
            ],
            "Resource": [
                "arn:aws:redshift-serverless:*:123456789012:workgroup/*"
            ]
        }
    ]
}
```

リージョン別の可用性

Identity Center 認証は、以下の AWS リージョンで使用できます。

- 商用リージョン: サポートされているすべての Amazon Redshift リージョン
- AWS GovCloud: us-gov-east-1 および us-gov-west-1 で利用可能
- 中国リージョン: cn-north-1 および cn-northwest-1 で利用可能

> **Note**
>
> 機能の高可用性は、最初のロールアウト中は異なる場合があります。

Identity Center 認証のトラブルシューティング

以下のシナリオを使用して、Identity Center 認証で発生する問題をトラブルシューティングします。

トピック

- API エラー
- 接続エラー

API エラー

`GetIdentityCenterAuthToken` API は次のエラーを返すことがあります。

- UnsupportedOperationFault - リクエストが ID 拡張認証情報なしで行われたか、Identity Center の ID の伝播が現在の設定でサポートされていません。

- ClusterNotFoundFault (redshift:GetIdentityCenterAuthToken) - 指定されたクラスター識別子が 1 つ以上存在しないか、アクセスするアクセス許可がありません。
- WorkgroupNotFoundFault (redshift-serverless:GetIdentityCenterAuthToken) - 指定されたワークグループ名の 1 つ以上が存在しないか、アクセスするアクセス許可がありません。
- InvalidParameterValueException - クラスターまたはワークグループのリストが空、またはリスト内の項目が 20 個を超えるなど、リクエストパラメータが無効です。

接続エラー

アイデンティティセンタートークンを使用して Amazon Redshift に接続すると、次のエラーが発生することがあります。

- Token credentials expired or invalid - `GetIdentityCenterAuthToken` API を使用して新しいトークンを生成し、接続設定を更新します。
- invalid token - トークン形式を確認し、正しいクラスターまたはワークグループに対して生成されたことを確認します。
- TYPE_INVALID_TOKEN - 接続パラメータで `token_type = SUBJECT_TOKEN` を使用していることを確認します。
- identity provider type "awsidc" is not supported - Amazon Redshift クラスターまたはワークグループが Identity Center 認証をサポートしていることを確認し、互換性のあるバージョンに更新します。
- AWS IdC identity provider does not exist - AWS コンソールを使用して Amazon Redshift リソースの Identity Center 統合を設定します。
- Invalid scope. User's credentials are not authorized to connect to Amazon Redshift - ユーザーに Identity Center の適切なアクセス許可があり、トークンが正しいリソースに対して生成されたことを確認します。
- 拡張 VPC ルーティングでの接続障害 – 拡張 VPC ルーティングを有効にすると、Amazon Redshift は VPC を介してアイデンティティセンターのサービスにアクセスします。これらのサービス用のインターフェイス VPC エンドポイントが VPC に存在することを確認してください。エンドポイントごとにプライベート DNS 名がオンになっていることを確認します。エンドポイントセキュリティグループで TCP ポート 443 へのインバウンドトラフィックを許可していることを確認してください。アイデンティティセンターの拡張 VPC ルーティング要件の詳細については、「拡張 VPC ルーティングでの AWS IAM アイデンティティセンター認証の使用」を参照してください。

ネットワークタスク

Redshift データベースへの接続のカスタマイズなどのネットワークタスクを行うことができます。これは、セキュリティやその他の目的でトラフィックを制御するときに実施することをお勧めします。また、Redshift リソースのカスタムドメイン名の設定など、DNS 関連のタスクを実行することもできます。これらの設定タスクは、Amazon Redshift でプロビジョニングされたクラスター、または Amazon Redshift Serverless ワークグループを使用している場合に実行できます。

トピック

- クライアント接続のカスタムドメイン名
- RedShift が管理する VPC エンドポイント
- VPC での Redshift リソース
- 拡張 VPC ルーティングを使用して Redshift でネットワークトラフィックを制御する

クライアント接続のカスタムドメイン名

Amazon Redshift クラスターおよび Amazon Redshift Serverless ワークグループの両方用に、カスタムドメイン名 (カスタム URL とも呼ばれます) を作成できます。SQL クライアント接続をエンドポイントにルーティングする、読みやすい DNS レコードです。既存のクラスターやワークグループにいつでも設定できます。これにはいくつかの利点があります。

- カスタムドメイン名は、クラスター名またはワークグループ名とリージョンが通常含まれているデフォルトの URL よりも単純な文字列です。簡単に思い出し、使用できます。
- フェイルオーバーの場合などにトラフィックを新しいクラスターやワークグループにすばやくルーティングできます。これにより、クライアントは再接続時に設定を変更する必要がなくなります。接続は、中断を最小限に抑えながら、一元的に再ルーティングできます。
- サーバー名などのプライベートな情報を接続 URL で共有しないように、カスタム URL では非表示にすることができます。

CNAME を使用してカスタムドメイン名を設定する場合、Amazon Redshift の追加料金は発生しません。新しいドメイン名を作成すると、DNS プロバイダーからドメイン名の料金が請求される場合がありますが、通常この費用は少額です。

ドメイン名の登録

カスタムドメイン名の設定は、DNS プロバイダーへのドメイン名の登録や証明書の作成など、いくつかのタスクで構成されます。これらのタスクを実行したら、Amazon Redshift コンソールまたは Amazon Redshift Serverless コンソールでカスタムドメイン名を設定するか、AWS CLI で設定します。

Amazon Redshift でカスタムドメイン名を設定するには、登録されたインターネットドメイン名が必要です。Amazon Route 53 を使用するか、サードパーティーのドメイン登録プロバイダーを使用して、インターネットドメインを登録できます。これらのタスクは Amazon Redshift コンソールの外部で行われます。カスタムドメイン作成の残りの手順を進めるには、登録済みドメインがあることが前提です。

> **Note**
>
> プロビジョニング済みのクラスターを使用する場合は、カスタムドメイン名の設定手順を実行する前に、再配置を有効にしておく必要があります。詳細については、「クラスターの再配置」を参照してください。Amazon Redshift Serverless では、この手順は必要ありません。

カスタムドメイン名には通常、ルートドメインとサブドメインが含まれます (例: `mycluster.example.com`)。設定するには、以下のステップを実行します。

カスタムドメイン名の DNS CNAME エントリを作成する

1. ルートドメイン、例えば `example.com` を登録します。必要に応じて、既存のドメインを選択できます。カスタム名は、特定の文字に対する制限やその他の命名検証によって制限される場合があります。Route 53 を使用してドメインを登録する方法の詳細については、「新しいドメインの登録」を参照してください。
2. カスタムドメイン名から、クラスターまたはワークグループの Redshift エンドポイントを参照する DNS CNAME レコードを追加します。エンドポイントは、クラスターまたはワークグループのプロパティ、Redshift コンソール、または Amazon Redshift Serverless コンソールで確認できます。[一般的な情報] の下に表示されているクラスターまたはワークグループのプロパティの [JDBC URL] をコピーします。URL は次のようになっています。

   - Amazon Redshift クラスター: `redshift-cluster-sample.abc123456.us-east-1.redshift.amazonaws.com`

- Amazon Redshift Serverless ワークグループ: `endpoint-name.012345678901.us-east-1-dev.redshift-serverless-dev.amazonaws.com`

URL に JDBC プレフィックスが付いている場合は、削除します。

> **Note**
>
> DNS レコードの名前はそれぞれ一意で、組織内で使用できる必要があるため、DNS レコードは空き状況により異なります。

制限事項

カスタムドメインの CNAME レコードの作成には、いくつか制約があります。

- 同一のプロビジョニング済みクラスターまたは Amazon Redshift Serverless ワークグループに複数のカスタムドメイン名を作成することはできません。CNAME レコードは 1 つだけ関連付けることができます。
- CNAME レコードを複数のクラスターまたはワークグループに関連付けることはできません。各 Redshift リソースの CNAME は一意にする必要があります。

ドメインを登録して CNAME レコードを作成したら、新しい証明書または既存の証明書を選択します。このステップは AWS Certificate Manager を使用して実行します。

マネージド更新の条件を満たした DNS 検証による証明書を作成しておくことをお勧めします。マネージド更新は AWS Certificate Manager で利用できます。マネージド更新では、ACM が証明書を自動的に更新するか、有効期限が近づくと E メール通知を送信します。詳細については、「ACM 証明書のマネージド更新」を参照してください。

ドメイン名の証明書のリクエスト

Amazon Redshift または Amazon Redshift Serverless では、通信を安全に保ち、ドメイン名の所有権を検証するために、カスタムエンドポイント用の検証済みの Secure Sockets Layer (SSL) 証明書が必要です。AWS KMS key の AWS Certificate Manager アカウントを使用して証明書を安全に管理できます。セキュリティ検証には、完全なホスト名検証 (sslmode=verify-full) が含まれます。

E メール検証ではなく DNS 検証を選択した場合にのみ、証明書の更新が Amazon Redshift によって管理されます。E メール検証を使用する場合は証明書を使用できますが、有効期限が切れる前に自分

で更新を行う必要があります。証明書には DNS 検証を選択することをお勧めします。インポートした証明書の有効期限は AWS Certificate Manager でモニタリングできます。

ACM にドメイン名の証明書をリクエストする

1. AWS マネジメントコンソール にサインインして、ACM コンソール (https://console.aws.amazon.com/acm/home) を開きます。
2. [Request a certificate] (証明書のリクエスト) を選択します。
3. [ドメイン名] フィールドにカスタムドメイン名を入力します。

   > **Note**
   >
   > 1 つの証明書を複数のカスタムドメインレコードに使用するには、証明書ドメインに加えて多数のプレフィックスを指定できます。例として、`one.example.com` や `two.example.com` などの追加レコード、`*.example.com` などのワイルドカード DNS レコードを同じ証明書で使用できます。

4. [Review and request] を選択します。
5. [Confirm and request] を選択します。
6. リクエストが有効であるためには、ACM が証明書を発行する前に、インターネットドメインの登録された所有者がリクエストに同意する必要があります。ステップを完了したら、ACM コンソールにステータスが [発行済み] と表示されていることを確認します。

カスタムドメインの設定

Amazon Redshift または Amazon Redshift Serverless コンソールを使用してカスタムドメイン URL を作成できます。まだ設定していない場合は、[一般情報] の下に [カスタムドメイン名] プロパティがダッシュ (—) で表示されます。CNAME レコードと証明書を作成したら、クラスターまたはワークグループのカスタムドメイン名を関連付けます。

カスタムドメイン名の関連付けを作成するには、次の IAM アクセス許可が必要です。

- `redshift:CreateCustomDomainAssociation` — ARN を追加することにより、特定のクラスターへのアクセス許可を制限できます。
- `redshiftServerless:CreateCustomDomainAssociation` — ARN を追加することにより、特定のワークグループへのアクセス許可を制限できます。
- `acm:DescribeCertificate`

ベストプラクティスとして、アクセス許可ポリシーを IAM ロールにアタッチし、それを必要に応じてユーザーやグループに割り当てることをお勧めします。詳細については、「Amazon Redshift での Identity and Access Management」を参照してください。

カスタムドメイン名を割り当てるには、次のステップを実行します。

1. Redshift コンソールでクラスターを選択するか、Amazon Redshift Serverless コンソールでワークグループを選択し、[アクション] メニューの [カスタムドメイン名の作成] を選択します。ダイアログが表示されます。
2. カスタムドメイン名を入力します。
3. [ACM 証明書]の ARN を AWS Certificate Manager から選択します。変更を保存します。証明書の作成手順のガイダンスに従い、AWS Certificate Manager によるマネージド更新の対象となる DNS 検証済み証明書を選択することをお勧めします。
4. クラスターのプロパティで、[カスタムドメイン名] と [カスタムドメイン証明書 ARN] にエントリが入力されていることを確認します。[カスタムドメイン証明書の有効期限] も表示されます。

カスタムドメインを設定した後、`sslmode=verify-full` は新しいカスタムドメインでのみ使用できます。デフォルトのエンドポイントでは機能しません。ただし、`sslmode=verify-ca` などの他の SSL モードを使用してデフォルトのエンドポイントに接続することはできます。

> **Note**
>
> 念のため、クラスターの再配置は、追加の Redshift ネットワーク機能を設定するための前提条件ではありません。有効にしなくても、次の項目を有効にすることができます。
>
> - クロスアカウントまたはクロスリージョン VPC から Redshift への接続 — ある AWS 仮想プライベートクラウド (VPC) から、Redshift データベースを含む別の仮想プライベートクラウド (VPC) に接続できます。これにより、データベースに接続する ID へのローカル VPC アクセスを提供しなくても、異なるアカウントや VPC からのクライアントアクセスなどを簡単に管理できます。詳細については、「Amazon Redshift マネージド VPC エンドポイントから Amazon Redshift Serverless への接続」を参照してください。
> - カスタムドメイン名の設定 — このトピックで説明されているように、カスタムドメイン名を作成して、エンドポイント名をよりわかりやすくすることができます。

Amazon Redshift のプロビジョニングされたクラスターまたは Amazon Redshift Serverless ワークグループへの接続

カスタムドメイン名で接続するには、プロビジョニング済みクラスターに対して以下の IAM アクセス許可 (`redshift:DescribeCustomDomainAssociations`) が必要です。Amazon Redshift Serverless では、アクセス許可を追加する必要はありません。

ベストプラクティスとして、アクセス許可ポリシーを IAM ロールにアタッチし、それを必要に応じてユーザーやグループに割り当てることをお勧めします。詳細については、「Amazon Redshift での Identity and Access Management」を参照してください。

コンソールで CNAME を作成してクラスターまたはワークグループに割り当てるステップを完了したら、SQL クライアントの接続プロパティでカスタム URL を指定できます。CNAME レコードを作成した直後に DNS の伝播が遅れる場合があることに注意してください。

1. SQL クライアントを開きます。例えば、SQL/Workbench J を使用できます。接続のプロパティを開き、接続文字列にカスタムドメイン名を追加します。例えば、`jdbc:redshift://mycluster.example.com:5439/dev?sslmode=verify-full`。この例では、`dev` はデフォルトのデータベースを指定します。
2. データベースユーザーの [ユーザー名] と [パスワード] を追加します。
3. 接続をテストします。特定のテーブルなどのデータベースリソースをクエリできるかどうかは、データベースユーザーに付与されるアクセス許可または割り当てられた Amazon Redshift データベースロールに付与されるアクセス許可によって異なる場合があります。

   クラスターまたはワークグループが VPC 内にある場合は、パブリックにアクセスできるように設定しないと接続できない場合があります。この設定はネットワークプロパティで変更できます。

> **Note**
>
> カスタムドメイン名への接続は、JDBC、ODBC と Python ドライバーでサポートされています。

カスタムドメインが割り当てられているクラスターの名前を変更する

> **Note**
>
> この一連の手順は Amazon Redshift Serverless ワークグループには適用されません。ワークグループ名を変更することはできません。

カスタムドメイン名を持つクラスターの名前を変更するには、`acm:DescribeCertificate` IAM アクセス許可が必要です。

1. Amazon Redshift コンソールに移動し、名前を変更するクラスターを選択します。[編集] を選択してクラスターのプロパティを編集します。
2. [クラスター識別子] を編集します。クラスターの他のプロパティを変更することもできます。次に、変更の保存を選択します。
3. クラスターの名前を変更したら、DNS レコードを更新して、カスタムドメインの CNAME エントリが更新された Amazon Redshift エンドポイントを指すように変更する必要があります。

カスタムドメインの関連付けを説明する

このセクションのコマンドを使用して、特定のプロビジョニング済みクラスターまたは Amazon Redshift Serverless ワークグループに関連付けられているカスタムドメイン名のリストを取得します。

次のアクセス許可が必要です。

- プロビジョニング済みクラスター: `redshift:DescribeCustomDomainAssociations`
- Amazon Redshift Serverless ワークグループ: `redshiftServerless:ListCnameAssociations`

ベストプラクティスとして、アクセス許可ポリシーを IAM ロールにアタッチし、それを必要に応じてユーザーやグループに割り当てることをお勧めします。詳細については、「Amazon Redshift での Identity and Access Management」を参照してください。

以下は、特定の Amazon Redshift クラスターのカスタムドメイン名を一覧表示するサンプルコマンドを示しています。

```
aws redshift describe-custom-domain-associations --custom-domain-name customdomainname
```

カスタムドメイン名が有効になっている場合にこのコマンドを実行すると、クラスターに関連付けられているカスタムドメイン名を確認できます。カスタムドメインの関連付けを記述するための CLI コマンドの詳細については、「describe-custom-domain-associations」を参照してください。

同様に、以下は、特定の Amazon Redshift Serverless ワークグループのカスタムドメイン名を一覧表示するサンプルコマンドを示しています。この操作には、複数の実行方法があります。カスタムドメイン名だけを指定できます。

```
aws redshift-serverless list-custom-domain-associations --custom-domain-
name customdomainname
```

証明書 ARN のみを指定して関連付けを取得することもできます。

```
aws redshift-serverless list-custom-domain-associations --custom-domain-certificate-
arn certificatearn
```

カスタムドメイン名が有効になっている場合にこれらのコマンドを実行して、ワークグループに関連付けられているカスタムドメイン名を確認できます。コマンドを実行してカスタムドメインの関連付けのプロパティを取得することもできます。そのためには、カスタムドメイン名とワークグループ名をパラメータとして指定する必要があります。証明書 ARN、ワークグループ名、カスタムドメインの証明書の有効期限が返されます。

```
aws redshift-serverless get-custom-domain-association --workgroup-name workgroupname --
custom-domain-name customdomainname
```

Amazon Redshift Serverless で利用可能な CLI リファレンスコマンドの詳細については、「redshift-serverless」を参照してください。

カスタムドメインを別の証明書に関連付ける

カスタムドメイン名との証明書の関連付けを変更するには、次の IAM アクセス許可が必要です。

- `redshift:ModifyCustomDomainAssociation`
- `acm:DescribeCertificate`

ベストプラクティスとして、アクセス許可ポリシーを IAM ロールにアタッチし、それを必要に応じてユーザーやグループに割り当てることをお勧めします。詳細については、「Amazon Redshift での Identity and Access Management」を参照してください。

次のコマンドを使用して、カスタムドメインを異なる証明書に関連付けます。--custom-domain-name 引数と custom-domain-certificate-arn 引数は必須です。新しい証明書の ARN は既存の ARN とは異なる必要があります。

```
aws redshift modify-custom-domain-association --cluster-id redshiftcluster --custom-domain-name customdomainname --custom-domain-certificate-arn certificatearn
```

次のサンプルは、カスタムドメインを Amazon Redshift Serverless ワークグループの別の証明書に関連付ける方法を示しています。

```
aws redshift-serverless modify-custom-domain-association --workgroup-name redshiftworkgroup --custom-domain-name customdomainname --custom-domain-certificate-arn certificatearn
```

クラスターに接続できるようになるまでに最大 30 秒の遅延があります。遅延の一部は Amazon Redshift クラスターがプロパティを更新するときに発生し、DNS が更新されるときにさらに遅延が発生します。API と各プロパティ設定の詳細については、「ModifyCustomDomainAssociation」を参照してください。

カスタムドメインを削除する

カスタムドメイン名を削除するには、ユーザーに次のアクションのアクセス許可が必要です。

- プロビジョニング済みクラスター: `redshift:DeleteCustomDomainAssociation`
- Amazon Redshift Serverless ワークグループ: `redshiftServerless:DeleteCustomDomainAssociation`

コンソール内

カスタムドメイン名を削除するには、[アクション] ボタンを選択し、[カスタムドメイン名を削除] を選択します。削除後も、コンソールに一覧表示されているエンドポイントを使用するようにツールを更新すれば、サーバーに接続できます。

CLI コマンドの使用

次のサンプルは、カスタムドメイン名を削除する方法を示しています。削除操作では、クラスターの既存のカスタムドメイン名を指定する必要があります。

```
aws redshift delete-custom-domain-association --cluster-id redshiftcluster --custom-domain-name customdomainname
```

次のサンプルは、Amazon Redshift Serverless ワークグループのカスタムドメイン名の削除方法を示しています。カスタムドメイン名は必須パラメータです。

```
aws redshift-serverless delete-custom-domain-association --workgroup-name workgroupname
 --custom-domain-name customdomainname
```

詳細については、「DeleteCustomDomainAssociation」を参照してください。

RedShift が管理する VPC エンドポイント

デフォルトでは、Amazon Redshift クラスターまたは Amazon Redshift Serverless ワークグループは仮想プライベートクラウド (VPC) 内にプロビジョニングされます。VPC は、パブリックアクセスを許可するか、インターネットゲートウェイ、NAT デバイス、または AWS Direct Connect 接続を設定してトラフィックをクラスターにルーティングすると、別の VPC またはサブネットからアクセスすることができます。また、Redshift 管理の VPC エンドポイント (AWS PrivateLink を使用) を設定して、クラスターまたはワークグループにアクセスすることもできます。

Redshift 管理の VPC エンドポイントは、クラスターまたはワークグループを含む VPC とクライアントツールを実行する VPC 間のプライベート接続としてセットアップすることができます。クラスターまたはワークグループが別のアカウントにある場合、アカウント所有者 (付与者) は、接続するアカウント (被付与者) へのアクセスを許可する必要があります。これにより、パブリック IP アドレスを使用したり、インターネット経由でトラフィックをルーティングしたりすることなく、データウェアハウスにアクセスできます。

Redshift 管理の VPC エンドポイントを使用してアクセスを許可する一般的な理由は次のとおりです。

- AWS アカウント A が AWS アカウント B の VPC にクラスターまたはワークグループへのアクセスを許可する。
- AWS アカウント A が AWS アカウント A の VPC にクラスターまたはワークグループへのアクセスを許可する。

- AWS アカウント A が AWS アカウント A の VPC にある別のサブネットに、クラスターまたはワークグループへのアクセスを許可する。

Redshift 管理の VPC エンドポイントを設定して、別のアカウントのクラスターまたはワークグループにアクセスするためのワークフローは次のとおりです。

1. 所有者アカウントは、別のアカウントにアクセス許可を付与し、被付与者の AWS アカウント ID と VPC 識別子 (またはすべての VPC) を指定します。
2. 被付与者アカウントには、Redshift 管理の VPC エンドポイントを作成する権限があることが通知されます。
3. 被付与者アカウントは、Redshift 管理の VPC エンドポイントを作成します。
4. 被付与者アカウントは、Redshift 管理の VPC エンドポイントを使用して、所有者アカウントのクラスターまたはワークグループにアクセスできるようになります。

このプロセスは、Amazon Redshift コンソール、AWS CLI、または Amazon Redshift API を使用して実行できます。

Redshift で管理される VPC エンドポイントを使用する場合の考慮事項

> **Note**
>
> RedShift が管理する VPC エンドポイントを作成または変更するには、`ec2:ModifyVpcEndpoint` マネージドポリシー `AmazonRedshiftFullAccess` に指定されている他のアクセス許可に加えて、IAM ポリシーに `ec2:CreateVpcEndpoint` と AWS のアクセス許可が必要です。

Redshift が管理する VPC エンドポイントを使用する場合は、以下に注意してください。

- プロビジョニングされたクラスターを使用している場合は、RG または RA3 ノードタイプが必要です。Amazon Redshift Serverless ワークグループは、VPC エンドポイントの設定にも機能します。
- プロビジョニングされたクラスターの場合、クラスターが再配置やマルチ AZ で有効になっていることを確認します。クラスターの再配置を有効にする要件については、「クラスターの再配置」を参照してください。マルチ AZ を有効にする方法については、「新しいクラスターを作成する際のマルチ AZ の設定」を参照してください。

- セキュリティグループを介してアクセスするクラスターまたはワークグループが、有効なポート範囲 5431 ~ 5455 と 8191 ~ 8215 で利用できることを確認します。デフォルトは 5439 です。
- 既存の Redshift で管理される VPC エンドポイントに関連付けられた VPC セキュリティグループを変更できます。他の設定を変更するには、現在の Redshift が管理する VPC エンドポイントを削除し、新しいエンドポイントを作成します。
- 作成できる Redshift 管理の VPC エンドポイントの数は、VPC エンドポイントのクォータに制限されます。
- Redshift 管理の VPC エンドポイントには、インターネットからアクセスできません。Redshift 管理の VPC エンドポイントは、エンドポイントがプロビジョニングされている VPC 内か、ルートテーブルとセキュリティグループによって許可されているエンドポイントがプロビジョニングされている VPC とピア接続されている VPC からのみアクセスできます。
- Amazon VPC コンソールを使用して Redshift 管理の VPC エンドポイントを管理することはできません。
- プロビジョニングされたクラスターに RedShift 管理の VPC エンドポイントを作成する場合、選択する VPC にはクラスターサブネットグループが必要です。サブネットグループを作成するには、「クラスターサブネットグループの作成」を参照してください。
- アベイラビリティーゾーンがダウンした場合、Amazon Redshift は、別のアベイラビリティーゾーンに新しい伸縮自在なネットワークインターフェイスを作成しません。この場合は、必要に応じて、新しいエンドポイントを作成します。

クォータと命名規則の詳細については、Amazon Redshift でのクォータと制限 を参照してください。

料金については、「AWS PrivateLink の料金」を参照してください。

VPC へのアクセス権の付与

クラスターまたはワークグループにアクセスする VPC が別の AWS アカウントにある場合は、必ず所有者 (付与者) アカウントから許可してください。

別の AWS アカウント内の VPC にクラスターまたはワークグループへのアクセスを許可するには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [クラスター] を選択します。Amazon Redshift Serverless の場合は、[サーバーレスダッシュボード] を選択します。

3. アクセスを許可するクラスターのクラスター名を選択して、クラスターの詳細を表示します。クラスターの [Properties] (プロパティ) タブを選択します。

   [Granted accounts] (付与されたアカウント) セクションには、クラスターにアクセスできるアカウントと対応する VPC が表示されます。Amazon Redshift Serverless ワークグループの場合は、ワークグループを選択します。[許可を付与されたアカウント] は [データアクセス] タブにあります。

4. アカウントを追加するために被付与者情報を入力するフォームを表示するには、[Grant access (アクセスの許可)] を選択します。

5. AWS アカウント ID に、アクセスを許可するアカウントの ID を入力します。特定のアカウント内の特定の VPC またはすべての VPC へのアクセス権を付与できます。

6. アクセスを許可するには、[Grant access] (アクセスの許可) を選択します。

Redshift で管理される VPC エンドポイントの作成

クラスターまたはワークグループを所有している場合、またはクラスターへの管理アクセス権が付与されている場合は、そのクラスターの Redshift 管理の VPC エンドポイントを作成できます。

Redshift で管理される VPC エンドポイントを作成するには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。

2. ナビゲーションメニューで、[Configurations] (設定) を選択します。

   [Configurations] (設定) ページには、作成された Redshift 管理の VPC エンドポイントが表示されます。エンドポイントの詳細を表示するには、その名前を選択します。Amazon Redshift Serverless の場合、ワークグループを選択すると VPC エンドポイントは [データアクセス] タブに表示されます。

3. [Create endpoint (エンドポイントの作成)] を選択して、追加するエンドポイントに関する情報を入力するためのフォームを表示します。

4. [エンドポイント名]、12 桁の [AWS アカウント ID]、エンドポイントが配置されている [仮想プライベートクラウド (VPC)]、[サブネット]、[VPC セキュリティグループ] の値を入力します。

   [サブネット] のサブネットグループは、Amazon Redshift がエンドポイントをデプロイするサブネットと IP アドレスを定義します。Amazon Redshift は、エンドポイントに関連付けられたネットワークインターフェイスで使用可能な IP アドレスを持つサブネットを選択します。

[VPC セキュリティグループ] のセキュリティグループルールは、エンドポイントに対して承認するインバウンドトラフィックのポート、プロトコル、およびソースを定義します。セキュリティグループまたは ワークロードが実行される CIDR 範囲を介して、選択したポートへのアクセスを許可します。

5. [Create endpoint] (エンドポイントの作成) を選択して、エンドポイントを作成します。

エンドポイントの作成後、Redshift 管理の VPC エンドポイント設定の [エンドポイント] URL に表示される URL で クラスタークラスターまたはワークグループにアクセスできます。

VPC での Redshift リソース

Amazon Redshift クラスターまたは Amazon Redshift Serverless ワークグループは、Amazon VPC サービスに基づいて EC2-VPC プラットフォームの VPC で起動できます。詳細については、「EC2 を使用してクラスターを作成する」を参照してください。

> **Note**
>
> 専用テナンシー VPC へのクラスターとサーバーレスワークグループの起動はサポートされていません。詳細については、Amazon VPC ユーザーガイドのハードウェア専有インスタンスを参照してください。

VPC にリソースをプロビジョニングするときは、次の操作を行う必要があります。

- VPC 情報を指定する

  VPC 内にプロビジョニングされたクラスターを作成するときは、クラスターサブネットグループを作成することにより、VPC 情報を提供する必要があります。この情報には、VPC の VPC ID とサブネットのリストが含まれます。クラスターの起動時に、Redshift が VPC 内のいずれかのサブネットにクラスターをプロビジョンできるように、クラスターサブネットグループを指定します。Amazon Redshift Serverless の場合もプロセスは似ています。サブネットはサーバーレスワークグループに直接割り当てます。ただし、サーバーレスの場合はサブネットグループを作成しません。Amazon Redshift でサブネットグループを作成する詳細方法については、Redshift リソースのサブネット を参照してください。VPC のセットアップの詳細については、「Amazon VPC 入門ガイド」の「Amazon VPC の開始方法」を参照してください。
- 必要に応じて、アクセシビリティオプションを設定します。

Amazon Redshift のプロビジョン済みクラスターとサーバーレスワークグループは、デフォルトでプライベートになっています。プロビジョン済みのクラスターまたはサーバーレスワークグループをパブリックにアクセスできるように設定する場合、Amazon Redshift は外部 IP アドレスに Elastic IP アドレスを使用します。Elastic IP アドレスは、静的 IP アドレスです。EIP を使用すると、クライアントが接続に使用する IP アドレスに影響を与えることなく、基本的な設定を変更することができます。このアプローチは、障害発生後の復旧などの状況に役立ちます。Elastic IP アドレスを作成するかどうかは、アベイラビリティーゾーンの再配置設定によって異なります。2 つのオプションがあります。

1. アベイラビリティーゾーンの再配置を有効にしていて、パブリックアクセスを有効にする場合は、Elastic IP アドレスを指定しないでください。Amazon Redshift によって管理される Elastic IP アドレスが割り当てられます。それは、AWS アカウントに関連付けられています。
2. アベイラビリティーゾーンの再配置が無効になっていて、パブリックアクセスを有効にしたい場合は、Amazon Redshift クラスターまたはワークグループを起動する前に、Amazon EC2 で VPC の Elastic IP アドレスを作成することを選択できます。IP アドレスを作成しない場合、Amazon Redshift は VPC に使用する設定済み Elastic IP アドレスを提供します。この Elastic IP アドレスは、Amazon Redshift によって管理され、AWS アカウントには関連付けられません。

詳細については、「Amazon EC2 ユーザーガイド」の「Elastic IP アドレス」を参照してください。

VPC にパブリックアクセス可能なクラスターがあり、VPC 内からプライベート IP アドレスを使って接続したい場合があります。その場合は、次の VPC パラメータを `true` に設定します。

- `DNS resolution`
- `DNS hostnames`

Amazon Redshift Serverless では、この方法では接続できないことに注意してください。

VPC にはパブリックアクセス可能なプロビジョニングされたクラスターがあるが、VPC でこれらのパラメータを `true` に設定しないとします。このような場合、VPC 内から確立された接続は、プライベート IP アドレスではなくリソースの Elastic IP アドレスに解決されます。VPC 内から接続する際には、これらのパラメータを `true` に設定し、パブリックにアクセス可能なクラスターにプライベート IP アドレスを使用することをお勧めします。詳細については、Amazon VPC ユーザーガイドの「VPC での DNS の使用」を参照してください。

> **Note**
>
> VPC 内にパブリックアクセス可能な既存のクラスターがある場合、VPC 内からの接続は、プロビジョニングされたクラスターのサイズを変更するまで、そのクラスターに接続するために Elastic IP アドレスを使用し続けます。これは、前述のパラメータセットでも発生します。新しく作成されたクラスターはすべて、パブリックアクセス可能なクラスターに同じ VPC 内から接続する際に、プライベート IP アドレスを使用するという新しい動作に従います。

Elastic IP アドレスは、VPC 外のリソースにアクセスするための外部 IP アドレスです。プロビジョニングされたクラスターでは、これは、[ノード IP アドレス] の下の Amazon Redshift コンソールに表示された [パブリック IP アドレス] と [プライベート IP アドレス] には関係ありません。パブリックおよびプライベートクラスターノードの IP アドレスは、クラスターがパブリックにアクセス可能であるかどうかに関係なく表示されます。これらのアドレスは、リモートホスト上の進入ルールを設定するため、特定の環境でのみ使用されます。これらの環境は、Secure Shell (SSH) 接続を使って Amazon EC2 インスタンスまたは他のリモートホストからデータをロードした場合に発生します。詳細については、Amazon Redshift データベースデベロッパーガイドの「ステップ 1: クラスター公開キーおよびクラスターノード IP アドレスを取得する」を参照してください。

> **Note**
>
> ノード IP アドレスは Redshift Serverless ワークグループには適用されません。

プロビジョニングされたクラスターを Elastic IP アドレスに関連付けるオプションは、クラスターを作成するか、スナップショットからクラスターを復元する場合に使用できます。場合によっては、クラスターを Elastic IP アドレスに関連付けたり、クラスターと関連付けられた Elastic IP アドレスを変更したりする必要が出てくることがあります。クラスターの作成後に Elastic IP アドレスをアタッチするには、まずクラスターを公開でアクセスできないように更新してから、公開でアクセスできるようにし、同じオペレーションで Elastic IP アドレスを追加します。

プロビジョニングされたクラスターまたは Amazon Redshift Serverless ワークグループをパブリックにアクセス可能にし、Elastic IP アドレスを割り当てる方法の詳細については、「デフォル

トまたはカスタムのセキュリティグループ設定によるパブリックアクセシビリティ」を参照してください。

- VPC セキュリティグループを関連付けます。

  VPC セキュリティグループを使用してインバウンドアクセスを許可します。詳細については、「Amazon Redshift クラスターのセキュリティグループ通信設定の設定」を参照してください。クライアントとプロビジョニングされたクラスターまたは Amazon Redshift Serverless ワークグループとの間のインバウンドルールとアウトバウンドルールの設定に関するガイダンスを記載しています。セキュリティグループの理解に役立つ別のリソースは、「Amazon VPC ユーザーガイド」の「VPC のセキュリティ」です。

VPC 内のプロビジョニングされたクラスターまたは Serverless ワークグループのスナップショットの復元

VPC 内のクラスターまたは Serverless ワークグループのスナップショットは、VPC の外部ではなく VPC の内部でのみ復元できます。これらは、アカウント内の同じ VPC または別の VPC 内で復元できます。 スナップショットの詳細については、「Amazon Redshift スナップショットとバックアップ」を参照してください。

VPC での Redshift でプロビジョンされたクラスターまたは Amazon Redshift Serverless ワークグループの作成

仮想プライベートクラウド (VPC) 内にクラスターまたはワークグループをデプロイするための通常のステップは、次のとおりです。

VPC でクラスターまたは Serverless ワークグループを作成するには

1. VPC の設定 – Redshift リソースは、アカウントのデフォルトの VPC 内 (アカウント内に VPC がある場合)、または作成した VPC 内に作成できます。詳細については、「EC2 を使用してクラスターを作成する」を参照してください。VPC を作成するには、「Amazon VPC ユーザーガイド」の「VPC のサブネット」を参照してください。VPC 識別子、サブネット、およびサブネットのアベイラビリティーゾーンを書きとめておきます。この情報は、クラスターまたはワークグループの起動時に必要です。

   > **Note**
   >
   > 次のステップでサブネットをサブネットグループに追加できるように、VPC 内に少なくとも 1 つのサブネットが定義されている必要があります。VPC へのサブネットの追加の

詳細については、Amazon VPC ユーザーガイドの「VPC へのサブネットの追加」を参照してください。

2. Amazon Redshift クラスターサブネットグループを作成して、VPC で Amazon Redshift クラスターが使用できるサブネットを指定します。Redshift Serverless の場合、サブネットグループを作成するのではなく、作成時にサブネットのコレクションをワークグループに割り当てます。これは、ワークグループの作成時に Serverless ダッシュボードで実行できます。

   サブネットグループは、Amazon Redshift コンソールを使用するか、またはプログラムで作成することができます。詳細については、「Redshift リソースのサブネット」を参照してください。

3. クラスターまたはワークグループに関連付ける VPC セキュリティグループ内のインバウンド接続に対して、アクセスを許可します。VPC (パブリックインターネット上) 外のクライアントを有効化して、クラスターを接続します。これを行うには、インバウンドアクセスを付与する VPC セキュリティグループにクラスターを関連付けます。詳細については、「Amazon Redshift クラスターまたは Amazon Redshift Serverless ワークグループのセキュリティグループ通信設定の構成」を参照してください。

4. 手順に従って、Redshift でプロビジョニングされたコンソールまたはワークグループ、あるいは Amazon Redshift Serverless コンソールでクラスターを作成します。[ネットワークとセキュリティ] の設定で、[仮想プライベートクラウド (VPC)]、[クラスターサブネットグループ]、および [VPC セキュリティグループ] を指定します。

   プロビジョニングされたデータウェアハウスクラスターを作成する詳細な手順を示すチュートリアルについては、「Amazon Redshift 入門ガイド」の「Amazon Redshift でプロビジョニングされたデータウェアハウスの使用を開始する」を参照してください。Amazon Redshift Serverless ワークグループの作成に関する詳しい情報については、「Amazon Redshift 入門ガイド」の「Amazon Redshift Serverless データウェアhガウスの使用開始」を参照してください。

入門ガイドのステップに従って、サンプルデータをアップロードし、サンプルクエリを試行することにより、クラスターまたはワークグループをテストできます。詳細については、「Amazon Redshift 入門ガイド」の「Amazon Redshift Serverless データウェアハウスの使用を開始」を参照してください。

VPC セキュリティグループ

Amazon Redshift クラスターまたは Amazon Redshift Serverless ワークグループをプロビジョニングすると、アクセスはデフォルトで制限されるため、誰もアクセスできません。他のユーザーにイン

バウンドアクセスを許可するには、クラスターをセキュリティグループに関連付けます。EC2-VPC プラットフォームを使用する場合は、既存の Amazon VPC セキュリティグループを使用するか、新しいグループを定義することができます。次に、下記の説明どおり、クラスターまたはワークグループに関連付けます。EC2-Classic プラットフォームでは、セキュリティグループを定義し、そのグループをクラスターまたはワークグループに関連付けます。EC2-Classic プラットフォームでのセキュリティグループの使用の詳細については、「Amazon Redshift セキュリティグループ」を参照してください。

VPC セキュリティグループは、使用中のクラスターなど、VPC 上のインスタンスへのアクセスを制御しているルールのセットで構成されます。個別のルールのセットは、IP アドレスの範囲または他の VPC セキュリティグループに基づいてアクセスします。VPC セキュリティグループをクラスターまたはワークグループに関連付けると、VPC セキュリティグループで定義されているルールがアクセスを制御します。

EC2-VPC プラットフォームでプロビジョンするクラスターごとに、1 つ以上の Amazon VPC セキュリティグループが関連付けられます。Amazon VPC は、デフォルトという VPC セキュリティグループを提供します。これは VPC を作成するときに自動的に作成されます。VPC で起動するクラスターは、Redshift のリソース作成時に別の VPC セキュリティグループを指定しない限り、それぞれがデフォルトの VPC セキュリティグループに自動的に関連付けられます。クラスターの作成時にクラスターに VPC セキュリティグループを関連付けるか、または後でクラスターを変更して VPC セキュリティグループを関連付けることができます。

次のスクリーンショットは、デフォルトの VPC セキュリティグループ用のデフォルトルールについて示しています。

Inbound			
Source	**Protocol**	**Port Range**	**Comments**
The security group ID (*sg-xxxxxxxx*)	All	All	Allow inbound traffic from instances assigned to the same security group
Outbound			
Destination	**Protocol**	**Port Range**	**Comments**
0.0.0.0/0	All	All	Allow all outbound traffic

必要に応じてデフォルトの VPC セキュリティグループのルールを変更できます。

デフォルトの VPC セキュリティグループだけで十分であれば、さらにグループを作成する必要はありません。ただし、オプションでインバウンドアクセスをより適切に管理するために、追加の VPC セキュリティグループを作成できます。例えば、Amazon Redshift クラスターまたは Serverless ワークグループでサービスを実行していて、顧客に複数の異なるサービスレベルを提供しているとします。すべてのサービスレベルに同じアクセス権を与えたくない場合は、サービスレベルごとに別の VPC セキュリティグループ (サービスレベルごとに 1 つ) を作成することをお勧めします。その後、

それらの VPC セキュリティグループをクラスターまたはワークグループに関連付けることができます。

VPC に対して最高 100 個の VPC セキュリティグループを作成し、VPC セキュリティグループを複数のクラスターおよびワークグループと関連付けることができます。ただし、クラスターまたはワークグループに関連付けることができる VPC セキュリティグループの数には制限があることに注意してください。

Amazon Redshift では、VPC セキュリティグループへの変更は直ちに適用されます。そのため、VPC セキュリティグループをクラスターに関連付けると、更新された VPC セキュリティグループのインバウンドクラスターアクセスルールは即座に適用されます。

VPC セキュリティグループの作成と変更は、https://console.aws.amazon.com/vpc/ で行うことができます。AWS CLI、Amazon EC2 CLI、および AWS Tools for Windows PowerShell を使用して、VPC セキュリティグループをプログラムで管理することもできます。VPC セキュリティグループの操作に関する詳細については、Amazon VPC ユーザーガイドの VPC のセキュリティグループを参照してください。

Amazon Redshift クラスターまたは Amazon Redshift Serverless ワークグループのセキュリティグループ通信設定の構成

このトピックは、ネットワークトラフィックを適切にルーティングおよび受信するようにセキュリティグループを設定するのに役立ちます。いくつかの一般的ユースケースは以下のとおりです。

- Amazon Redshift クラスターまたは Amazon Redshift Serverless ワークグループのパブリックアクセシビリティを有効にしても、トラフィックは受信されません。この場合、インターネットからのトラフィックを許可するようにインバウンドルールを設定する必要があります。
- クラスターまたはワークグループはパブリックにアクセスできないため、Redshift の設定済みのデフォルトの VPC セキュリティグループを使用してインバウンドトラフィックを許可します。ただし、要件に従ってデフォルト以外のセキュリティグループを使用している場合、このカスタムセキュリティグループはインバウンドトラフィックを許可しません。通信を許可するように設定する必要があります。

以下のセクションは、ユースケースごとに適した対応を選択するのに役立ち、要件に応じてネットワークトラフィックを設定する方法を示しています。オプションで、他のプライベートセキュリティグループからの通信を設定するための手順を使用できます。

> **Note**
>
> ほとんどの場合、ネットワークトラフィック設定は Amazon Redshift で自動的に設定されません。これは、トラフィックのソースがインターネットかプライベートセキュリティグループかによって設定の粒度が異なる場合があり、セキュリティ要件も異なるためです。

デフォルトまたはカスタムのセキュリティグループ設定によるパブリックアクセシビリティ

クラスターやワークグループを作成中であるか、作成済みである場合は、次の設定手順を実行してクラスターをパブリックにアクセスできるようにします。これは、デフォルトセキュリティグループを選択した場合とカスタムセキュリティグループを選択した場合の両方に当てはまります。

1. ネットワーク設定を検索します。
   - プロビジョニングされた Amazon Redshift クラスターの場合は、[プロパティ] タブを選択し、[ネットワークとセキュリティ設定] でクラスターの VPC を選択します。
   - Amazon Redshift Serverless ワークグループの場合は、[ワークグループの設定] を選択します。リストからワークグループを選択します。[ネットワークとセキュリティ] パネルの [データアクセス] で、[編集] を選択します。
2. VPC のインターネットゲートウェイとルートテーブルを設定します。VPC を名前で選択して設定を開始します。VPC ダッシュボードが開きます。インターネットからパブリックにアクセス可能なクラスターまたはワークグループに接続するには、インターネットゲートウェイをルートテーブルにアタッチする必要があります。これを設定するには、VPC ダッシュボードで [ルートテーブル] を選択します。インターネットゲートウェイのターゲットがソース 0.0.0.0/0、またはパブリック IP CIDR に設定されていることを確認します。ルートテーブルは、クラスターが存在する VPC に関連付ける必要があります。ここで説明しているような VPC のインターネットアクセス設定の詳細については、Amazon VPC ドキュメントで「インターネットアクセスを有効にする」を参照してください。ルートテーブルの設定の詳細については、「ルートテーブルの設定」を参照してください。
3. インターネットゲートウェイとルートテーブルを設定したら、Redshift のネットワーク設定に戻ります。セキュリティグループを選択し、次に [インバウンドルール] を選択して、インバウンドアクセスを開きます。インバウンドのルールを編集 を選択します。
4. インバウンドルールの [プロトコル] と [ポート] を要件に応じて選択し、クライアントからのトラフィックを許可します。RG または RA3 クラスターの場合は、5431 ~ 5455 または 8191 ~ 8215 の範囲内のポートを選択します。終了したら、各ルールを保存します。

5. [パブリックアクセス可能] 設定を編集して有効にします。この操作は、クラスターまたはワークグループの [アクション] メニューから行うことができます。

パブリックにアクセス可能な設定を有効にすると、Redshift は Elastic IP アドレスを作成します。これは AWS アカウントに関連付けられた静的 IP アドレスです。VPC 外部のクライアントはそれを使用して接続できます。

セキュリティグループの設定の詳細については、「Amazon Redshift セキュリティグループ」を参照してください。

クライアントに接続することでルールをテストできます。Amazon Redshift Serverless に接続する場合は、以下を実行します。ネットワーク設定が完了したら、Amazon Redshift RSQL などのクライアントツールを用いて接続します。Amazon Redshift Serverless のドメインをホストとして使用し、以下を入力します。

```
rsql -h workgroup-name.account-id.region.amazonaws.com -U admin -d dev -p 5439
```

デフォルトまたはカスタムのセキュリティグループ設定によるプライベートアクセシビリティ

インターネット経由でクラスターやワークグループと通信しない場合は、「プライベート」にアクセス可能と呼ばれます。作成時にデフォルトのセキュリティグループを選択した場合、セキュリティグループには以下のデフォルトの通信ルールが含まれます。

- セキュリティグループに割り当てられたすべてのリソースからのトラフィックを許可するインバウンドルール。
- すべてのアウトバウンドトラフィックを許可するアウトバウンドルール。このルールの宛先は 0.0.0.0/0 です。クラスレスドメイン間ルーティング (CIDR) 表記では、使用可能なすべての IP アドレスを表します。

クラスターまたはワークグループのセキュリティグループを選択すると、コンソールにルールを表示できます。

クラスターまたはワークグループとクライアントの両方がデフォルトのセキュリティグループを使用している場合は、ネットワークトラフィックを許可するための追加設定は必要ありません。ただし、Redshift またはクライアントのデフォルトのセキュリティグループのルールを削除または変更す

ると、これは適用されなくなります。この場合、インバウンドとアウトバウンドの通信を許可するルールを設定する必要があります。一般的なセキュリティグループ設定は次のとおりです。

- クライアントの Amazon EC2 インスタンスの場合:
  - クライアントの IP アドレスを許可するインバウンドルール。
  - Redshift 用に提供されているすべてのサブネットの IP アドレス範囲 (CIDR ブロック) を許可するアウトバウンドルール。または、0.0.0.0/0 を指定することもできます。これは、すべて IP アドレス範囲を表します。
- Redshift クラスターまたはワークグループの場合:
  - クライアントセキュリティグループを許可するインバウンドルール。
  - 0.0.0.0/0 へのトラフィックを許可するアウトバウンドルール。通常、アウトバウンドルールはすべてのアウトバウンドトラフィックを許可します。オプションで、クライアントセキュリティグループへのトラフィックを許可するアウトバウンドルールを追加できます。このオプションの場合、各リクエストのレスポンストラフィックがインスタンスに到達することが許可されるため、アウトバウンドルールは必ずしも必要ではありません。リクエストとレスポンスの動作の詳細については、「Amazon VPC ユーザーガイド」の「セキュリティグループ」を参照してください。

Redshift 用に指定されたサブネットまたはセキュリティグループの設定を変更した場合、通信を開いたままにしておくために、トラフィックルールの変更が必要になる場合があります。インバウンドルールとアウトバウンドルールの作成の詳細については、「Amazon VPC ユーザーガイド」の「VPC CIDR ブロック」を参照してください。クライアントから Amazon Redshift に接続する方法の詳細については、「Amazon Redshift での接続の設定」を参照してください。

Redshift リソースのサブネット

仮想プライベートクラウド (VPC) でプロビジョニングされたクラスターを作成する場合は、サブネットグループを作成します。各 VPC は、セキュリティおよびオペレーションのニーズに基づいたリソースのグループ化を可能にする VPC 内の IP アドレスのサブセットである 1 つまたは複数のサブネットを持つことができます。プロビジョニングされたクラスターを作成するときに、VPC 内のサブネットのセットを指定するサブネットグループを作成します。[プロビジョニングされたクラスターダッシュボード] では、[設定] でクラスターサブネットグループを検索および編集できます。プロビジョニングされたクラスターの初期設定時にサブネットグループを指定し、Amazon Redshift はそのサブネットの 1 つにクラスターを作成します。VPC サービスの詳細については、Amazon VPC の製品詳細ページを参照してください。

Amazon Redshift Serverless ワークグループのサブネット設定は、プロビジョニングされたクラスターと似ていますが、ステップは若干異なります。Serverless ワークグループを作成して設定するときは、ワークグループのサブネットを指定し、リストに追加されます。[Serverless ダッシュボード] でワークグループのプロパティを選択すると、既存のワークグループのサブネットを表示できます。これらは [ネットワークとセキュリティ] プロパティで使用できます。詳細については、「名前空間を伴うワークグループの作成」を参照してください。

VPC 作成の詳細については、「Amazon VPC ユーザーガイド」のドキュメントを参照してください。

プロビジョニングされたクラスターのサブネットグループを作成した後、または Serverless ワークグループのサブネットを選択すると、以前に追加されたサブネットを削除したり、さらに追加したりできます。これらの変更は、コンソールまたは API オペレーションを使用して行うことができます。プロビジョニングされたクラスターの API オペレーションの詳細については、「ModifyClusterSubnetGroup」を参照してください。Serverless ワークグループの API オペレーションについては、「UpdateWorkgroup」を参照してください。

サブネットグループのサブネットの 1 つでクラスターをプロビジョニングすることができます。クラスターサブネットグループにより、Virtual Private Cloud (VPC) 内にサブネットのセットを指定できます。

> **Warning**
>
> 従来のサイズ変更、一時停止と再開、マルチ AZ フェイルオーバー、その他のイベントなどのクラスターメンテナンスオペレーション中に、プロビジョニングされたコンピューティングノードが Amazon Redshift クラスターサブネットグループ内の別のサブネットに移動する場合があります。サブネットグループ内のすべてのサブネットには、同じネットワーク ACL インバウンドルールとアウトバウンドルール、および同じルートテーブルルートが必要です。これにより、Amazon Redshift コンピューティングリソースとの接続が保証されるため、こうしたメンテナンスイベントの後に最適に通信して機能することができます。ネットワーク ACL またはルートテーブル設定が異なるサブネットを同じ Amazon Redshift クラスターサブネットグループに追加することは避けてください。
> サブネットの設定に関する詳細については、「Amazon VPC ユーザーガイド」の「VPC のサブネット」を参照してください。Redshift マルチ AZ 配置の詳細については、Redshift 管理ガイドの「マルチ AZ デプロイ」を参照してください。クラスターのサイズ変更は、Redshift 管理ガイドでも説明されています。

クラスターサブネットグループの作成

以下の手順では、プロビジョニングされたクラスターのサブネットグループを作成する方法について説明します。VPC でクラスターをプロビジョニングするには、クラスターサブネットグループが少なくとも 1 つ定義されている必要があります。

プロビジョニングされたクラスターのクラスターサブネットグループを作成するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Configurations] (設定)、次に [Subnet groups] (サブネットグループ) を選択します。サブネットグループのリストが表示されます。
3. [クラスターサブネットグループの作成] を選択して作成ページを表示します。
4. 追加するサブネットを含む、サブネットグループの情報を入力します。
5. [クラスターサブネットグループの作成] を選択して、選択したサブネットでグループを作成します。

> **Note**
>
> サブネットのコレクションを使用して Amazon Redshift Serverless ワークグループを作成する方法については、「Amazon VPC ユーザーガイド」の「名前空間を伴うワークグループの作成」または「サブネットの作成」を参照してください。

クラスターサブネットグループの変更

サブネットグループを作成したら、Amazon Redshift コンソールでその情報を変更できます。以下の手順では、プロビジョニングされたクラスターのサブネットグループを変更する方法について説明します。

プロビジョニングされたクラスターのクラスターサブネットグループを修正するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Configurations] (設定)、次に [Subnet groups] (サブネットグループ) を選択します。サブネットグループのリストが表示されます。
3. 変更するサブネットグループを選択します。

4. [アクション] で [変更] を選択してサブネットグループの詳細を表示します。
5. サブネットグループの情報を更新します。
6. [保存] を選択してグループを変更します。

サブネットを変更または削除するために、追加の手順が必要な場合があります。例えば、この AWS ナレッジセンターの記事「How do I move my provisioned Amazon Redshift cluster into a different subnet?」(プロビジョニングされた Amazon Redshift クラスターを別のサブネットに移動する方法を教えてください) などでは、クラスターの移動を対象とするユースケースについて説明します。

プロビジョニングされたクラスターのクラスターサブネットグループの削除

クラスターサブネットグループの使用が完了したら、グループを削除してクリーンアップする必要があります。以下の手順では、プロビジョニングされたクラスターのサブネットグループを削除する手順について説明します。

クラスターサブネットグループを削除する方法

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで、[Configurations] (設定)、次に [Subnet groups] (サブネットグループ) を選択します。サブネットグループのリストが表示されます。
3. 削除するサブネットグループを選択して、[削除] を選択します。

> **Note**
>
> クラスターで使用されているクラスターサブネットグループは削除することができません。

VPC とサブネットへのパブリックアクセスをブロックする

VPC ブロックパブリックアクセス (BPA) は、AWS リージョンで所有している VPC とサブネットのリソースが、インターネットゲートウェイおよび送信専用 (Egress-Only) のインターネットゲートウェイを通じてインターネットに到達したり、インターネットから到達されたりしないよう一元的にブロックできるセキュリティ機能です。AWS アカウントでこの機能を有効にすると、デフォルトでは、Amazon Redshift が使用するすべての VPC またはサブネットに影響します。つまり、Amazon Redshift でパブリックオペレーションがすべて阻止されます。

VPC BPA を有効にしていて、パブリックインターネット経由で Amazon Redshift API を使用する場合は、VPC またはサブネットに対して Amazon EC2 API を使用するための除外を追加する必要があります。除外では、次のいずれかのモードを使用できます:

- [双方向]: 除外された VPC とサブネットとの間のすべてのインターネットトラフィックが許可されます。
- [エグレスのみ]: 除外された VPCとサブネットからのアウトバウンドインターネットトラフィックが許可されます。除外された VPC とサブネットに対するインバウンドインターネットトラフィックはブロックされます。これは、BPA が双方向に設定されている場合にのみ適用されます。

VPC BPA の除外設定では、VPC 全体または VPC 内の特定のサブネットをパブリックアクセス可能として指定します。その境界内のネットワークインターフェイスでは、パブリックインターネットへのルートとアクセス権の有無に関して、セキュリティグループ、ルートテーブル、ネットワーク ACL など、通常の VPC ネットワークコントロールが尊重されます。除外の追加の詳細については、「Amazon VPC ユーザーガイド」の「除外を作成および削除する」を参照してください。

プロビジョニング済みクラスター

サブネットグループとは、同じ VPC からの複数のサブネットを組み合わせたものです。プロビジョニング済みクラスターのサブネットグループが、VPC BPA が有効なアカウントにある場合、次の機能はブロックされます。

- パブリッククラスターを作成する
- パブリッククラスターを復元する
- プライベートクラスターをパブリックに変更する
- サブネットグループ内にパブリッククラスターが 1 つ以上ある場合に、VPC BPA が有効なサブネットをそのグループに追加する

Serverless クラスター

Redshift Serverless はサブネットグループを使用しません。代わりに、各クラスターに独自のサブネットセットがあります。VPC BPA が有効なアカウントにワークグループがある場合、次の機能はブロックされます。

- パブリックアクセスワークグループを作成する
- プライベートワークグループをパブリックに変更する

- ワークグループがパブリックの場合に、VPC BPA が有効なサブネットをそのワークグループに追加する

拡張 VPC ルーティングを使用して Redshift でネットワークトラフィックを制御する

Amazon Redshift の拡張された VPC のルーティングを使用すると、Amazon Redshift は Amazon VPC サービスに基づく Virtual Private Cloud (VPC) を介して、クラスターとデータリポジトリ間のすべての COPY と UNLOAD トラフィックを強制します。拡張 VPC ルーティングを使用することで、Amazon VPC ユーザーガイドに記載されているように、標準の VPC 機能を使用できます。これらの機能には、VPC セキュリティグループ、ネットワークアクセスコントロールリスト (ACL)、VPC エンドポイント、VPC エンドポイントポリシー、インターネットゲートウェイ、ドメインネームシステム (DNS) サーバーなどがあります。これらの機能を使用して、Amazon Redshift クラスターと他のリソースの間のデータフローを制御します。拡張された VPC ルーティングを使用して VPC からトラフィックをルーティングする場合には、VPC フローログを使って COPY と UNLOAD トラフィックを監視することもできます。

Amazon Redshift クラスターと Amazon Redshift Serverless ワークグループはどちらも高度な VPC ルーティングをサポートしています。Redshift Spectrum で拡張 VPC ルーティングを使用することはできません。詳細については、「拡張された VPC ルーティングを使用したデータレイクテーブルのクエリ」を参照してください。

拡張された VPC ルーティングが有効でない場合、Amazon Redshift は AWS ネットワーク内のその他のサービスなどへのトラフィックをインターネット経由でルーティングします。

> **Important**
>
> 拡張された VPC ルーティングはそのほかのリソースに Amazon Redshift がアクセスする方法に影響を与えるため、VPC を正しく設定しないと COPY と UNLOAD コマンドが失敗する場合があります。次に説明するように、クラスターの VPC とデータリソース間のネットワークパスを別に作成する必要があります。

拡張された VPC ルーティングが有効化されているクラスターで COPY または UNLOAD コマンドを実行すると、VPC が利用可能なネットワークパスで最も厳密、または最も限定的なものを使用して、指定されたリソースにトラフィックをルーティングします。

たとえば、VPC で次のパスを設定できます。

- VPC エンドポイント – クラスターまたはワークグループと同じ AWS リージョンの Amazon S3 バケットへのトラフィックで、バケットに直接トラフィックする VPC エンドポイントを作成できます。VPC のエンドポイントを使用すると、Amazon S3 へのアクセスを管理するためにエンドポイントのポリシーをアタッチできます。Redshift でのエンドポイントの使用に関する詳細は、「VPC エンドポイントを使用したデータベーストラフィックの制御」を参照してください。Lake Formation を使用する場合、VPC と AWS Lake Formation との間にプライベート接続を確立する方法の詳細については、「AWS Lake Formation とインターフェイス VPC エンドポイント (AWS PrivateLink)」を参照してください。

  > **Note**
  >
  > Redshift VPC エンドポイントを Amazon S3 VPC ゲートウェイエンドポイントと共に使用する場合は、Redshift で拡張 VPC のルーティングを有効にする必要があります。詳細については、「Amazon S3 のゲートウェイエンドポイント」を参照してください。

- NAT ゲートウェイ – 別の AWS リージョンにある Amazon S3 バケットと、AWS ネットワーク内にある別のサービスに接続できます。また、AWS ネットワーク外のホストインスタンスにアクセスすることもできます。これを行うには、Amazon VPC ユーザーガイドの説明に従って、ネットワークアドレス変換 (NAT) ゲートウェイを設定します。
- インターネットゲートウェイ – VPC の外にある AWS サービスに接続するには、Amazon VPC ユーザーガイドの説明通りに、インターネットゲートウェイを VPC サブネットにアタッチします。インターネットゲートウェイを使用するには、クラスターまたはワークグループは、他のサービスとの通信を可能にするためにアクセスできる必要があります。

詳細については、Amazon VPC ユーザーガイドの「VPC エンドポイント」を参照してください。

> **Note**
>
> 拡張 VPC ルーティングは、ユーザーのサインイン時に Redshift が AWS IAM アイデンティティセンターに対して行う呼び出しにも影響します。クラスターまたはワークグループが AWS IAM アイデンティティセンター認証を使用している場合は、これらのサービス用のインターフェイス VPC エンドポイントを作成する必要があります。そうしないと、サインインは失敗します。拡張 VPC ルーティングと AWS IAM アイデンティティセンターの使用の詳細については、「拡張 VPC ルーティングでの AWS IAM アイデンティティセンター認証の使用」を参照してください。

拡張 VPC ルーティングは追加料金なしで使用できます。特定の操作では、追加のデータ転送料金が発生する可能性があります。これには、異なる AWS リージョンでの Amazon S3 への UNLOAD などのオペレーションが含まれます。Amazon EMR から COPY、またはパブリック IP アドレスを持つ Secure Shell (SSH)。料金の詳細については、「Amazon EC2 料金表」を参照してください。

トピック

- VPC エンドポイントを使用したデータベーストラフィックの制御
- 拡張 VPC ルーティングを有効にする
- 拡張された VPC ルーティングを使用したデータレイクテーブルのクエリ

VPC エンドポイントを使用したデータベーストラフィックの制御

VPC エンドポイントを使用して、VPC にある Amazon Redshift クラスターまたは Serverless ワークグループと Amazon Simple Storage Service (Amazon S3) 間でマネージド接続を作成することができます。このとき、データベースと Amazon S3 のデータ間の COPY と UNLOAD トラフィックは Amazon VPC に留まります。エンドポイントにエンドポイントポリシーをアタッチして、より厳密なデータアクセスの管理ができます。たとえば、アカウントの特定の Amazon S3 バケットにのみデータをアップロード許可するようなポリシーを VPC エンドポイントに追加できます。

VPC エンドポイントを使用するには、データウェアハウスがある VPC への VPC エンドポイントを作成し、それから拡張された VPC ルーティングを有効化します。クラスターまたはワークグループを作成する場合、拡張された VPC ルーティングを有効にするか、あるいは VPC のクラスターまたはワークグループを変更して拡張された VPC ルーティングとして使用することもできます。

VPC エンドポイントは、ルートテーブルを使用して VPC のクラスターまたはワークグループと Amazon S3 間のトラフィックのルーティングをコントロールします。特定のルートテーブルに関連付けられたサブネットのすべてのクラスターまたはワークグループは、自動的にそのエンドポイントを使用してサービスにアクセスします。

VPC はトラフィックをルーティングするために、トラフィックに合う最も特定された、または最も限定的なルートを使用します。たとえば、ルートテーブルで、インターネットゲートウェイと Amazon S3 エンドポイントを指すすべてのインターネットトラフィック (0.0.0.0/0) にルートがあるとします。この場合、Amazon S3 を宛先とするすべてのトラフィックでエンドポイントルートが優先されます これは、Amazon S3 サービスの IP アドレス範囲が 0.0.0.0/0 よりも具体的であるためです。この例では、他の AWS リージョン にある Amazon S3 バケット行きのトラフィックを含め、その他のすべてのインターネットトラフィックはインターネットゲートウェイに移動します。

エンドポイントの作成の詳細については、Amazon VPC ユーザーガイドの「VPC エンドポイントの作成」を参照してください。

クラスターまたはワークグループからデータファイルがある Amazon S3 バケットへのアクセスを制御するために、エンドポイントポリシーを使用します。特定の制御については、オプションでカスタムエンドポイントポリシーをアタッチできます。詳細については、「AWS PrivateLink ガイド」の「Control access to services using endpoint policies」を参照してください。

> **Note**
>
> AWS Database Migration Service (AWS DMS) は、リレーショナルデータベース、データウェアハウス、その他のタイプのデータストアを移行できるようにするクラウドサービスです。VPC が有効になっている Amazon Redshift データベースなど、任意の AWS ソースまたはターゲットデータベースに接続できますが、いくつかの設定制限があります。Amazon VPC エンドポイントのサポートにより、AWS DMS は、レプリケーションタスクにおいてエンドツーエンドのネットワークセキュリティを簡単に維持できます。AWS DMS での Redshift の使用の詳細については、「AWS Database Migration Service ユーザーガイド」の「VPC エンドポイントを AWS DMS ソースエンドポイントとターゲットエンドポイントとして設定する」を参照してください。

エンドポイントは追加料金なしで使用できます。データ転送とリソースの使用量に対する標準料金が適用されます。料金の詳細については、「Amazon EC2 料金表」を参照してください。

拡張 VPC ルーティングを有効にする

クラスターを作成または変更するとき、および Amazon Redshift Serverless ワークグループを作成または変更するときに、拡張された VPC ルーティングを有効にできます。

拡張された VPC ルーティングを操作するためには、クラスターまたは Serverless ワークグループが次の要件と制約を満たしている必要があります。

- クラスターは VPC 内にある必要があります。

  Amazon S3 VPC エンドポイントをアタッチする場合、VPC エンドポイントは同じ AWS リージョンにある Amazon S3 バケットへのアクセスのためだけに使用します。他の AWS リージョンでバケットにアクセスしたり (VPC エンドポイントを使用せずに)、他の AWS サービスにアクセスするには、クラスターまたはServerless ワークグループのパブリックアクセスを有効にするか、ネットワークアドレス変換 (NAT) ゲートウェイを使用します。詳細については、「VPC での Redshift

でプロビジョンされたクラスターまたは Amazon Redshift Serverless ワークグループの作成」を参照してください。

- VPC でドメインネームサービス (DNS) 解決を有効にする必要があります。または、自分で所有する DNS サーバーを使用している場合は、Amazon S3 に送られる DNS リクエストが AWS により維持される IP アドレスに正しく変換されていることを確認する必要があります。詳細については、Amazon VPC ユーザーガイドの VPC での DNS の使用を参照してください。
- DNS ホスト名を VPC で有効にする必要があります。DNS ホスト名はデフォルトで有効化されています。
- VPC エンドポイントポリシーは、すべてのマニフェストファイル関連へのアクセスを含む Amazon Redshift で呼び出される COPY、UNLOAD、CREATE LIBRARY で使用されるすべての Amazon S3 バケットへのアクセスを許可している必要があります。リモートホストからの COPY では、エンドポイントポリシーが各ホストマシンへのアクセスを許可している必要があります。詳細については、Amazon Redshift データベースデベロッパーガイドの「COPY、UNLOAD、CREATE LIBRARY のための IAM 許可」を参照してください。

プロビジョニングされたクラスターの拡張された VPC ルーティングを有効にするには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [Provisioned Clusters] (プロビジョニングされたクラスター)、[Create cluster] (クラスターを作成) の順に選択し、[Cluster details] (クラスターの詳細) プロパティを入力します。
3. [Additional configurations (追加設定)] セクションを表示するには、[Use defaults (デフォルトの使用)] をオフにします。
4. [Network and security] (ネットワークとセキュリティ) セクションに移動します。
5. [Enhanced VPC routing] (拡張された VPC のルーティング)を有効化するには、[Turn on] (有効) を選択してクラスタートラフィックが VPC を経由するよう強制します。
6. [クラスターを作成] を選択して、クラスターを作成します。クラスターの使用準備ができるまで、数分かかることがあります。

Amazon Redshift Serverless の拡張された VPC ルーティングを有効化するには

1. AWS マネジメントコンソールにサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。

2. ナビゲーションメニューで [Serverless dashboard] (サーバーレスダッシュボード) を選択し、次に [Create workgroup] (ワークグループの作成)を選択して、ワークグループのプロパティを入力します。
3. [Network and security] (ネットワークとセキュリティ) セクションに移動します。
4. ネットワークトラフィックを VPC 経由でルーティングするには、[Turn on enhanced VPC routing] (拡張された VPC ルーティングを有効にする) を選択します。
5. [Next] (次へ) を選択し、ワークグループプロパティを入力して、ワークグループを作成します。

拡張された VPC ルーティングを使用したデータレイクテーブルのクエリ

拡張された VPC のルーティングがオンになっていると、Amazon Redshift は、トラフィックを VPC 経由でルーティングします。データレイクテーブルは、データを Amazon S3 に保存し、メタデータを AWS Glue Data Catalog に保存します。どちらも VPC の外部にあります。データレイククエリを成功させるには、クラスターまたはワークグループが Amazon S3 と AWS Glue に到達できる必要があります。データレイクテーブルが AWS Lake Formation によって管理されている場合は、Lake Formation に到達できる必要もあります。拡張された VPC ルーティングは、Amazon Redshift が外部リソースにアクセスする方法に影響を与えるため、VPC を正しく設定しないとクエリが失敗する場合があります。

クラスターまたはワークグループが実行されている VPC およびサブネットに次の VPC エンドポイントを作成し、適切なルートテーブルに関連付けます。これらのエンドポイントは、統合されたデータレイククエリエンジンと Amazon Redshift Spectrum の両方に必要です。

- Amazon S3 ゲートウェイエンドポイント – データレイクのデータファイルを読み取るために、クラスターまたはワークグループに AWS ネットワーク経由で Amazon S3 へのルートを提供します。詳細については、「Amazon VPC ユーザーガイド」の「Amazon S3 のゲートウェイエンドポイント」を参照してください。
- AWS Glue インターフェイスエンドポイント – データレイクスキーマとテーブルを解決するために、クラスターまたはワークグループに AWS Glue Data Catalog へのルートを提供します。手順については、「Amazon VPC ユーザーガイド」の「インターフェイスエンドポイントの作成」を参照してください。
- AWS Lake Formation インターフェイスエンドポイント – データレイクテーブルが AWS Lake Formation によって管理されている場合にのみ必要です。クラスターまたはワークグループに Lake Formation へのルートを提供します。手順については、「Amazon VPC ユーザーガイド」の「インターフェイスエンドポイントの作成」を参照してください。

エンドポイント設定は、外部テーブルのクエリ方法に関係なく同じですが、トラフィックのルーティング方法はクエリエンジンによって異なります。以下のセクションでは、RG プロビジョニングクラスターと Amazon Redshift Serverless の統合データレイククエリエンジン、および RA3 と DC2 プロビジョニングクラスターの Redshift Spectrum の動作について説明します。

RG プロビジョン済みクラスターと Amazon Redshift Serverless の統合データレイククエリエンジン

RG プロビジョン済みクラスターと Amazon Redshift Serverless には、統合データレイククエリエンジンが含まれています。このエンジンは、VPC 内で、クラスターまたはワークグループ独自のコンピューティングリソースで実行されます。拡張 VPC ルーティングがオンになっており、必要な VPC エンドポイントが設定されている場合、Amazon S3 と AWS Glue へのアクセスは VPC 内コンピューティングから開始され、これらの VPC エンドポイントを流れます。このトラフィックは、VPC 境界内および AWS ネットワーク上にとどまります。このアクセスは、クラスターまたはワークグループにアタッチされている IAM ロールに基づいて承認されます。このトラフィックをさらに管理するには、IAM ロールと、Amazon S3 バケットに添付されているポリシーを変更します。

RA3 および DC2 でプロビジョニングされたクラスターの Amazon Redshift Spectrum

Redshift Spectrum は、Amazon Redshift が所有する AWS 管理のリソースで実行されますが、これは VPC の外にあります。その結果、拡張 VPC ルーティングが有効になっていても、Redshift Spectrum が Amazon S3 に送信するトラフィックは VPC を通過しませんが、クラスターがデータレイククエリを実行するには、前述の Amazon S3 と AWS Glue VPC エンドポイントを必要とします。

Redshift Spectrum と Amazon S3 間のトラフィックは、VPC の外部で AWS プライベートネットワークを介して安全にルーティングされます。処理中のトラフィックは Amazon Signature Version 4 プロトコル (SIGv4) を使用して署名され、HTTPS を使用して暗号化されます。このトラフィックは、Amazon Redshift クラスターに添付されている IAM ロールに基づいて承認されます。Redshift Spectrum トラフィックをさらに管理するには、クラスターの IAM ロールと、Amazon S3 バケットに添付されているポリシーを変更します。

データレイククエリのアクセス許可ポリシー設定

Amazon S3 のデータレイクテーブルをクエリする場合は、次の点を考慮してください。

- Amazon S3 バケットアクセスポリシーと IAM ロール
- IAM ロールを引き受けるためのアクセス許可

- Amazon S3 アクセスのログ記録と監査
- AWS Glue または Amazon Athena へのアクセス

Amazon S3 バケットアクセスポリシーと IAM ロール

バケットに添付されたバケットポリシーを使用、およびプロビジョニングされたクラスターにアタッチされた IAM ロールを使用することで、Amazon S3 バケットにあるデータへのアクセスを管理できます。

プロビジョニングされたクラスターの Redshift Spectrum は、指定された VPC エンドポイントへのアクセスのみを制限するバケットポリシーを使用する Amazon S3 バケットに格納されているデータにはアクセスできません。代わりに、特定の AWS アカウントやユーザーなど、特定のプリンシパルへのアクセスのみを制限するバケットポリシーを使用します。RG でプロビジョニングされたクラスターと Amazon Redshift Serverless の統合データレイククエリエンジンは、拡張 VPC ルーティングが有効になっている場合、Amazon S3 へのアクセスが VPC エンドポイントを介して流れるため、このようなバケットにアクセスできます。

バケットへのアクセス権限を付与された IAM ロールには、Amazon Redshift サービスプリンシパルのみがロールを引き受けることを許可する信頼関係を使用します。ロールがクラスターに添付されると、ロールは Amazon Redshift のコンテキストでのみ使用でき、クラスター外で共有することはできません。詳細については、「IAM ロールへのアクセスの制限」を参照してください。サービスコントロールポリシー (SCP) を使用して、ロールをさらに制限することもできます。「AWS Organizations ユーザーガイド」の「特定の管理者ロールを除いて、IAM ユーザーとロールが特定の変更を行わないようにする」を参照してください。

> **Note**
>
> Redshift Spectrum を使用する場合は、Amazon S3 の署名付き URL の使用をブロックする IAM ポリシーを設定できません。Amazon Redshift Spectrum によって生成された署名付き URL は 1 時間有効です。これにより、Amazon Redshift は Amazon S3 バケットからすべてのファイルをロードするのに十分な時間を確保できます。Redshift Spectrum によってスキャンされたファイルごとに一意の署名付き URL が生成されます。`s3:signatureAge` アクションを含むバケットポリシーの場合は、値を少なくとも 3,600,000 ミリ秒に設定してください。

次のバケットポリシーの例では、AWS アカウント (123456789012) が所有する、指定されたバケットへのアクセスを許可します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "BucketPolicyForSpectrum",
            "Effect": "Allow",
            "Principal": {
                "AWS": ["arn:aws:iam::123456789012:role/redshift"]
            },
            "Action": [
                "s3:GetObject",
                "s3:ListBucketVersions",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        }
    ]
}
```

IAM ロールを引き受けるためのアクセス許可

クラスターに添付されたロールには、次に示すように、Amazon Redshift サービスによってのみ引き受けられることを許可する信頼関係があります。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "redshift.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
```

```
  ]
}
```

詳細については、「Amazon Redshift データベース開発者ガイド」の「Redshift Spectrum 用 IAM ポリシー」を参照してください。

Amazon S3 アクセスのログ記録と監査

Amazon Redshift 拡張された VPC のルーティングを使用する利点の 1 つは、すべての COPY および UNLOAD トラフィックが VPC フローログに記録されることです。Redshift Spectrum から Amazon S3 へのトラフィックは VPC を通過しないため、VPC フローログに記録されません。Redshift Spectrum が Amazon S3 にあるデータにアクセスする際、AWS アカウントとそれぞれのロール権限のコンテキストでこれらのオペレーションが実行されます。AWS CloudTrail と Amazon S3 でサーバーアクセスのログ記録を使用して、Amazon S3 のアクセスをログ記録し、監査を行えます。

S3 IP 範囲が許可リストに追加されていることを確認します。必要な S3 IP 範囲の詳細については、「ネットワークの隔離」を参照してください。

AWS CloudTrail ログ

Redshift Spectrum を含め、Amazon S3 にあるオブジェクトへのすべてのアクセスを追跡するには、Amazon S3 オブジェクトの CloudTrail ログ記録を有効化します。

CloudTrail を使用することで、AWS インフラストラクチャ全体のアカウントアクティビティを表示、検索、ダウンロード、アーカイブ、分析し、それに対応できます。詳細については、「CloudTrail の開始方法」を参照してください。

デフォルトでは、CloudTrail はバケットレベルのアクションのみを追跡します。オブジェクトレベルのアクション (`GetObject` など) を追跡するには、ログが記録されたバケットそれぞれのデータイベントおよび管理イベントを有効化します。

Amazon S3 サーバーアクセスログ

サーバーアクセスのログには、バケットに対するリクエストの詳細が記録されます。アクセスログ情報は、セキュリティとアクセス監査に役立ちます。詳細については、Amazon Simple Storage Service ユーザーガイドのサーバーアクセスのログ記録を有効にする方法を参照してください。

詳細については、AWS セキュリティブログの記事「How to Use Bucket Policies and Apply Defense-in-Depth to Help Secure Your Amazon S3 Data」を参照してください。

AWS Glue または Amazon Athena へのアクセス

Redshift Spectrum と統合されたデータレイククエリエンジンの両方が、AWS Glue または Athena のデータカタログにアクセスします。データカタログの専用 Hive メタストアを使用してアクセスすることもできます。

AWS Glue Data Catalog に到達するには、AWS Glue のインターフェイス VPC エンドポイント (AWS PrivateLink) を作成することをお勧めします。VPC インターフェイスエンドポイントを使用すると、AWS ネットワーク内で VPC と AWS Glue 間の通信がルーティングされます。詳細については、インターフェイスエンドポイントの作成を参照してください。

または、AWS Glue および Athena のパブリックエンドポイントに到達するには、インターネットゲートウェイまたは NAT ゲートウェイを使用して VPC を設定し、それらのパブリックエンドポイントへのアウトバウンドトラフィックを許可するように VPC セキュリティグループを設定します。VPC で次のパスを設定できます。

- インターネットゲートウェイ – VPC の外にある AWS サービスに接続するには、Amazon VPC ユーザーガイドの説明通りに、インターネットゲートウェイを VPC サブネットにアタッチします。インターネットゲートウェイを使用するには、プロビジョニングされたクラスターは他のサービスとの通信を可能にするパブリック IP アドレスが必要です。
- NAT ゲートウェイ – 別の AWS リージョンの Amazon S3 バケットまたは AWS ネットワーク内の別のサービスに接続するには、Amazon VPC ユーザーガイドの説明通りに、ネットワークアドレス変換 (NAT) ゲートウェイを設定します。この設定を使用して、AWS ネットワーク外のホストインスタンスにアクセスすることもできます。

Amazon Redshift のイベント

Amazon Redshift はクラスターイベントを追跡し、その情報を AWS アカウントに数週間保持します。Amazon Redshift はイベントごとに、イベントが発生した日付、説明、イベントのソース (クラスター、パラメータグループ、またはスナップショット)、ソース ID などの情報を報告します。

Amazon Redshift では、いくつかのイベントを事前に通知します。これらのイベントは、pendingのイベントカテゴリーを持ちます。例えば、クラスターにあるノードの 1 つに更新が必要な場合、事前通知を送信します。他の Amazon Redshift イベントと同様、保留中のイベントにサブスクライブできます。詳細については、「Amazon Redshift クラスターイベント通知のサブスクリプション」を参照してください。

Amazon Redshift マネジメントコンソール、Amazon Redshift API、または AWS SDK を使って、イベント情報を取得できます。すべてのイベントのリストを取得することも、フィルター (イベント期間、開始日と終了日など) を適用して、特定期間のイベント情報を取得することもできます。

また、特定のソースタイプによって生成されたイベント (クラスターイベントやパラメータグループイベントなど) を取得することもできます。[ソース] 列には、特定のアクションをトリガーするリソースの名前とタイプが表示されます。

一連のイベントフィルターを指定する、Amazon Redshift イベント通知サブスクリプションを作成できます。フィルター条件に一致するイベントが発生すると、Amazon Redshift は Amazon Simple Notification Service を使用してイベントが発生したことをアクティブに通知します。

ソースタイプおよびカテゴリ別の Amazon Redshift イベントのリストについては、「the section called “プロビジョニングされたクラスターのイベント通知”」を参照してください。

Amazon Redshift クラスターイベント通知のサブスクリプション

Amazon Redshift では、Amazon Simple Notification Service (Amazon SNS) を使用して、Amazon Redshift イベントの通知を送信します。通知は、Amazon Redshift イベントサブスクリプションを作成して有効にします。特定のクラスター、スナップショット、セキュリティグループ、パラメータグループでイベントが発生したときに通知を受け取ることができます。サブスクリプションを作成する最も簡単な方法は、Amazon SNS コンソールを使用する方法です。Amazon SNS トピックの作成とサブスクライブについては、Amazon SNS の開始方法を参照してください。

Amazon Redshift サブスクリプションでは、Amazon Redshift イベントと Amazon SNS トピックの一連のフィルターを指定します。フィルター条件に一致するイベントが発生するたびに、Amazon Redshift は Amazon SNS トピックに通知メッセージを発行します。

次に、Amazon SNS は、トピックへの Amazon SNS サブスクリプションを持っている Amazon SNS コンシューマにメッセージを送信します。Amazon SNS ユーザーに送信されるメッセージは、AWS リージョンの Amazon SNS によりサポートされているいずれかの形式にすることができます (メール、テキストメッセージ、HTTP エンドポイントへの呼び出しなど)。例えば、すべてのリージョンでメール通知をサポートしていますが、SMS 通知は米国東部 (バージニア北部) リージョンでのみ作成できます。

> **Note**
>
> 現在、イベントサブスクリプションは Amazon SNS 標準トピックに対してのみ作成できます (Amazon SNS FIFO トピックに対しては作成できません)。詳細については、「Amazon Simple Notification Service デベロッパーガイド」の「Amazon SNS イベントソース」を参照してください。

イベント通知サブスクリプションを作成するときは、1 つ以上のイベントフィルターを指定します。Amazon Redshift は、すべてのフィルター基準に一致するイベントが発生するたびに、サブスクリプションを通じて通知を送信します。フィルタ条件には、ソースのタイプ (クラスターやスナップショットなど)、ソース ID (クラスターまたはスナップショットの名前など)、イベントカテゴリ (モニタリングやセキュリティなど)、イベントの重大度 (INFO や ERROR など) が含まれます。

CLI または API を使用してイベント通知サブスクリプションを作成する場合、Amazon Simple Notification Service トピックを作成し、Amazon SNS コンソールまたは Amazon SNS API を使用してそのトピックにサブスクライブする必要があります。トピックの Amazon Resource Name (ARN) は、CLI コマンドや API アクションを送信するときに使用されるため維持する必要があります。

AWS マネジメントコンソール で [有効] ラジオボタンを [No] に設定するか、Amazon Redshift CLI または API を使用して `Enabled` パラメータを `false` に設定することで、サブスクリプションを削除せずに通知を無効にできます。

Amazon Redshift イベントサブスクリプションは、これらのイベント条件を指定できます。

- ソースタイプの値は、クラスター、スナップショット、パラメータグループ、およびセキュリティグループです。
- リソースのソース ID (`my-cluster-1` や `my-snapshot-20130823` など)。ID は、イベントサブスクリプションと同じ AWS リージョンのリソースの ID にする必要があります。
- イベントカテゴリの値は、Configuration、Management、Monitoring、Security および Pending です。

- イベントの重大度の値は、INFO または ERROR です。

イベント条件は、別個に指定できますが、コンソールでソース ID を指定するにはソースタイプを指定する必要があります。たとえば、ソースタイプ、ソース ID、重大度を指定しなくてもイベントカテゴリを指定できます。ソースタイプで指定したタイプではないリソースのソース ID を指定できますが、それらのリソースからのイベントには通知が送信されません。たとえば、クラスターのソースタイプとセキュリティグループの ID を指定した場合、そのセキュリティグループが生じたどのイベントも、ソースタイプのフィルタ条件に一致しないので、それらのイベントに通知は送信されません。

Amazon Redshift は、サブスクリプションで指定されているすべての条件に一致するすべてのイベントに通知を送信します。返されるイベントセットの例

- サブスクリプションは、クラスターのソースタイプ、ソース ID my-cluster-1、カテゴリ Monitoring、重大度 ERROR を指定します。サブスクリプションは、my-cluster-1 から生じた、重大度が ERROR のモニタリングイベントにのみ通知を送信します。
- サブスクリプションは、クラスターのソースタイプ、カテゴリ Configuration、INFO の重大度を指定します。サブスクリプションは、AWS アカウントの Amazon Redshift クラスターから生じた、重大度が INFO の設定イベントに通知を送信します。
- サブスクリプションは、カテゴリ Configuration と重大度 INFO を指定します。サブスクリプションは、AWS アカウントの Amazon Redshift リソースから生じた、重大度が INFO の設定イベントに通知を送信します。
- サブスクリプションは、重大度 ERROR を指定します。サブスクリプションは、AWS アカウントの Amazon Redshift リソースから生じた、重大度が ERROR のすべてのイベントに通知を送信します。

既存のサブスクリプションで名前がソース ID として参照されているオブジェクトを削除または名前変更する場合、サブスクリプションはアクティブのままになりますが、そのオブジェクトから転送されるイベントはありません。サブスクリプションソース ID で参照されるのと同じ名前の新しいオブジェクトを後で作成する場合、サブスクリプションは新しいオブジェクトから生じたイベントの通知の送信を開始します。

Amazon Redshift は、Amazon リソースネーム (ARN) によって識別される Amazon SNS トピックにイベントの通知を発行します。Amazon Redshift コンソールを使用してイベントサブスクリプションを作成するときは、既存の Amazon SNS トピックを指定するか、サブスクリプションの作成時にコンソールがトピックを作成することリクエストすることができます。

Amazon SNS トピックに送信されたすべての Amazon Redshift イベント通知は、そのトピックにサブスクライブしているすべての Amazon SNS ユーザーに順番に送信されます。Amazon SNS コンソールを使用して Amazon SNS トピックに変更を加えます (トピックへのユーザーサブスクリプションの追加や削除など)。

以下のセクションに、通知を受け取ることができるすべてのカテゴリとイベントを示します。さらに、Amazon Redshift のイベントサブスクリプションにサブスクライブして操作する方法に関する情報も示します。

イベント通知サブスクリプションを作成する

Amazon Simple Notification Service (Amazon SNS) イベント通知サブスクリプションを作成すると、特定の Amazon Redshift クラスター、スナップショット、セキュリティグループ、またはパラメータグループにイベントが発生したときに通知を送信できます。これらの通知は SNS トピックに送信され、トピックにサブスクライブした SNS ユーザーに順番にメッセージが送信されます。

ユーザーへの SNS メッセージは、AWS リージョンの Amazon SNS によりサポートされているいずれかの形式にすることができます (メール、テキストメッセージ、HTTP エンドポイントへの呼び出しなど)。たとえば、すべてのリージョンでメール通知がサポートされますが、SMS 通知は 米国東部 (バージニア北部) リージョンでのみ作成できます。詳細については、「Amazon Redshift プロビジョニングされたクラスターのイベント通知」を参照してください。

イベントサブスクリプションを作成するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [Events] (イベント) を選択します。
3. [イベントサブスクリプション] タブを選択してから、[イベントサブスクリプションの作成] を選択します。
4. 名前、ソースタイプ、カテゴリ、重要度など、イベントサブスクリプションのプロパティを入力します。また、Amazon SNS トピックを有効化して、イベントの通知を受け取ることもできます。
5. [イベントサブスクリプションの作成] を選択してサブスクリプションを作成します。

Amazon Redshift プロビジョニングされたクラスターのイベント通知

このページには、各 Amazon Redshift ソースタイプのイベント ID とカテゴリが表示されます。

クラスターソースタイプのカテゴリとイベント

次の表は、クラスターがソースタイプである場合のイベントカテゴリとイベントのリストを示します。

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
設定	REDSHIFT-EVENT-1000	INFO	パラメータグループ [パラメータグループ名] が [時間] に更新されました。動的パラメータのみを変更した場合は、関連付けられているクラスターが現在変更中です。静的パラメータを変更した場合、動的パラメータを含む、すべての更新は、関連するクラスターを再起動した場合に適用されます。
設定	REDSHIFT-EVENT-1001	INFO	Amazon Redshift クラスター [cluster name] が、[time] にパラメータグループ [parameter group name] を使用するように変更されました。
設定	REDSHIFT-EVENT-1500	ERROR	Amazon VPC [VPC name] が存在しません。クラスター [クラスター名] の設定の変更が適用されませんでした。AWS マネジメントコンソール にアクセスして問題を修正してください。
設定	REDSHIFT-EVENT-1501	ERROR	Amazon VPC [VPC name] に指定した顧客サブネット [subnet name] が存在しないか、無効です。クラスター [クラスター名] の設定の変更が適用されませんでした。AWS マネジメ

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
			ントコンソール にアクセスして問題を修正してください。
設定	REDSHIFT-EVENT-1502	ERROR	クラスターサブネットグループ [サブネットグループ名] のサブネットに使用可能な IP アドレスがありません。クラスター [クラスター名] を作成できませんでした。
設定	REDSHIFT-EVENT-1503	ERROR	Amazon VPC [VPC name] にインターネットゲートウェイが接続されていません。クラスター [クラスター名] の設定の変更が適用されませんでした。AWS マネジメントコンソールにアクセスして問題を修正してください。
設定	REDSHIFT-EVENT-1504	ERROR	クラスター [クラスター名] の HSM に到達できません。
設定	REDSHIFT-EVENT-1505	ERROR	クラスター [クラスター名] の HSM を登録できません。別の設定を試してください。
設定	REDSHIFT-EVENT-1506	ERROR	Amazon Redshift は、アカウントの Elastic Network Interface (ENI) 制限を超えています。[Elastic Network Interface の最大数] までの Elastic Network Interface を削除するか、EC2 での AWS リージョンあたりのネットワークインターフェイスの数の制限の引き上げをリクエストします。

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
設定	REDSHIFT-EVENT-1509	ERROR	アカウントの VPC エンドポイントの制限に達したため、Amazon Redshift クラスター [cluster name] を作成できません。未使用の VPC エンドポイントを削除するか、VPC エンドポイント制限の引き上げをリクエストします。 詳細については、Amazon VPC ユーザーガイドの「VPC エンドポイント」を参照してください。
設定	REDSHIFT-EVENT-1510	エラー	Amazon Redshift クラスター [クラスター名] へのサンプルデータのロード試行が成功しなかったことを検知しました。サンプルデータをロードするには、まず Amazon S3 バケットにアクセスできるように VPC を設定してから、新しいクラスターを作成してサンプルデータをロードします。 詳細については、「Amazon Redshift 管理ガイド」の「拡張された VPC ルーティングの有効化」を参照してください。
設定	REDSHIFT-EVENT-1511	エラー	アカウントの Elastic IP アドレスの上限を超過したため、Amazon Redshift クラスター [クラスター名] を作成できません。未使用の Elastic IP アドレスを削除するか、Amazon EC2 で上限の引き上げをリクエストしてください。
管理	REDSHIFT-EVENT-2000	INFO	Amazon Redshift クラスター: [cluster name] が作成され、使用可能です。

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
管理	REDSHIFT-EVENT-2001	INFO	Amazon Redshift クラスター [cluster name] が [time] に削除されました。最終スナップショットが保存 [されました/されませんでした]。
管理	REDSHIFT-EVENT-2002	情報	クラスター [クラスター名] の VPC セキュリティグループが [UTC の時刻] に更新されました。
管理	REDSHIFT-EVENT-2003	情報	メンテナンスはクラスター [クラスター名] で [時刻] に開始されました。
管理	REDSHIFT-EVENT-2004	情報	メンテナンスはクラスター [クラスター名] で [UTC の時刻] に完了しました。
管理	REDSHIFT-EVENT-2006	情報	クラスター [クラスター名] のサイズ変更が [UTC の時刻] に開始されました。クラスターは読み取り専用モードです。
管理	REDSHIFT-EVENT-2007	情報	クラスター [クラスター名] のサイズ変更リクエストが確認されました。
管理	REDSHIFT-EVENT-2008	INFO	新しい Amazon Redshift クラスター [cluster name] スナップショット [snapshot name] を作成する復元オペレーションが [time] に開始されました。復元の進行状況をモニタリングするには、AWS マネジメントコンソール にアクセスしてください。
管理	REDSHIFT-EVENT-2013	INFO	Amazon Redshift クラスター [cluster name] の名前が [time] に変更されました。
管理	REDSHIFT-EVENT-2014	INFO	Amazon Redshift クラスター [cluster name] のテーブル復元リクエストが受信されました。

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
管理	REDSHIFT-EVENT-2015	INFO	Amazon Redshift クラスター [cluster name] のテーブル復元が [time] にキャンセルされました。
管理	REDSHIFT-EVENT-2016	INFO	Amazon Redshift クラスター [cluster name] の置換が [time] に開始されました。
管理	REDSHIFT-EVENT-2017	INFO	お客様が開始したメンテナンスは、Amazon Redshift クラスター [クラスター名] で [時間] に開始されました。メンテナンス中、クラスターを使用できない場合があります。
管理	REDSHIFT-EVENT-2018	INFO	お客様が開始したメンテナンスは、Amazon Redshift クラスター [クラスター名] で [時間] に完了しました。
管理	REDSHIFT-EVENT-2019	ERROR	お客様が開始したメンテナンスは、Amazon Redshift クラスター [クラスター名] で [時間] に失敗しました。クラスターを元の状態に戻しています。
管理	REDSHIFT-EVENT-2020	INFO	Amazon Redshift クラスター [クラスター名] のトラックは、[from track (トラック元)] から [to track (トラック先)] に変更する必要があります。
管理	REDSHIFT-EVENT-2021	ERROR	Amazon Redshift クラスター [クラスター名] の [オペレーション] は、当社のキャパシティープールから容量を取得中に成功しませんでした。容量の獲得に取り組んでいますが、今のところ、リクエストをキャンセルしました。このクラスターを削除して、後で再試行してください。

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
管理	REDSHIFT-EVENT-2022	ERROR	Amazon Redshift クラスター [クラスター名] の [オペレーション] は、当社のキャパシティープールから容量を取得中に成功しませんでした。容量の獲得に取り組んでいますが、今のところ、リクエストをキャンセルしました。容量は [代替アベイラビリティーゾーン] で利用できます。このクラスターを削除して、代替アベイラビリティーゾーンで再試行してください。
管理	REDSHIFT-EVENT-2023	ERROR	単一ノードの Amazon Redshift クラスター [クラスター名] でハードウェア障害が検出されました。これにより、クエリの失敗またはクラスターの断続的な高可用性がもたらされる可能性があります。当社のキャパシティープールから容量を取得中にクラスターの交換は成功しませんでした。スナップショットから新しいクラスターを復元する必要があります。このクラスターを削除し、利用可能な最新のスナップショットを選択して、そのスナップショットから新しいクラスターを復元します。これにより、正常なハードウェアに自動的にプロビジョニングされます。

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
管理	REDSHIFT-EVENT-2024	ERROR	単一ノードの Amazon Redshift クラスター [クラスター名] でハードウェア障害が検出されました。これにより、クエリの失敗またはクラスターの断続的な高可用性がもたらされる可能性があります。当社のキャパシティープールから容量を取得中にクラスターの交換は成功しませんでした。容量はアベイラビリティーゾーンで利用できます [代替アベイラビリティーゾーン]。このクラスターを削除し、利用可能な最新のスナップショットを選択して、そのスナップショットから新しいクラスターを復元します。これにより、正常なハードウェアに自動的にプロビジョニングされます。
管理	REDSHIFT-EVENT-3011	INFO	Amazon Redshift クラスター「[cluster name]」の伸縮自在なサイズ変更が [time] に開始されました。サイズ変更中はデータベース接続を保持します。このオペレーション中に、一部のクエリおよび接続が終了やタイムアウトすることがあります。
管理	REDSHIFT-EVENT-3012	INFO	[時刻] に開始されたクラスター「[クラスター名]」の伸縮サイズ変更リクエストを受け取りました。サイズ変更が始まると、イベント通知が送信されます。
Pending	REDSHIFT-EVENT-2025	情報	クラスター [cluster name] のデータベースは [start time] から [end time] の間に更新されます。クラスターはアクセスできません。変化に応じて計画を立てましょう。

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
Pending	REDSHIFT-EVENT-2026	情報	クラスター [cluster name] は [start time] から [end time] の間に更新されます。クラスターはアクセスできません。変化に応じて計画を立てましょう。
モニタリング	REDSHIFT-EVENT-2050	INFO	Amazon Redshift クラスター [cluster name] でハードウェア問題が検出されました。置換リクエストが [time] に開始されました。
モニタリング	REDSHIFT-EVENT-3000	INFO	Amazon Redshift クラスター [cluster name] が [time] に再起動されました。
モニタリング	REDSHIFT-EVENT-3001	INFO	Amazon Redshift クラスターのノード: [cluster name] が [time] に自動的に置き換えられ、クラスターは通常動作中です。
モニタリング	REDSHIFT-EVENT-3002	INFO	Amazon Redshift クラスター [cluster name] のサイズ変更が完了したため、クラスターを読み込みおよび書き込みできます。サイズ変更は [時間] に開始され、完了までに [時間数] かかりました。
モニタリング	REDSHIFT-EVENT-3003	INFO	Amazon Redshift クラスター [cluster name] は、スナップショット [snapshot name] から正常に作成されたため、使用できます。
モニタリング	REDSHIFT-EVENT-3007	INFO	Amazon Redshift スナップショット [スナップショット名] が [ソース AWS リージョン] から [コピー先 AWS リージョン] に、[時間] に正常にコピーされました。
モニタリング	REDSHIFT-EVENT-3008	INFO	Amazon Redshift クラスター [cluster name] のテーブル復元が [time] に開始されました。

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
モニタリング	REDSHIFT-EVENT-3009	INFO	Amazon Redshift クラスター [cluster name] のテーブル復元が [time] に完了しました。
モニタリング	REDSHIFT-EVENT-3010	ERROR	Amazon Redshift クラスター [cluster name] のテーブル復元が [time] に失敗しました。
モニタリング	REDSHIFT-EVENT-3013	ERROR	リクエストされた Amazon Redshift クラスター [cluster name] の伸縮自在なサイズ変更オペレーションが [reason] のために [time] に失敗しました。
モニタリング	REDSHIFT-EVENT-3014	INFO	Amazon Redshift がクラスター [cluster name] を [time] に再起動しました。
保留中	REDSHIFT-EVENT-3015	情報	[start time in UTC] から [end time in UTC] の間に、メンテナンスのため、クラスター [cluster name] の再起動を予定しています。この間、クラスターにはアクセスできません。問題が発生しないように、サービス停止を計画することをお勧めします。
モニタリング	REDSHIFT-EVENT-3016	情報	クラスターは再起動されます。
モニタリング	REDSHIFT-EVENT-3500	ERROR	Amazon Redshift クラスター [cluster name] のサイズ変更に失敗しました。サイズ変更は数分後に自動的に再試行されます。
モニタリング	REDSHIFT-EVENT-3501	ERROR	スナップショット [snapshot name] から Amazon Redshift クラスター [cluster name] を作成する復元オペレーションが [time] に失敗しました。操作を再実行してください。

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
モニタリング	REDSHIFT-EVENT-3504	ERROR	Amazon S3 バケット [bucket name] はクラスター [cluster name] のログ作成には無効です。
モニタリング	REDSHIFT-EVENT-3505	ERROR	Amazon S3 バケット [bucket name] にはクラスター [cluster name] に関する正しい IAM ポリシーがありません。
モニタリング	REDSHIFT-EVENT-3506	ERROR	Amazon S3 バケット [bucket name] が存在しません。クラスター [クラスター名] のログ作成を続行できません。
モニタリング	REDSHIFT-EVENT-3507	ERROR	Amazon Redshift クラスター [cluster name] が EIP [IP address] を使用して作成できません。この EIP は既に使われています。
モニタリング	REDSHIFT-EVENT-3508	ERROR	Amazon Redshift クラスター [cluster name] が EIP [IP address] を使用して作成できません。EIP が見つかりません。
モニタリング	REDSHIFT-EVENT-3509	ERROR	クラスター [クラスター名] のクロスリージョンスナップショットのコピーが有効になっていません。
モニタリング	REDSHIFT-EVENT-3510	ERROR	Amazon Redshift クラスター [cluster name] のテーブル復元が [time] に開始に失敗しました。理由: [理由]。
モニタリング	REDSHIFT-EVENT-3511	ERROR	Amazon Redshift クラスター [cluster name] のテーブル復元が [time] に失敗しました。

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
モニタリング	REDSHIFT-EVENT-3512	ERROR	Amazon Redshift クラスター [cluster name] は、ハードウェアの問題により失敗しました。クラスターは、[時間] に作成された最新スナップショット [スナップショット名] から自動的に復元されています。
モニタリング	REDSHIFT-EVENT-3513	ERROR	Amazon Redshift クラスター [cluster name] は、ハードウェアの問題により失敗しました。クラスターは、[時間] に作成された最新スナップショット [スナップショット名] から自動的に復元されています。この時間後にデータベースに行った変更は、再送信する必要があります。
モニタリング	REDSHIFT-EVENT-3514	ERROR	Amazon Redshift クラスター [cluster name] は、ハードウェアの問題により失敗しました。クラスターは、ハードウェア障害の状態に配置されます。クラスターを削除して、[時間] に作成した最新スナップショット [スナップショット名] から復元してください。
モニタリング	REDSHIFT-EVENT-3515	ERROR	Amazon Redshift クラスター [cluster name] は、ハードウェアの問題により失敗しました。クラスターは、ハードウェア障害の状態に配置されます。クラスターを削除して、[時間] に作成した最新スナップショット [スナップショット名] から復元してください。この時間後にデータベースに行った変更は、再送信する必要があります。

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
モニタリング	REDSHIFT-EVENT-3516	ERROR	Amazon Redshift クラスター [cluster name] は、ハードウェアの問題により失敗し、このクラスターにはバックアップが存在しません。クラスターは、ハードウェア障害のステータスに配置され、削除されます。
モニタリング	REDSHIFT-EVENT-3519	INFO	クラスター [cluster name] は [time] に再起動を開始しました。
モニタリング	REDSHIFT-EVENT-3520	INFO	クラスター [cluster name] は [time] に再起動を完了しました。
モニタリング	REDSHIFT-EVENT-3521	INFO	クラスター「[クラスター名]」に接続の問題が検出されました。[時刻] に自動診断が開始されました。
モニタリング	REDSHIFT-EVENT-3522	INFO	クラスター「[クラスター名]」の復旧アクションが [時刻] に失敗しました。Amazon Redshift チームがソリューションに向けて動いています。
モニタリング	REDSHIFT-EVENT-3533	ERROR	「[クラスター名]」のクラスターのサイズ変更は、[時刻] にキャンセルされました。[理由] のため、オペレーションキャンセルされました。[アクションが必要]。
モニタリング	REDSHIFT-EVENT-3534	INFO	Amazon Redshift クラスター 「[cluster name]」 の伸縮自在なサイズ変更が [time] に完了しました。クラスターは、データの転送中に読み取り/書き込みオペレーションに使用できるようになりました。一部のクエリではデータ転送が完了するまでに時間がかかる場合があります。

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
モニタリング	REDSHIFT-EVENT-3537	INFO	クラスター '[cluster name]' のデータ転送は [time in UTC] に完了しました。
モニタリング	REDSHIFT-EVENT-3600	INFO	Amazon Redshift クラスター [クラスター名] のリクエストされたサイズ変更オペレーションは過去にキャンセルされました。ロールバックは [時刻] に完了しました。
Pending	REDSHIFT-EVENT-3601	情報	クラスター [cluster name] のノードは [start time] から [end time] の間に置き換えられます。このメンテナンスを延期することはできません。変化に応じて計画を立てましょう。
Pending	REDSHIFT-EVENT-3602	情報	クラスター [cluster name] のノードは [start time] から [end time] の間に置き換えられる予定です。クラスターはアクセスできません。変化に応じて計画を立てましょう。
管理	REDSHIFT-EVENT-3603	INFO	内部エラーのため、スナップショット [スナップショット名] からクラスター [クラスター名] を作成する復元オペレーションが失敗しました。クラスターは、互換性のない復元ステータスにあり、削除できます。異なる設定のクラスターでスナップショットの復元を試みてください。
管理	REDSHIFT-EVENT-3614	INFO	スケジュールされたアクション [スケジュールされたアクション名] は [UTC での時刻] に作成されました。最初の呼び出しは [UTC での時刻] にスケジュールされています。
管理	REDSHIFT-EVENT-3615	INFO	スケジュールされたアクション [スケジュールされたアクション名] は [UTC での時刻] にスケジュールされています。

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
モニタリング	REDSHIFT-EVENT-3616	INFO	[UTC での時刻] にスケジュールされたアクション [スケジュールされたアクション名] が「SUCCEEDED」ステータスで終了しました。
モニタリング	REDSHIFT-EVENT-3617	ERROR	スケジュールされたアクション [スケジュールされたアクション名] は、遅延のため [UTC での時刻] にスキップされました。
モニタリング	REDSHIFT-EVENT-3618	INFO	クラスター [クラスター名] の一時停止オペレーションが [UTC 時間] に開始しました。一時停止開始
モニタリング	REDSHIFT-EVENT-3619	情報	Amazon Redshift クラスター [cluster name] が [UTC time] に正常に一時停止されました。
管理	REDSHIFT-EVENT-3626	INFO	スケジュールされたアクション [スケジュールされたアクション名] が [UTC での時刻] に変更されました。最初の呼び出しは [UTC での時刻] にスケジュールされています。
管理	REDSHIFT-EVENT-3627	INFO	スケジュールされたアクション [スケジュールされたアクション名] が [UTC での時刻] に削除されました。
モニタリング	REDSHIFT-EVENT-3628	ERROR	[UTC での時刻] にスケジュールされたアクション [スケジュールされたアクション名] が「FAILED」ステータスで終了しました。
モニタリング	REDSHIFT-EVENT-3629	情報	クラスター [cluster name] の内部フェイルオーバーリクエストが開始されました。

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
モニタリング	REDSHIFT-EVENT-3630	情報	Amazon Redshift クラスターは、復旧のために Amazon Redshift クラスター [cluster name] を [availability-zone] から [availability-zone] に正常に再配置しました。
管理	REDSHIFT-EVENT-3631	情報	Amazon Redshift [クラスター名] は、お客様の再配置リクエストを受け取りました。アベイラビリティーゾーンの再配置が完了すると、Amazon Redshift はイベント通知を送信します。
モニタリング	REDSHIFT-EVENT-3632	情報	Amazon Redshift クラスター [cluster name] は、[availability-zone] から [availability-zone] に正常に再配置されました。これで、クラスターを使用できます。
モニタリング	REDSHIFT-EVENT-3658	エラー	Redshift クラスター [cluster id] の EC2-Classic から EC2-VPC への移行に失敗しました。
モニタリング	REDSHIFT-EVENT-3659	情報	Redshift クラスター [cluster id] の EC2-Classic から EC2-VPC への移行が成功しました。
モニタリング	REDSHIFT-EVENT-3660	情報	クラスターは、ハードウェア障害の状態に配置されます。EC2-Classic クラスターを削除した上で、[time in UTC] に作成された最新のスナップショット [snapshot name] から EC2-VPC クラスターに復元してください。
管理	REDSHIFT-EVENT-3666	情報	Amazon Redshift マルチ AZ クラスター [クラスター名] は [UTC 時間] に障害を検出し、自動復旧をトリガーしました。

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
管理	REDSHIFT-EVENT-3667	情報	Amazon Redshift マルチ AZ クラスター [クラスター名] は [UTC 時間] に正常に復旧し、[1 番目のアベイラビリティーゾーン] で使用できます。別の AZ でのセカンダリコンピューティングは間もなく利用可能になります。
モニタリング	REDSHIFT-EVENT-3668	エラー	Amazon Redshift マルチ AZ クラスター [クラスター名] は [UTC 時間] に復旧に失敗しました。
管理	REDSHIFT-EVENT-3669	情報	Amazon Redshift マルチ AZ クラスター [クラスター名] は [UTC 時間] に正常に復旧し、[1 番目のアベイラビリティーゾーン] および [2 番目のアベイラビリティーゾーン] のコンピューティングリソースで使用できます。
管理	REDSHIFT-EVENT-3670	情報	Amazon Redshift クラスター [クラスター名] のメンテナンスは [UTC 時間] に完了し、[1 番目のアベイラビリティーゾーン] のコンピューティングリソースで使用できます。別の AZ でのセカンダリコンピューティングは間もなく利用可能になります。
管理	REDSHIFT-EVENT-3671	情報	Amazon Redshift クラスター [クラスター名] のサイズ変更は [UTC 時間] に完了し、[1 番目のアベイラビリティーゾーン] で使用できます。別の AZ でのセカンダリコンピューティングは間もなく利用可能になります。
管理	REDSHIFT-EVENT-3672	情報	Amazon Redshift マルチ AZ クラスター [クラスター名] は [UTC 時間] に [2 番目のアベイラビリティーゾーン] で障害を検出し、自動復旧をトリガーしました。

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
管理	REDSHIFT-EVENT-3673	情報	Amazon Redshift クラスター [クラスター名] のマルチ AZ を有効にするオペレーションが [UTC 時間] に開始されました。
管理	REDSHIFT-EVENT-3674	情報	Amazon Redshift クラスター [クラスター名] のマルチ AZ を有効にするオペレーションが [UTC 時間] に正常に完了しました。
モニタリング	REDSHIFT-EVENT-3675	エラー	Amazon Redshift クラスター [クラスター名] のマルチ AZ を有効にするオペレーションが [UTC 時間] に失敗しました。
管理	REDSHIFT-EVENT-3676	情報	Amazon Redshift クラスター [クラスター名] のマルチ AZ を無効にするオペレーションが [UTC 時間] に開始されました。
管理	REDSHIFT-EVENT-3677	情報	Amazon Redshift クラスター [クラスター名] のマルチ AZ を無効にするオペレーションが [UTC 時間] に正常に完了しました。
モニタリング	REDSHIFT-EVENT-3678	エラー	Amazon Redshift クラスター [クラスター名] のマルチ AZ を無効にするオペレーションが [UTC 時間] に失敗しました。
設定	REDSHIFT-EVENT-3679	情報	Amazon Redshift クラスター [クラスター名] のポートは正常に変更されました。
設定	REDSHIFT-EVENT-3680	エラー	このオペレーションに必要なサービスリンクロール (SLR) にアクセスできないため、Amazon Redshift はクラスター [クラスター名] を作成できませんでした。Amazon Redshift コンソールから作成を試みてください。Amazon Redshift は SLR を自動的に作成します。

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
モニタリング	REDSHIFT-EVENT-3684	エラー	Amazon S3 バケット [バケット名] は、不明またはアクセスできない AWS KMS キーで暗号化されています。Amazon S3 バケットの暗号化を変更します。
管理	REDSHIFT-EVENT-3685	エラー	十分なディスク容量が利用できないため、クラスター [クラスター名] での復元オペレーションが失敗しました。オペレーションはロールバックされます。別の設定のクラスターで復元を試してください。
管理	REDSHIFT-EVENT-3686	エラー	十分なディスク容量が利用できないため、クラスター [クラスター名] でのサイズ変更オペレーションが失敗しました。オペレーションはロールバックされます。別の設定のクラスターでサイズ変更を試してください。
管理	REDSHIFT-EVENT-3687	情報	要求された Amazon Redshift クラスター [クラスター名] のサイズ変更オペレーションは、[時間 (UTC)] に完了しました。
セキュリティ	REDSHIFT-EVENT-3688	エラー	Amazon Redshift クラスター [クラスター名] の暗号化キーのローテーションを完了できませんでした。
データ共有	REDSHIFT-EVENT-3689	情報	Amazon Redshift 名前空間 <クラスター名> を Glue Data Catalog アカウント <アカウント ID> に登録するオペレーションが、<UTC 時刻> に開始しました。
データ共有	REDSHIFT-EVENT-3690	情報	Amazon Redshift 名前空間 <クラスター名> を Glue Data Catalog アカウント <アカウント ID> に登録するオペレーションが、<UTC 時刻> に正常に完了しました。

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
データ共有	REDSHIFT-EVENT-3691	情報	Amazon Redshift 名前空間 <クラスター名> を Glue Data Catalog アカウント <アカウント ID> に登録するオペレーションが、<UTC 時刻> に失敗しました。
データ共有	REDSHIFT-EVENT-3692	情報	Amazon Redshift 名前空間 <クラスター名> を Glue Data Catalog アカウント <アカウント ID> から登録解除するオペレーションが、<UTC 時刻> に開始しました。
データ共有	REDSHIFT-EVENT-3693	情報	Amazon Redshift 名前空間 <クラスター名> を Glue Data Catalog アカウント <アカウント ID> から登録解除するオペレーションが、<UTC 時刻> に正常に完了しました。
データ共有	REDSHIFT-EVENT-3694	情報	Amazon Redshift 名前空間 <クラスター名> を Glue Data Catalog アカウント <アカウント ID> から登録解除するオペレーションが、<UTC 時刻> に失敗しました。
セキュリティ	REDSHIFT-EVENT-4000	情報	Amazon Redshift クラスター: [cluster name] の管理者認証情報が [time] に更新されました。
セキュリティ	REDSHIFT-EVENT-4001	INFO	セキュリティグループ [セキュリティグループ名] が [時間] に変更されました。変更は、関連付けられたすべてのクラスターに対して自動的に行われます。
セキュリティ	REDSHIFT-EVENT-4005	エラー	クラスター [クラスター名] に無効な AWS KMS key アクセス許可があります。クラスターは「INACCESSIBLE_KMS_KEY」状態になり、アクセスできなくなります。

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
セキュリティ	REDSHIFT-EVENT-4006	情報	AWS KMS key アクセス許可が復元されたため、クラスター [クラスター名] は以前の状態に復元されました。
セキュリティ	REDSHIFT-EVENT-4500	エラー	Amazon Redshift クラスター [クラスター名] に対して指定した 1 つ以上のセキュリティグループが見つかりませんでした。有効なセキュリティグループがある場合は、それらがクラスターに適用されます。無効なセキュリティグループは適用されません。既存のクラスターを変更していた場合、有効なセキュリティグループがなければ、クラスターは元のセキュリティグループを維持します。新しいクラスターを作成していた場合、有効なセキュリティグループがなければ、クラスターの作成は失敗します。いずれの場合も、有効なセキュリティグループを使用してリクエストを再試行してください。Amazon Redshift でのセキュリティグループの管理の詳細については、「Amazon Redshift 管理ガイド」の「VPC セキュリティグループ」を参照してください。
セキュリティ	REDSHIFT-EVENT-4501	ERROR	クラスターセキュリティグループ [クラスターセキュリティグループ名] で指定されたセキュリティグループ [セキュリティグループ名] が見つかりませんでした。認可を完了できません。

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
セキュリティ	REDSHIFT-EVENT-4502	エラー	同時に実行されるアクティビティがあったため、Amazon Redshift クラスター [cluster name] の管理者認証情報が、[time] に更新できませんでした。現在のワークロードがアクティブなワークロードを完了または減らしてから、オペレーションを再試行します。
セキュリティ	REDSHIFT-EVENT-4503	エラー	Amazon Redshift はクラスター [クラスター名] のシークレットにアクセスできません。
セキュリティ	REDSHIFT-EVENT-4504	エラー	Amazon Redshift は、クラスター [クラスター名] の管理者認証情報シークレットの暗号化に使用された KMS キー [KMS キー] にアクセスできません。
セキュリティ	REDSHIFT-EVENT-4505	エラー	Amazon Redshift はクラスター [クラスター名] のシークレットをローテーションできません。これは、クラスター上でオペレーションが進行中であるためです。
セキュリティ	REDSHIFT-EVENT-4506	エラー	Amazon Redshift クラスター [クラスター名] は、一時停止中です。Amazon Redshift は、一時停止中のクラスターのシークレットをローテーションできません。

パラメータグループソースタイプのカテゴリとイベント

次の表は、パラメータグループがソースタイプである場合のイベントカテゴリとイベントのリストを示します。

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
設定	REDSHIFT-EVENT-1002	INFO	パラメータ [パラメータ名] が [時間] に [value] から [value] に更新されました。
設定	REDSHIFT-EVENT-1003	INFO	クラスターパラメータグループ [グループ名] が作成されました。
設定	REDSHIFT-EVENT-1004	INFO	クラスターパラメータグループ [グループ名] が削除されました。
設定	REDSHIFT-EVENT-1005	INFO	クラスターパラメータグループ [名前] が [時間] に更新されました。動的パラメータのみを変更した場合は、関連付けられているクラスターが現在変更中です。静的パラメータを変更した場合、動的パラメータを含む、すべての更新は、関連するクラスターを再起動した場合に適用されます。

セキュリティグループソースタイプのカテゴリとイベント

次の表は、セキュリティグループがソースタイプである場合のイベントカテゴリとイベントのリストを示します。

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
セキュリティ	REDSHIFT-EVENT-4002	INFO	クラスターセキュリティグループ [グループ名] が作成されました。
セキュリティ	REDSHIFT-EVENT-4003	INFO	クラスターセキュリティグループ [グループ名] が削除されました。

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
セキュリティ	REDSHIFT-EVENT-4004	INFO	クラスターセキュリティグループ [グループ名] が [時間] に変更されました。変更は、関連するすべてのクラスターに自動的に適用されます。

スナップショットソースタイプのカテゴリとイベント

次の表は、スナップショットがソースタイプである場合のイベントカテゴリとイベントのリストを示します。

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
管理	REDSHIFT-EVENT-2009	INFO	Amazon Redshift クラスター [cluster name] のユーザースナップショット [snapshot name] が [time] に開始されました。スナップショットの進行状況をモニタリングするには、AWS マネジメントコンソール にアクセスしてください。
管理	REDSHIFT-EVENT-2010	INFO	Amazon Redshift クラスター [cluster name] のユーザースナップショット [snapshot name] が [time] にキャンセルされました。
管理	REDSHIFT-EVENT-2011	INFO	Amazon Redshift クラスター [cluster name] のユーザースナップショット [snapshot name] が [time] に削除されました。

Amazon Redshift のカテゴリ	イベント ID	イベントの重大度	説明
管理	REDSHIFT-EVENT-2012	INFO	Amazon Redshift クラスター [cluster name] の最終スナップショット [snapshot name] が [time] に開始されました。
モニタリング	REDSHIFT-EVENT-3004	INFO	Amazon Redshift クラスター [cluster name] のユーザースナップショット [snapshot name] が [time] に正常に完了しました。
モニタリング	REDSHIFT-EVENT-3005	INFO	Amazon Redshift クラスター [name] の最終スナップショット [name] が [time] に正常に完了しました。
モニタリング	REDSHIFT-EVENT-3006	INFO	Amazon Redshift クラスター [cluster name] の最終スナップショット [snapshot name] が [time] にキャンセルされました。
モニタリング	REDSHIFT-EVENT-3502	ERROR	Amazon Redshift クラスター [cluster name] の最終スナップショット [snapshot name] が [time] に失敗しました。チームは問題を調査しています。AWS マネジメントコンソール にアクセスしてオペレーションを再試行してください。
モニタリング	REDSHIFT-EVENT-3503	ERROR	Amazon Redshift クラスター [cluster name] のユーザースナップショット [snapshot name] が [time] に失敗しました。チームは問題を調査しています。AWS マネジメントコンソール にアクセスしてオペレーションを再試行してください。

Amazon EventBridge による Amazon Redshift Serverless イベント通知

Amazon Redshift Serverless は Amazon EventBridge を使用してイベント通知を管理し、データウェアハウスの変更に関する最新情報を伝えます。Amazon EventBridge は、アプリケーションをさまざまなソースからのデータに接続するために使用できるサーバーレスのイベントバスサービスです。この場合、イベントソースは Amazon Redshift です。環境内の監視された変更であるイベントは、Amazon Redshift データウェアハウスから EventBridge に自動的に送信されます。イベントは、ほぼリアルタイムに配信されます。

EventBridge には、イベントルールを記述するための環境を提供するの機能が含まれ、特定のイベントに対して実行するアクションを指定できます。ターゲットを設定することもできます。ターゲットは、EventBridge がイベントを送信できるリソースです。ターゲットには、API 送信先、Amazon CloudWatch ロググループなどを含めることができます。ルールの詳細については、「Amazon EventBridge ルール」を参照してください。ターゲットの詳細については、「Amazon EventBridge ターゲット」を参照してください。

イベントは、重大度とカテゴリに分類できます。以下のフィルタが利用可能です。

- リソースフィルター - イベントが関連付けられているリソースに基づいてメッセージを受信します。リソースには、ワークグループ、スナップショットなどが含まれます。
- 時間枠フィルター - 特定の期間のイベントのスコープを設定します。
- カテゴリフィルタリング - 指定したカテゴリ内のすべてのイベントについてイベント通知を受け取ることができます。

次の表には、Amazon Redshift Serverless イベントと、追加のメタデータが含まれています。

Amazon Redshift のカテゴリ	外部イベント ID	イベントの重大度	メッセージ説明
RateChange	REDSHIFT-SERVERLESS-EVENT-1001	情報	ワークグループベース RPU の変更が <time in UTC> に正常に完了しました。

Amazon Redshift のカテゴリ	外部イベント ID	イベントの重大度	メッセージ説明
RateChange	REDSHIFT-SERVERLESS-EVENT-1002	エラー	ワークグループベース RPU の変更が <time in UTC> に完了に失敗しました。
モニタリング	REDSHIFT-SERVERLESS-EVENT-1003	情報	ソフトウェアは、<time in UTC> に Amazon Redshift データウェアハウス <endpoint name> で更新されました。
設定	REDSHIFT-SERVERLESS-EVENT-1011	エラー	このオペレーションに必要なサービスリンクロール (SLR) にアクセスできないため、Amazon Redshift Serverless はワークグループ [ワークグループ名] を作成できませんでした。Amazon Redshift コンソールで作成を試みてください。Amazon Redshift は SLR を自動的に作成します。

Amazon Redshift のカテゴリ	外部イベント ID	イベントの重大度	メッセージ説明
モニタリング	REDSHIFT-SERVERLESS-EVENT-1029	エラー	ワークグループのベース RPU の変更は、使用可能なディスク容量が不足しているため [UTC の時刻] に完了できませんでした。別の設定でもう一度試してください。
モニタリング	REDSHIFT-SERVERLESS-EVENT-1500	エラー	アカウントの Elastic IP アドレスの上限を超過したため、ワークグループ <workgroup name> を作成または更新できません。未使用の Elastic IP アドレスを削除するか、Amazon EC2 で上限の引き上げをリクエストしてください。

Amazon Redshift のカテゴリ	外部イベント ID	イベントの重大度	メッセージ説明
モニタリング	REDSHIFT-SERVERLESS-EVENT-1501	エラー	サブネット <subnet id> には使用可能な IP アドレスがありません。これにより、次のクエリタイプがワークグループ <workgroup name> で正常に実行されなくなります: EMR、フェデレーションクエリ、Amazon EC2 からの COPY/UNLOAD。この問題を解決するには、ENI を削除してサブネットの IP を解放します。

Amazon Redshift のカテゴリ	外部イベント ID	イベントの重大度	メッセージ説明
モニタリング	REDSHIFT-SERVERLESS-EVENT-1502	エラー	サブネット <subnet id> には使用可能な IP アドレスがありません。これにより、Amazon EMR、Redshift フェデレーテッドクエリ、Redshift COPY/UNLOAD、Redshift ML クエリタイプをワークグループ <workgroup name> で正常に実行されなくなります。この問題を解決するには、Elastic Network Interface (ENI) を削除してサブネットの IP を解放します。
管理	REDSHIFT-SERVERLESS-EVENT-1008	情報	Amazon Redshift ワークグループ <workgroup name> が作成され、使用可能です。
管理	REDSHIFT-SERVERLESS-EVENT-1009	情報	Amazon Redshift ワークグループ <workgroup name> は <time in UTC> に削除されました。

Amazon Redshift のカテゴリ	外部イベント ID	イベントの重大度	メッセージ説明
モニタリング	REDSHIFT-SERVERLESS-EVENT-1000	情報	スナップショット <snapshot name> は <time in UTC> に正常に完了しました。
管理	REDSHIFT-SERVERLESS-EVENT-1004	情報	名前空間 <namespace name> のスナップショットからの復元が <time in UTC> に正常に完了しました。
管理	REDSHIFT-SERVERLESS-EVENT-1005	エラー	名前空間 <namespace name> のスナップショットからの復元が <time in UTC> に失敗しました。
管理	REDSHIFT-SERVERLESS-EVENT-1006	情報	名前空間 <namespace name> のリカバリポイントからの復元が <time in UTC> に正常に完了しました。
管理	REDSHIFT-SERVERLESS-EVENT-1007	情報	名前空間 <namespace name> のリカバリポイントからの復元が <time in UTC> に失敗しました。

Amazon Redshift のカテゴリ	外部イベント ID	イベントの重大度	メッセージ説明
セキュリティ	REDSHIFT-SERVERLESS-EVENT-1012	エラー	Amazon Redshift は、名前空間 <namespace name> のシークレットにアクセスできません。
セキュリティ	REDSHIFT-SERVERLESS-EVENT-1013	エラー	Amazon Redshift は、名前空間 <namespace name> の管理者認証情報のシークレットの暗号化に使用された KMS キーにアクセスできません。
セキュリティ	REDSHIFT-SERVERLESS-EVENT-1014	エラー	ワークグループでオペレーションが進行中であるため、Amazon Redshift は名前空間 <namespace name> のシークレットをローテーションできません。

Amazon Redshift のカテゴリ	外部イベント ID	イベントの重大度	メッセージ説明
セキュリティ	REDSHIFT-SERVERLESS-EVENT-1015	エラー	この名前空間 <namespace name> にはワークグループが接続されていません。Amazon Redshift は、ワークグループがアタッチされている名前空間のシークレットのみをローテーションできます。
セキュリティ	REDSHIFT-SERVERLESS-EVENT-1016	情報	名前空間 <namespace name> の管理者認証情報が <time in UTC> に失敗しました。
セキュリティ	REDSHIFT-SERVERLESS-EVENT-1030	情報	Amazon Redshift 名前空間 <名前空間名> を Glue Data Catalog アカウント <アカウント ID> に登録するオペレーションが、<UTC 時刻> に開始しました。

Amazon Redshift のカテゴリ	外部イベント ID	イベントの重大度	メッセージ説明
セキュリティ	REDSHIFT-SERVERLESS-EVENT-1031	情報	Amazon Redshift 名前空間 <名前空間名> を Glue Data Catalog アカウント <アカウント ID> に登録するオペレーションが、<UTC 時刻> に正常に完了しました。
セキュリティ	REDSHIFT-SERVERLESS-EVENT-1032	情報	Amazon Redshift 名前空間 <名前空間名> を Glue Data Catalog アカウント <アカウント ID> に登録するオペレーションが、<UTC 時刻> に失敗しました。
セキュリティ	REDSHIFT-SERVERLESS-EVENT-1033	情報	Amazon Redshift 名前空間 <名前空間名> を Glue Data Catalog アカウント <アカウント ID> から登録解除するオペレーションが、<UTC 時刻> に開始しました。

Amazon Redshift のカテゴリ	外部イベント ID	イベントの重大度	メッセージ説明
セキュリティ	REDSHIFT-SERVERLESS-EVENT-1034	情報	Amazon Redshift 名前空間 <名前空間名> を Glue Data Catalog アカウント <アカウント ID> から登録解除するオペレーションが、<UTC 時刻> に正常に完了しました。
セキュリティ	REDSHIFT-SERVERLESS-EVENT-1035	情報	Amazon Redshift 名前空間 <名前空間名> を Glue Data Catalog アカウント <アカウント ID> から登録解除するオペレーションが、<UTC 時刻> に失敗しました。
管理	REDSHIFT-SERVERLESS-EVENT-1036	エラー	Redshift Serverless ワークグループ <ワークグループ名> で、お客様が開始したトラックの更新に失敗しました。ワークグループを元のトラックに戻します。

Amazon Redshift のカテゴリ	外部イベント ID	イベントの重大度	メッセージ説明
管理	REDSHIFT-SERVERLESS-EVENT-1037	エラー	Redshift Serverless ワークグループ <ワークグループ名> で、トラックの更新に失敗しました。その時点ではワークグループがビジー状態だったため、Amazon Redshift Serverless ワークグループのトラックの更新に失敗しました。その結果、ワークグループは元のトラックに戻されました。ワークグループのアクティビティが少ない時間待ってから、トラックを再度更新してください。
管理	REDSHIFT-SERVERLESS-EVENT-1038	情報	Amazon Redshift ワークグループ <ワークグループ名> のトラックが変更されました。トラックの変更が完了しました。

Amazon Redshift のカテゴリ	外部イベント ID	イベントの重大度	メッセージ説明
名前空間	REDSHIFT-SERVERLESS-EVENT-1039	エラー	名前空間 <名前空間名> には無効な AWS KMS key アクセス許可があります。名前空間は「INACCESSIBLE_KMS_KEY」状態になり、アクセスできなくなります。
名前空間	REDSHIFT-SERVERLESS-EVENT-1040	情報	AWS KMS key アクセス許可が復元されたため、名前空間 <名前空間名> が以前の状態に復元されました。
名前空間	REDSHIFT-SERVERLESS-EVENT-1045	情報	Redshift Serverless 名前空間 <namespace name> のデータ共有 <datashare name> は、サーバーレス復元のため削除されました。
名前空間	REDSHIFT-SERVERLESS-EVENT-1046	情報	Redshift Serverless 名前空間 <namespace name> のデータ共有 <datashare name> のコンシューマー <consumer principle id> は、サーバーレス復元のため取り消されました。

Amazon Redshift のカテゴリ	外部イベント ID	イベントの重大度	メッセージ説明
名前空間	REDSHIFT-SERVERLESS-EVENT-1047	情報	サーバーレス復元のため、Redshift Serverless 名前空間 <namespace name> のデータ共有 <datashare name> のパブリックアクセシビリティが変更されました。
セキュリティ	REDSHIFT-SERVERLESS-EVENT-1048	エラー	内部の問題により、名前空間 <名前空間名> の Redshift-Secrets Manager 統合をオプトアウトするリクエストが失敗しました。この問題を解決するには、オプトアウト操作を手動で再試行します。

Amazon Redshift のカテゴリ	外部イベント ID	イベントの重大度	メッセージ説明
セキュリティ	REDSHIFT-SERVERLESS-EVENT-1049	エラー	内部の問題により、名前空間 <名前空間名> の Redshift-Secrets Manager 統合へのオプトインリクエストが正常に完了できませんでした。Redshift Serverless 名前空間に関連付けられたシークレットを手動でローテーションして、この問題を解決します。
セキュリティ	REDSHIFT-SERVERLESS-EVENT-1050	エラー	内部の問題により、名前空間 <名前空間名> で管理者認証情報をリセットするリクエストが失敗しました。操作を再試行して、管理者ユーザーの認証情報をリセットしてください。

Amazon EventBridge によるイベント通知のゼロ ETL 統合

ゼロ ETL 統合では、Amazon EventBridge を使用してイベント通知を管理し、統合の変更に関する最新情報を伝えます。Amazon EventBridge は、アプリケーションをさまざまなソースからのデータに接続するために使用できるサーバーレスのイベントバスサービスです。この場合、イベントソースは Amazon Redshift です。環境内の監視された変更であるイベントは、Amazon Redshift データ

ウェアハウスから EventBridge に自動的に送信されます。イベントは、ほぼリアルタイムに配信されます。

EventBridge は、イベントルールを記述するための環境を用意します。ルールでは、特定のイベントに対して実行するアクションを指定できます。ターゲットを設定することもできます。ターゲットは、EventBridge がイベントを送信できるリソースです。ターゲットには、API 送信先、Amazon CloudWatch ロググループなどを含めることができます。ルールの詳細については、「Amazon EventBridge ルール」を参照してください。ターゲットの詳細については、「Amazon EventBridge ターゲット」を参照してください。

イベントは、重大度とカテゴリに分類できます。以下のフィルタが利用可能です。

- リソースフィルター - イベントが関連付けられているリソースに基づいてメッセージを受信します。リソースには、ワークグループ、スナップショットなどが含まれます。
- 時間枠フィルター - 特定の期間のイベントのスコープを設定します。
- カテゴリフィルタリング - 指定したカテゴリ内のすべてのイベントについてイベント通知を受け取ることができます。

次の表には、ゼロ ETL 統合イベントと、追加のメタデータをまとめています。

Amazon Redshift のカテゴリ	外部イベント ID	イベントの重大度	メッセージ説明
モニタリング	REDSHIFT-INTEGRATION-EVENT-0000	情報	ゼロ ETL 統合 <統合名> が作成され、<UTC 時刻> にアクティブになりました。
モニタリング	REDSHIFT-INTEGRATION-EVENT-0001	情報	ゼロ ETL 統合 <統合名> が <UTC 時刻> に削除されました。
モニタリング	REDSHIFT-INTEGRATION-EVENT-0002	情報	ゼロ ETL 統合 <統合名> の削除を <UTC

Amazon Redshift のカテゴリ	外部イベント ID	イベントの重大度	メッセージ説明
			時刻> に開始しました。
モニタリング	REDSHIFT-INTEGRATION-EVENT-0003	情報	ゼロ ETL 統合 <統合名> が、トランザクションデータを Amazon Redshift データウェアハウスに同期しています。
モニタリング	REDSHIFT-INTEGRATION-EVENT-0004	WARNING	テーブルにプライマリキーがないため、Amazon Redshift はテーブルをレプリケートできません。ソーステーブルにプライマリキー (複数可) を追加すると、テーブルは自動的に再同期されます。

Amazon Redshift のカテゴリ	外部イベント ID	イベントの重大度	メッセージ説明
モニタリング	REDSHIFT-INTEGRATION-EVENT-0005	WARNING	1 つ以上の列でサポートされていないデータ型が使用されているため、Amazon Redshift はテーブルをレプリケートできません。フィルターでこのテーブルを除外するように統合を変更するか、ソーステーブルから列を削除し、「ALTER DATABASE <Redshift データベース名> INTEGRATION REFRESH TABLE <スキーマ名>.<テーブル名>」を実行してこのテーブルを同期します。
モニタリング	REDSHIFT-INTEGRATION-EVENT-0006	エラー	統合を作成できません。統合を削除して再作成してください。それでもエラーが解決しない場合は、AWS サポートまでお問い合わせください。

Amazon Redshift のカテゴリ	外部イベント ID	イベントの重大度	メッセージ説明
モニタリング	REDSHIFT-INTEGRATION-EVENT-0007	エラー	内部エラーが発生したため、データを読み込めません。統合を削除して再作成してください。それでもエラーが解決しない場合は、AWS サポートまでお問い合わせください。
モニタリング	REDSHIFT-INTEGRATION-EVENT-0008	エラー	ソース DB クラスターのアクセス許可が取り消されたため、認可に失敗しました。カスタマーマネージドキー (CMK) を使用して統合を暗号化している場合は、KMS キーのアクセス許可を確認します。統合を削除して再作成してください。
モニタリング	REDSHIFT-INTEGRATION-EVENT-0009	エラー	テーブルとスキーマの数が Amazon Redshift の制限を超えているため、Amazon Redshift にデータを送信できません。統合を削除して再作成してください。

Amazon Redshift のカテゴリ	外部イベント ID	イベントの重大度	メッセージ説明
モニタリング	REDSHIFT-INTEGRATION-EVENT-0012	エラー	復元ポイントからの復元が、復元先のサーバーレス名前空間で呼び出されました。統合を削除して再作成してください。
モニタリング	REDSHIFT-INTEGRATION-EVENT-0013	情報	ゼロ ETL 統合 <統合名> がアクティブになりました。
モニタリング	REDSHIFT-INTEGRATION-EVENT-0014	エラー	内部エラーにより変更できないため、統合 <統合名> は失敗状態です。統合を削除して再作成してください。それでもエラーが解決しない場合は、AWS サポートまでお問い合わせください。
Operation	REDSHIFT-INTEGRATION-EVENT-0016	情報	ゼロ ETL 統合 <統合名> が変更リクエストを処理しています。
Operation	REDSHIFT-INTEGRATION-EVENT-0017	情報	ゼロ ETL 統合 <統合名> への変更が適用されました。

Amazon Redshift のカテゴリ	外部イベント ID	イベントの重大度	メッセージ説明
Operation	REDSHIFT-INTEGRATION-EVENT-0018	WARNING	ターゲットの Amazon Redshift クラスターが一時停止中です。クラスターが一時停止するまで待ってから、再開してデータのストリーミングを継続してください。
Operation	REDSHIFT-INTEGRATION-EVENT-0019	WARNING	ターゲットの Amazon Redshift クラスターが一時停止しました。データのストリーミングを続行するには、クラスターを再開する必要があります。
Operation	REDSHIFT-INTEGRATION-EVENT-0020	WARNING	ターゲットの Amazon Redshift クラスターが再開中です。クラスターがアクティブになるまで待ってから、データのストリーミングを継続してください。

Amazon Redshift のカテゴリ	外部イベント ID	イベントの重大度	メッセージ説明
モニタリング	REDSHIFT-INTEGRATION-EVENT-0025	エラー	サーバーレス復元がターゲットのサーバーレス名前空間で呼び出され、復元されたスナップショットで使用できないため、統合は非アクティブ化されます。統合が利用可能になったときに作成されたスナップショットを復元するか、統合を削除します。
設定	REDSHIFT-INTEGRATION-EVENT-1000	エラー	ソース Aurora DB クラスターの 1 つまたは複数のパラメーターが誤って設定されています。パラメータグループを修正し、クラスターを再起動して変更を適用し、統合を再作成してください。

Amazon Redshift のカテゴリ	外部イベント ID	イベントの重大度	メッセージ説明
設定	REDSHIFT-INTEGRATION-EVENT-1001	エラー	enable_case_sensitive_identifier パラメーターの値が間違っているため、統合できません。ソース Aurora DB クラスターの値を true に設定し、統合を削除して再作成してください。
設定	REDSHIFT-INTEGRATION-EVENT-1002	エラー	cdc_insert_enabled パラメーターの値が正しくないため、統合できませんでした。ソース Aurora DB クラスターの値を true に設定し、統合を削除して再作成してください。
設定	REDSHIFT-INTEGRATION-EVENT-1003	エラー	ソース DB クラスターパラメータグループの binlog_format パラメータは ROW に設定する必要があります。パラメータグループを修正し、クラスターを再起動して変更を適用し、統合を再作成してください。

Amazon Redshift のカテゴリ	外部イベント ID	イベントの重大度	メッセージ説明
設定	REDSHIFT-INTEGRATION-EVENT-1004	エラー	binlog_transaction_compression クラスターパラメーターが有効になっているため、データを読み込めません。パラメーター値を OFF に設定し、ライターインスタンスを再起動して変更を適用し、統合を再作成してください。
設定	REDSHIFT-INTEGRATION-EVENT-1005	エラー	binlog_row_value_options クラスターパラメーターが PARTIAL_JSON に設定されているため、データを読み込めません。この設定はサポートされていません。パラメータグループを修正し、ライターインスタンスを再起動して変更を適用し、統合を再作成してください。
設定	REDSHIFT-INTEGRATION-EVENT-1006	WARNING	統合フィルターを解析できません。フィルター構文を修正してください。

Amazon Redshift でのクォータと制限

Amazon Redshift には、AWS リージョンごとに AWS アカウント内の複数のリソースの使用を制限するクォータがあります。各クォータにはデフォルト値があり、一部のクォータは調整可能です。

Amazon Redshift オブジェクトのクォータ

Amazon Redshift には、複数のオブジェクトタイプの使用を制限するクォータがあります。それぞれにデフォルト値があります。

クォータ名	AWS デフォルト値	引き上げ可能	説明
AWSスナップショットごとにスナップショットの復元を許可できる アカウント	20	いいえ	スナップショットごとにスナップショットの復元を許可できる AWS アカウントの最大数。
AWS ごとにスナップショットの復元を許可できる アカウント AWS KMS key	100	いいえ	KMS キーごとにスナップショットの復元を許可できる AWS アカウントの最大数。つまり、1 つの KMS キーで暗号化された 10 個のスナップショットがある場合、10 個の AWS アカウントに各スナップショットを復元することを許可できます。または、最大 100 個のアカウントのその他の組み合わせや、スナップショットごとに 20 個のアカウントを超えないその他の組み合わせを許可できます。
他の AWS サービスにアクセスするための Amazon	50[1]	いいえ	クラスターと IAM ロールを所有するユーザーの他の AWS サービスへのアクセスを Amazon Redshift に許可するために、クラスターに関連付けることができる IAM ロールの最大数。

クォータ名	AWS デフォルト値	引き上げ可能	説明
Redshift のクラスター IAM ロール			[1]次の AWS リージョン では、クォータは 10 です。us-iso-east-1、us-iso-west-1、us-isob-east-1。
すべてのユーザー定義の手動 WLM キューの同時実行レベル (クエリスロット)	50	いいえ	手動ワークロード管理によって定義されるすべてのユーザー定義キューの最大クエリスロット。
同時実行スケーリングクラスター	10	[Yes (はい)]	同時実行スケーリングクラスターの最大数。
クラスター内の DC2 ノード	128	はい	クラスターに割り当てることができる DC2 ノードの最大数。各ノードタイプのノード制限の詳細については、「Amazon Redshift のクラスターとノード」を参照してください。
イベントサブスクリプション	20	はい	現在の AWS リージョンにおけるこのアカウントのイベントサブスクリプションの最大数。
ノード	200	はい	現在の AWS リージョンにある、このアカウントのすべてのデータベースインスタンスにおけるノードの最大数。
パラメータグループ	20	いいえ	現在の AWS リージョンにおけるこのアカウントのパラメータグループの最大数。

クォータ名	AWS デフォルト値	引き上げ可能	説明
クラスター内の RG ノード	128	はい	クラスターに割り当てることができる RG ノードの最大数。各ノードタイプのノード制限の詳細については、「Amazon Redshift のクラスターとノード」を参照してください。
クラスター内の RA3 ノード	128	はい	クラスターに割り当てることができる RA3 ノードの最大数。各ノードタイプのノード制限の詳細については、「Amazon Redshift のクラスターとノード」を参照してください。
クラスターに接続された RedShift マネージド VPC エンドポイント	30	はい	クラスターに接続できる RedShift マネージド VPC エンドポイントの最大数。RedShift マネージド VPC エンドポイントの詳細については、「RedShift が管理する VPC エンドポイント」を参照してください。
RedShift マネージド VPC エンドポイントを介してアクセスされるクラスターに対する被付与者	10	[Yes (はい)]	クラスターの所有者がクラスターの RedShift マネージド VPC エンドポイントを作成するよう許可できる被付与者の最大数。RedShift マネージド VPC エンドポイントの詳細については、「RedShift が管理する VPC エンドポイント」を参照してください。
認可ごとの RedShift マネージド VPC エンドポイント	10	[Yes (はい)]	認可ごとに作成できる RedShift マネージド VPC エンドポイントの最大数。RedShift マネージド VPC エンドポイントの詳細については、「RedShift が管理する VPC エンドポイント」を参照してください。
リザーブドノード	200	はい	現在の AWS リージョンにおけるこのアカウントのリザーブドノードの最大数。

クォータ名	AWS デフォルト値	引き上げ可能	説明
クラスターごとの各データベースのスキーマ	9,900	いいえ	クラスターごとに各データベースで作成できるスキーマの最大数。ただし、pg_temp_* スキーマはこのクォータにはカウントされません。
セキュリティグループ	20	はい	現在の AWS リージョンにおけるこのアカウントのセキュリティグループの最大数。
COPY によりロードするときの単一行のサイズ	64	いいえ	COPY コマンドを使用してロードするときの単一行の最大サイズ (MB 単位)。これは、COPY コマンドがスキャンできる単一の未解析レコードの最大サイズです。行内の各 SUPER オブジェクトはそれぞれ 16 MB に制限されます。
スナップショット	700	はい	現在の AWS リージョンにおけるこのアカウントのユーザースナップショットの最大数。
サブネットグループ	50	はい	現在の AWS リージョンにおけるこのアカウントのサブネットグループの最大数。
サブネットグループ内のサブネット	20	はい	サブネットグループのサブネットの最大数。

クォータ名	AWS デフォルト値	引き上げ可能	説明
単一ノードクラスターを持つ rg.large クラスターノードタイプのテーブル	9,900	いいえ	単一ノードクラスターを持つ rg.large クラスターノードタイプのテーブルの最大数。この制限には、永続テーブル、一時テーブル、データシェアテーブル、マテリアライズドビューが含まれています。外部テーブルは、一時テーブルとしてカウントされます。一時テーブルには、ユーザー定義の一時テーブルや、クエリの処理またはシステムメンテナンス中に Amazon Redshift によって作成された一時テーブルが含まれます。ビューおよびシステムテーブルはこの制限には含まれません。
マルチノードクラスターを持つ rg.large クラスターノードタイプのテーブル	20,000	いいえ	マルチノードクラスターを持つ rg.large ノードタイプのテーブルの最大数。この制限には、永続テーブル、一時テーブル、データシェアテーブル、マテリアライズドビューが含まれています。外部テーブルは、一時テーブルとしてカウントされます。一時テーブルには、ユーザー定義の一時テーブルや、クエリの処理またはシステムメンテナンス中に Amazon Redshift によって作成された一時テーブルが含まれます。ビューおよびシステムテーブルはこの制限には含まれません。
rg.xlarge クラスターノードタイプのテーブル。	20,000	いいえ	rg.xlarge クラスターノードタイプのテーブルの最大数。この制限には、永続テーブル、一時テーブル、データシェアテーブル、マテリアライズドビューが含まれています。外部テーブルは、一時テーブルとしてカウントされます。一時テーブルには、ユーザー定義の一時テーブルや、クエリの処理またはシステムメンテナンス中に Amazon Redshift によって作成された一時テーブルが含まれます。ビューおよびシステムテーブルはこの制限には含まれません。

クォータ名	AWS デフォルト値	引き上げ可能	説明
rg.4xlarge クラスターノードタイプのテーブル	200,000 件の	いいえ	rg.4xlarge クラスターノードタイプのテーブルの最大数。この制限には、永続テーブル、一時テーブル、データシェアテーブル、マテリアライズドビューが含まれています。外部テーブルは、一時テーブルとしてカウントされます。一時テーブルには、ユーザー定義の一時テーブルや、クエリの処理またはシステムメンテナンス中に Amazon Redshift によって作成された一時テーブルが含まれます。ビューおよびシステムテーブルはこの制限には含まれません。
rg.12xlarge クラスターノードタイプのテーブル	200,000 件の	いいえ	rg.12xlarge クラスターノードタイプのテーブルの最大数。この制限には、永続テーブル、一時テーブル、データシェアテーブル、マテリアライズドビューが含まれています。外部テーブルは、一時テーブルとしてカウントされます。一時テーブルには、ユーザー定義の一時テーブルや、クエリの処理またはシステムメンテナンス中に Amazon Redshift によって作成された一時テーブルが含まれます。ビューおよびシステムテーブルはこの制限には含まれません。
ra3.large クラスターノードタイプのテーブル	9,900	いいえ	ra3.large クラスターノードタイプのテーブルの最大数。この制限には、永続テーブル、一時テーブル、データシェアテーブル、マテリアライズドビューが含まれています。外部テーブルは、一時テーブルとしてカウントされます。一時テーブルには、ユーザー定義の一時テーブルや、クエリの処理またはシステムメンテナンス中に Amazon Redshift によって作成された一時テーブルが含まれます。ビューおよびシステムテーブルはこの制限には含まれません。

クォータ名	AWS デフォルト値	引き上げ可能	説明
単一ノードクラスターを持つ ra3.xlplus クラスターノードタイプのテーブル。	9,900	いいえ	単一ノードクラスターを持つ ra3.xlplus クラスターノードタイプのテーブルの最大数。この制限には、永続テーブル、一時テーブル、データシェアテーブル、マテリアライズドビューが含まれています。外部テーブルは、一時テーブルとしてカウントされます。一時テーブルには、ユーザー定義の一時テーブルや、クエリの処理またはシステムメンテナンス中に Amazon Redshift によって作成された一時テーブルが含まれます。ビューおよびシステムテーブルはこの制限には含まれません。
マルチノードクラスターを持つ ra3.xlplus クラスターノードタイプのテーブル。	20,000	いいえ	マルチノードクラスターを持つ ra3.xlplus クラスターノードタイプのテーブルの最大数。この制限には、永続テーブル、一時テーブル、データシェアテーブル、マテリアライズドビューが含まれています。外部テーブルは、一時テーブルとしてカウントされます。一時テーブルには、ユーザー定義の一時テーブルや、クエリの処理またはシステムメンテナンス中に Amazon Redshift によって作成された一時テーブルが含まれます。ビューおよびシステムテーブルはこの制限には含まれません。
ra3.4xlarge クラスターノードタイプのテーブル	200,000 件の	いいえ	ra3.4xlarge クラスターノードタイプのテーブルの最大数。この制限には、永続テーブル、一時テーブル、データシェアテーブル、マテリアライズドビューが含まれています。外部テーブルは、一時テーブルとしてカウントされます。一時テーブルには、ユーザー定義の一時テーブルや、クエリの処理またはシステムメンテナンス中に Amazon Redshift によって作成された一時テーブルが含まれます。ビューおよびシステムテーブルはこの制限には含まれません。

クォータ名	AWS デフォルト値	引き上げ可能	説明
ra3.16xlarge クラスターノードタイプのテーブル	200,000 件の	いいえ	ra3.16xlarge クラスターノードタイプのテーブルの最大数。この制限には、永続テーブル、一時テーブル、データシェアテーブル、マテリアライズドビューが含まれています。外部テーブルは、一時テーブルとしてカウントされます。一時テーブルには、ユーザー定義の一時テーブルや、クエリの処理またはシステムメンテナンス中に Amazon Redshift によって作成された一時テーブルが含まれます。ビューおよびシステムテーブルはこの制限には含まれません。
dc2.large クラスターノードタイプのテーブル	9,900	いいえ	dc2.large クラスターノードタイプのテーブルの最大数。この制限には、永続テーブル、一時テーブル、データシェアテーブル、マテリアライズドビューが含まれています。外部テーブルは、一時テーブルとしてカウントされます。一時テーブルには、ユーザー定義の一時テーブルや、クエリの処理またはシステムメンテナンス中に Amazon Redshift によって作成された一時テーブルが含まれます。ビューおよびシステムテーブルはこの制限には含まれません。
dc2.8xlarge クラスターノードタイプのテーブル	200,000 件の	いいえ	dc2.8xlarge クラスターノードタイプのテーブルの最大数。この制限には、永続テーブル、一時テーブル、データシェアテーブル、マテリアライズドビューが含まれています。外部テーブルは、一時テーブルとしてカウントされます。一時テーブルには、ユーザー定義の一時テーブルや、クエリの処理またはシステムメンテナンス中に Amazon Redshift によって作成された一時テーブルが含まれます。ビューおよびシステムテーブルはこの制限には含まれません。
データベースの数	60	いいえ	Amazon Redshift クラスターで許可されるデータベースの最大数です。これにはデータ共有から作成されたデータベースは含まれません。

クォータ名	AWS デフォルト値	引き上げ可能	説明
アイドル状態または非アクティブなセッションのタイムアウト	4 時間	いいえ	この設定は、クラスターにのみ適用されます。ユーザーのアイドルセッションのタイムアウト値の設定については、Amazon Redshift データベースデベロッパーガイドの「ALTER USER」を参照してください。ユーザー設定は、クラスター設定よりも優先されます。
アイドル状態のトランザクションのタイムアウト	6 時間	いいえ	Amazon Redshift がトランザクションに関連付けられたセッションを終了するまでの、オープントランザクションのアイドル状態の最大期間。この設定は、ユーザー定義のアイドル状態のタイムアウト設定よりも優先されます。クラスターに適用されます。
データベースのストアドプロシージャ	10,000	いいえ	ストアドプロシージャの最大数です。制限の詳細については「ストアドプロシージャサポートの制限と違い」を参照してください。
RG ノードの最大接続数	2,000	いいえ	RG クラスターへの最大接続数。これはすべての RG ノードタイプに適用されるハード制限であり、引き上げることはできません。
RA3 ノードの最大接続数	2,000	いいえ	RA3 クラスターへの最大接続数。これはすべての RA3 ノードタイプに適用されるハード制限であり、引き上げることはできません。
DS2 ノードの最大接続数	可変	いいえ	ds2.large クラスターへの最大接続数は 500 です。dc2.8xlarge クラスターへの最大接続数は 2000 です。
クラスター内の Amazon Redshift ロールの数	1,000	はい	クラスターごとに作成できる Amazon Redshift ロールの最大数。ロールベースのアクセスコントロール (RBAC) ロールの詳細については、「Amazon Redshift データベース開発者ガイド」の「ロールベースのアクセスコントロール (RBAC)」を参照してください。

Amazon Redshift Serverless オブジェクトのクォータ

Amazon Redshift には、Amazon Redshift Serverless 名前空間で複数のオブジェクトタイプの使用を制限するクォータがあります。それぞれにデフォルト値があります。

クォータ名	AWS デフォルト値	引き上げ可能	説明
データベースの数	100	いいえ	Amazon Redshift Serverless 名前空間で許可されるデータベースの最大数です。これにはデータ共有から作成されたデータベースは含まれません。
スキーマの数	9,900	いいえ	Amazon Redshift Serverless 名前空間で許可されるスキーマの最大数です。
テーブルの数	200,000 件の	いいえ	Amazon Redshift Serverless 名前空間で許可されるテーブルの最大数です。
4 つの Redshift 処理ユニット (RPU) を使用するデータウェアハウス用の Redshift マネージドストレージユニット	最大 32 TB	いいえ	4 RPU のデータウェアハウスでサポートされる最大 RMS ストレージ。32 TB の制限内で、Amazon Redshift は数百 MB のストレージを内部で使用します。
8、16、または 24 個の Redshift 処理ユニット (RPU) を使用するデータウェアハウス用	最大 128 TB	いいえ	8、16、または 24 個の RPU を使用するデータウェアハウスでサポートされる最大 RMS ストレージ。128 TB の制限内で、Amazon Redshift は数百 MB のストレージを内部で使用します。

クォータ名	AWS デフォルト値	引き上げ可能	説明
の Redshift マネージドストレージユニット			
アイドル状態または非アクティブなセッションのタイムアウト	1 時間	いいえ	ユーザーのアイドルセッションのタイムアウト値の設定については、Amazon Redshift データベースデベロッパーガイドの「ALTER USER」を参照してください。ユーザー設定が優先されます。
実行中のクエリのタイムアウト	86,399 秒 (24 時間)	いいえ	Amazon Redshift がクエリを終了するまでの実行中のクエリの最大時間。
アイドル状態のトランザクションのタイムアウト	6 時間	いいえ	Amazon Redshift Serverless がトランザクションに関連付けられたセッションを終了するまでの、オープントランザクションのアイドル状態の最大期間。この設定は、ユーザー定義のアイドル状態のタイムアウト設定よりも優先されます。
最大接続数	2000	いいえ	ワークグループに接続できる接続の最大数。
ワークグループの数	25	はい	サポートされるワークグループの数。
名前空間の数	25	はい	サポートされる名前空間の数。
ワークグループ内の Amazon Redshift ロールの数	1,000	はい	ワークグループごとに作成できる Amazon Redshift ロールの最大数。ロールベースのアクセスコントロール (RBAC) ロールの詳細については、「Amazon Redshift データベース開発者ガイド」の「ロールベースのアクセスコントロール (RBAC)」を参照してください。

クォータ名	AWS デフォルト値	引き上げ可能	説明
AWS アカウント ごとのベース Redshift 処理ユニット (RPU)	3,200 RPU または過去 6 か月の最大集計ベース RPU の 1.5 倍のいずれか大きい方。	はい	Redshift Serverless のベース Redshift 処理ユニット (RPU) の AWS アカウント クォータは、3,200 RPU または過去 6 か月の 最大集計ベース RPU の 1.5 倍のいずれか大きい方になります。 .

Amazon Redshift Serverless 請求がタイムアウト設定によってどのように影響を受けるかについては、「Amazon Redshift Serverless での請求」を参照してください。

Amazon Redshift Data API のクォータ

Amazon Redshift には Redshift Data API の使用を制限するクォータがあります。それぞれにデフォルト値があります。Amazon Redshift Data API の詳細については、「Amazon Redshift Data API の使用」を参照してください。

クォータ名	AWS デフォルト値	引き上げ可能	説明
`BatchExecuteStatement` API のトランザクション/秒 (TPS)	20	いいえ	スロットリングなしで 1 秒あたりに実行できるオペレーションリクエストの最大数。
`CancelStatement` API のトランザク	3	いいえ	スロットリングなしで 1 秒あたりに実行できるオペレーションリクエストの最大数。

クォータ名	AWS デフォルト値	引き上げ可能	説明
ション/秒 (TPS)			
DescribeStatement API のトランザクション/秒 (TPS)	100	いいえ	スロットリングなしで 1 秒あたりに実行できるオペレーションリクエストの最大数。
DescribeTable API のトランザクション/秒 (TPS)	3	いいえ	スロットリングなしで 1 秒あたりに実行できるオペレーションリクエストの最大数。
ExecuteStatement API のトランザクション/秒 (TPS)	30	いいえ	スロットリングなしで 1 秒あたりに実行できるオペレーションリクエストの最大数。
GetStatementResult API のトランザクション/秒 (TPS)	20	いいえ	スロットリングなしで 1 秒あたりに実行できるオペレーションリクエストの最大数。
ListDatabases API のトランザクション/秒 (TPS)	3	いいえ	スロットリングなしで 1 秒あたりに実行できるオペレーションリクエストの最大数。

クォータ名	AWS デフォルト値	引き上げ可能	説明
ListSchemas API のトランザクション/秒 (TPS)	3	いいえ	スロットリングなしで 1 秒あたりに実行できるオペレーションリクエストの最大数。
ListStatements API のトランザクション/秒 (TPS)	3	いいえ	スロットリングなしで 1 秒あたりに実行できるオペレーションリクエストの最大数。
ListTables API のトランザクション/秒 (TPS)	3	いいえ	スロットリングなしで 1 秒あたりに実行できるオペレーションリクエストの最大数。

クエリエディタ v2 オブジェクトのクォータ

Amazon Redshift には、Amazon Redshift クエリエディタ v2 で複数のオブジェクトタイプの使用を制限するクォータがあります。それぞれにデフォルト値があります。

クォータ名	AWS デフォルト値	引き上げ可能	説明
Connections	500	はい	現在のリージョンにおいて、このアカウントでクエリエディタ v2 を使用して作成できる最大の接続数。
アカウントあたりのアクティブな	50	はい	現在のリージョンにおいて、このアカウントでクエリエディタ v2 を使用できる同時プリンシパルの最大数。

クォータ名	AWS デフォルト値	引き上げ可能	説明
プリンシパル数			
保存されたクエリ	2,500	はい	現在のリージョンで、このアカウントでクエリエディタ v2 を使用して作成できる保存済みクエリの最大数。
クエリのバージョン	20	はい	現在のリージョンでこのアカウントのクエリエディタ v2 を使用して作成できる、クエリごとの最大バージョン数。
保存済みのチャート	500	はい	現在のリージョンで、このアカウントのクエリエディタ v2 を使用して作成できる保存済みのチャートの最大数。
クエリごとにフェッチされるデータのサイズ	100	いいえ	現在のリージョンのこのアカウントのクエリエディタ v2 によってクエリごとにフェッチされるデータの最大サイズ (MB 単位) 。
100 行ごとにフェッチされるデータのサイズ	5	いいえ	このアカウントにおいて現在のリージョンで、クエリエディタ v2 によってクエリの結果ページ (100 行) ごとにフェッチされるデータの最大サイズ (MB 単位)。
同時接続の最大数	3	いいえ	ユーザーあたりの最大データベース接続数 (分離セッションを含む)。この値は、クエリエディタ v2 管理者が [Account settings] (アカウント設定) で 1 ~ 10 の範囲で設定することができます。管理者が設定した制限に達した場合は、SQL の実行時に分離セッションではなく共有セッションを使用することを検討してください。接続の詳細については、「クエリエディタ v2 を開く」を参照してください。制限の設定に関する詳細については、「アカウント設定」を参照してください。

Amazon Redshift Spectrum のクォータと制限

Amazon Redshift Spectrum には、次のクォータと制限があります。

- AWS Glue Data Catalog を使用する場合の AWS アカウントあたりのデータベースの最大数。この値については、Amazon Web Services 全般のリファレンス の「AWS Glue Service Quotas」を参照してください。
- AWS Glue Data Catalog を使用する場合のデータベースあたりのテーブルの最大数。この値については、Amazon Web Services 全般のリファレンス の「AWS Glue Service Quotas」を参照してください。
- AWS Glue Data Catalog を使用する場合のテーブルあたりのパーティションの最大数。この値については、Amazon Web Services 全般のリファレンス の「AWS Glue Service Quotas」を参照してください。
- AWS Glue Data Catalog を使用する場合の AWS アカウントあたりのパーティションの最大数。この値については、Amazon Web Services 全般のリファレンス の「AWS Glue Service Quotas」を参照してください。
- AWS Glue Data Catalog を使用する場合の外部テーブルの列の最大数。疑似列が有効な場合は 1,597、疑似列が有効でないときは 1,600 です。
- AWS Glue Data Catalog を使用する場合の ION ファイルまたは JSON ファイルの文字列値の最大サイズは 16 KB です。この制限に達すると、文字列は切り捨てられます。
- 単一の ALTER TABLE ステートメントを使用して、最大 100 のパーティションを追加することができます。
- すべての S3 データは、Amazon Redshift クラスターと同じ AWS リージョンにある必要があります。
- ION および JSON 形式のタイムスタンプには、ISO8601 形式を使用する必要があります。
- ORC ファイルの外部圧縮はサポートされていません。
- テキスト、OpenCSV、Regex SERDE では、'\177' より大きい 8 進数の区切り記号はサポートされていません。
- すべてのパーティションからの読み取りを回避するには、パーティション列に述語を指定する必要があります。

  たとえば、次の術語は `ship_dtm` 列でフィルタリングしますが、パーティション列 `ship_yyyymm` にはフィルタリングを適用しません。

  ```
  WHERE ship_dtm > '2018-04-01'.
  ```

不要なパーティションをスキップするには、術後 `WHERE ship_yyyymm = '201804'` を追加する必要があります。この述語はパーティション `\ship_yyyymm=201804\` への読み込みオペレーションを制限します。

これらの制限は Apache Hive メタストアに適用されません。

命名に関する制約

以下の表に、Amazon Redshift での命名に関する制約を示します。

クラスター識別子	• クラスター識別子には小文字のみ使用できます。 • 1 ~ 63 文字の英数字またはハイフンを使用する必要があります。 • 1 字目は文字である必要があります。 • ハイフンを、文字列の最後に使用したり、2 つ続けて使用したりすることはできません。 • AWS アカウントのすべてのクラスター間で一意である必要があります。
データベース名	• データベース名には、1 ~ 64 文字の英数字を使用する必要があります。 • 小文字のみを使用する必要があります。 • 予約語は使用できません。予約語の一覧については、Amazon Redshift データベースデベロッパーガイドの予約語を参照してください。
RedShift マネージド VPC エンドポイントのエンドポイント名	• エンドポイント名には 1 ~ 30 文字を使用する必要があります。 •

	有効な文字は、A ~ Z、a ~ z、0 ~ 9、- (ハイフン) です。 • 1 字目は文字である必要があります。 • 名前にハイフンを 2 つ続けて使用したり、文字列の最後にハイフンを使用したりすることはできません。
管理者ユーザー名	• 管理者ユーザー名には、小文字のみを使用できます。 • 1 ~ 128 個の英数字を使用する必要があります。 • 1 字目は文字である必要があります。 • 予約語は使用できません。予約語の一覧については、Amazon Redshift データベースデベロッパーガイドの予約語を参照してください。
Admin パスワード	• 管理者パスワードには、8 ~ 64 文字を使用する必要があります。 • 最低 1 個の大文字を含める必要があります。 • 最低 1 個の小文字を含める必要があります。 • 数字を 1 個含める必要があります。 • ASCII 文字 (ASCII コード 33 ~ 126) のうち、' (一重引用符)、" (二重引用符)、\、/、@ を除く任意の文字を使用できます。

パラメータグループ名	• パラメータグループ名には 1 ~ 255 文字の英数字またはハイフンを使用する必要があります。 • 小文字のみを使用する必要があります。 • 1 字目は文字である必要があります。 • 文字列の最後にハイフンを使用したり、ハイフンを 2 つ続けて使用したりすることはできません。
クラスターセキュリティグループ名	• クラスターセキュリティグループ名には 255 文字以下の英数字またはハイフンを使用する必要があります。 • 小文字のみを使用する必要があります。 • 「**Default**」という文字列を使用することはできません。 • AWS アカウントで作成したすべてのセキュリティグループ間で一意である必要があります。
サブネットグループ名	• サブネットグループ名には 255 文字以下の英数字またはハイフンを使用する必要があります。 • 小文字のみを使用する必要があります。 • 「**Default**」という文字列を使用することはできません。 • AWS アカウントで作成したすべてのサブネットグループ間で一意である必要があります。

クラススナップショット識別子	• クラススナップショット識別子には 255 文字以下の英数字またはハイフンを使用する必要があります。 • 小文字のみを使用する必要があります。 • 「**Default**」という文字列を使用することはできません。 • AWS アカウントで作成したすべてのスナップショット識別子間で一意である必要があります。

Amazon Redshift のリソースのタグ付け

AWS では、タグはキーと値のペアで構成されるユーザー定義のラベルです。Amazon Redshift では、一目で分かるリソースに関するメタデータを提供し、コスト配分に基づいて請求レポートを分類するタグ付けがサポートされます。コスト配分にタグを使用するには、まず AWS Billing and Cost Management サービスでタグをアクティブ化する必要があります。請求目的でタグを設定および使用する方法の詳細については、「コスト配分タグを使用したカスタム請求レポート」と「月別コスト割り当てレポートの設定」を参照してください。

タグは、Amazon Redshift のリソースに必須ではありませんが、コンテキストの提供に役立ちます。コストセンター、プロジェクト名、リソースについての他の関連情報に関するメタデータを持つリソースにタグ付けできます。たとえば、テスト環境に属するリソースと本稼働環境に属するリソースを追跡するとします。`environment` というキーを作成し、値 `test` または `production` を指定して、各環境で使用されるリソースを特定できます。他の AWS サービスでタグ付けを使用する場合や、ビジネスの標準カテゴリがある場合、一貫性を保つために Amazon Redshift のリソースには同じキーと値のペアを作成することをお勧めします。

タグは、クラスターのサイズを変更した後や、同じリージョン内のクラスターのスナップショットを復元した後も保持されます。一方、別のリージョンにスナップショットをコピーした場合はタグが保持されないため、新しいリージョンでタグを再作成する必要があります。リソースを削除した場合、関連付けられたタグが削除されます。

各リソースには、リソースに割り当てられた 1 つ以上のタグの集合であるタグセットが 1 つあります。リソースごとに、タグセットあたり最大 50 個のタグを含めることができます。タグは、リソースの作成時とリソースの作成後に追加できます。Amazon Redshift では、以下のリソースタイプにタグを追加できます。

- CIDR/IP
- クラスター
- クラスターセキュリティグループ
- クラスターのセキュリティグループの進入ルール
- Amazon EC2 セキュリティグループ
- ハードウェアセキュリティモジュール (HSM) 接続
- HSM クライアント証明書
- パラメータグループ

- スナップショット
- サブネットグループ
- 統合 (ゼロ ETL 統合)

Amazon Redshift コンソールからタグ付けを使用する場合、ユーザーは AWS マネージドポリシー `AmazonRedshiftFullAccess` をアタッチできます。Amazon Redshift コンソールユーザーにアタッチできる、タグ付け権限が制限された IAM ポリシーの例については、例 7: ユーザーが Amazon Redshift コンソールを使用してリソースにタグを付けることを許可する を参照してください。タグ付けの詳細については、「What is AWS Resource Groups?」を参照してください。

タグ付け要件

タグには、次の要件があります。

- キーにプレフィックス `aws:` を付けることはできません。
- キーはタグセットごとに一意であることが必要です。
- キーに使用できる文字数は 1 ~ 128 文字です。
- 値に使用できる文字数は 0 ~ 256 文字です。
- 値は、タグセットごとに一意にする必要はありません。
- キーと値に使用できる文字は、Unicode 文字、数字、空白、および _ . : / = + - @ の記号です。
- キーと値は大文字と小文字が区別されます。

リソースのタグの管理

次の手順では、リソースのタグを使用する方法について説明します。

Amazon Redshift リソースでタグを管理するには

1. AWS マネジメントコンソール にサインインして、https://console.aws.amazon.com/redshiftv2/ で Amazon Redshift コンソールを開きます。
2. ナビゲーションメニューで [Configurations] (設定)、[Manage tags] (タグの管理) の順に選択します。
3. リソースの選択肢を入力して、追加、変更、または削除するタグを選択します。次に、[Manage tags of the resources that you chose (選択したリソースのタグを管理する)] を選択します。

タグ付けできるリソースには、クラスター、パラメータグループ、サブネットグループ、HSM クライアント証明書、HSM 接続、スナップショットがあります。

4. [Manage tags] (タグの管理) ナビゲーションページで、[Review and apply tag changes] (タグの変更をレビューして適用する)、[Apply] (適用) の順に選択して変更を保存します。

AWS Backup と Amazon Redshift の統合

AWS Backup はフルマネージド型のサービスであり、AWS のサービス、クラウド内、オンプレミス間におけるデータ保護の一元化と自動化に役立ちます。

AWS Backup を Amazon Redshift に使用すると、データ保護ポリシーを設定し、Amazon Redshift リソースに対するアクティビティを 1 か所でモニタリングすることができます。Amazon Redshift のプロビジョニングされたクラスターとサーバーレス名前空間にスナップショットを作成して保存することもできます。これにより、以前は個別に行わなければならなかったバックアップタスクが、手動プロセスなしで自動化および統合できます。

> **Note**
>
> RG または RA3 でプロビジョニングされたクラスターと Amazon Redshift Serverless ワークグループでは、バックアップ用ではないテーブルはサポートされていません。RG または RA3 クラスターおよびサーバーレスワークグループでバックアップしないとマークされたテーブルは、スナップショットの作成中に常にバックアップされ、スナップショットから復元するときに復元される永続テーブルとして扱われます。バックアップ用ではないテーブルのスナップショットコストを回避するには、スナップショットを作成する前にそれらを切り捨てます。

バックアップまたは復旧ポイントは、指定した時刻における Amazon Redshift クラスターなどのリソースのコンテンツを表します。AWS Backup はバックアップボールトにバックアップを保存します。バックアップボールトは、ビジネスニーズに合わせて構成できます。復旧ポイントという用語とバックアップという用語は同じ意味で使用されます。AWS Backup の詳細については、「AWS Backup Developer Guide」の「Backup creation, maintenance, and restore」を参照してください。

Amazon Redshift は AWS Backup とネイティブに統合されています。これにより、バックアッププランを定義し、Amazon Redshift リソースをバックアッププランに割り当てることができます。AWS Backup は Amazon Redshift 手動スナップショットの作成を自動化して、バックアッププランで指定したバックアップボールトに安全に保存します。ボールトの詳細については、「AWS Backup Developer Guide」の「Backup vaults」を参照してください。バックアッププランでは、バックアップ頻度、バックアップウィンドウ、ライフサイクル、またはバックアップボールトを定義できます。バックアッププランの詳細については、「AWS Backup Developer Guide」の「Backup plans」を参照してください。

AWS Backup を使用せずに Amazon Redshift Serverless スナップショットを作成および復元する方法については、「スナップショットと復旧ポイント」を参照してください。AWS Backup を使用せずに Amazon Redshift でプロビジョニングされたクラスタースナップショットを作成および復元する方法については、「Amazon Redshift スナップショットとバックアップ」を参照してください。

トピック

- AWS Backup をAmazon Redshift で使用する場合の考慮事項
- AWS Backup を Amazon Redshift で使用する場合の制限
- Amazon Redshift による AWS Backup の管理

AWS Backup をAmazon Redshift で使用する場合の考慮事項

Amazon Redshift で AWS Backup を使用する場合、以下の点を考慮します。

- Amazon Redshift 向け AWS Backup は、AWS Backup と Amazon Redshift の両方が同じ AWS リージョン で使用可能な場合に使用できます。AWS Backup が利用可能なリージョンについては、「AWS 全般のリファレンス」の「Amazon Redshift endpoints and quotas」を参照してください。
- AWS Backup の使用を開始するには、すべての前提条件を満たしていることを確認します。詳細については、「AWS Backup Developer Guide」の「Prerequisites」を参照してください。
- AWS Backup サービスに積極的にオプトインしてください。オプトインの選択肢は特定のアカウントと AWS リージョン に適用されます。特定のアカウントにおいて複数のリージョンでバックアップを使用する場合は、そのアカウントで個々のリージョンにオプトインする必要があります。詳細については、「AWS Backup Developer Guide」の「Opt in to managing services with AWS Backup」を参照してください。
- Amazon Redshift での AWS Backup 統合は、プロビジョニングされたクラスターとサーバーレス名前空間の手動スナップショットのみをサポートします。
- AWS Backup を使用してスナップショット設定を管理すると、Amazon Redshift を使用して手動スナップショット設定を管理し続けることはできません。代わりに、AWS Backup プランを使用して引き続き設定を管理できます。詳細については、「AWS Backup Developer Guide」の「Backup plans」を参照してください。
- データウェアハウススナップショット全体をサーバーレス名前空間に復元することは、破壊的な変更です。データウェアハウススナップショットをターゲット名前空間に復元すると、その名前空間内の既存のデータはすべて失われます。これは、データウェアハウススナップショットの復元にの

み該当します。単一のテーブルスナップショットを名前空間に復元しても、既存のデータは削除されません。

- スナップショットをプロビジョニングされたクラスターに復元するには、RestoreFromClusterSnapshot アクセス許可を持つ IAM ポリシーが必要です。スナップショットをサーバーレス名前空間に復元するには、RestoreFromSnapshot アクセス許可を持つ IAM ポリシーが必要です。これらのアクセス許可は、ソーススナップショットタイプではなく、ターゲットデータウェアハウスタイプに適用されます。例えば、クラススナップショットを名前空間に復元するには、RestoreFromClusterSnapshot ではなく、RestoreFromSnapshot アクセス許可が必要です。IAM ポリシーの管理の詳細については、「Amazon Redshift での Identity and Access Management」を参照してください。

AWS Backup を Amazon Redshift で使用する場合の制限

Amazon Redshift で AWS Backup を使用する場合の制限事項を以下に示します。

- Amazon Redshift の自動スナップショットの管理には AWS Backup は使用できません。自動スナップショットを管理するには、タグを使用します。リソースのタグ付けの詳細については、「Amazon Redshift でのリソースのタグ付け」を参照してください。
- スナップショットから単一のテーブルを復元する場合、プロビジョニングされたクラススナップショットからサーバーレス名前空間への復元、またはその逆の復元はできません。スナップショット全体は任意の設定で復元できます。例えば、プロビジョニングされたクラススナップショット内のすべてのデータベースをサーバーレス名前空間に復元することはできますが、同じスナップショット内の単一のテーブルを同じ名前空間に復元することはできません。

Amazon Redshift による AWS Backup の管理

Amazon Redshift データウェアハウスのリソースを保護するには、AWS Backup コンソールを使用するか、プログラムで AWS Backup API または AWS Command Line Interface (AWS CLI) を使用することができます。リソースを復元する必要がある場合は、AWS Backup コンソールまたは AWS CLI を使用して、必要なリソースを検索して復元できます。詳細については、 を参照してください。AWS Command Line Interface

AWS Backup を Amazon Redshift に使用すると、以下のアクションを実行できます。

- Amazon Redshift スナップショットを自動的に開始する定期的なバックアップを作成します。定期的なバックアップは、長期的なデータ保持のニーズを満たすのに役立ちます。詳細については、「AWS Backup Developer Guide」の「Amazon Redshift backups」を参照してください。
- バックアッププランを一元的に設定して、バックアップのスケジュールと保存を自動化します。
- プロビジョニングされたクラスターまたはサーバーレス名前空間を、選択した保存済みバックアップに復元します。スナップショット内のすべてのデータを復元するか、スナップショットから 1 つのテーブルを復元するかを選択できます。リソースをバックアップする頻度を設定します。プロビジョニングされたクラススナップショットの復元については、「AWS Backup Developer Guide」の「Restore an Amazon Redshift cluster」を参照してください。サーバーレス名前空間スナップショットの復元については、「AWS Backup Developer Guideの「Amazon Redshift Serverless restore」を参照してください。

Amazon Redshift のクラスターバージョン

Amazon Redshift はクラスターバージョンを定期的にリリースしています。Amazon Redshift クラスターは、システムメンテナンスウィンドウ中にパッチが適用されます。パッチのタイミングは、AWS リージョン やメンテナンスウィンドウの設定によって異なります。Amazon Redshift コンソールでメンテナンスウィンドウの設定を表示または変更できます。メンテナンスの詳細については、「クラスターのメンテナンス」を参照してください。

クラスターのバージョンは、Amazon Redshift コンソールのクラスターの詳細のメンテナンスタブで確認できます。または、SQL コマンドの出力でクラスターのバージョンを確認できます。

```
SELECT version();
```

> **Note**
>
> Amazon Redshift の動作に影響する重要な更新は、Amazon Redshift の進化に伴って導入されます。これらの変更に対応し、適宜対策を講じて、ワークロードが中断する可能性を回避するには、「Amazon Redshift の動作の変更」を参照してください。

トピック

- Amazon Redshift パッチ 205
- Amazon Redshift パッチ 204
- Amazon Redshift パッチ 203
- Amazon Redshift パッチ 202
- Amazon Redshift パッチ 201
- Amazon Redshift パッチ 200
- Amazon Redshift パッチ 199
- Amazon Redshift パッチ 198
- Amazon Redshift パッチ 197
- Amazon Redshift パッチ 196
- Amazon Redshift パッチ 195
- Amazon Redshift パッチ 194
- Amazon Redshift パッチ 193

- Amazon Redshift パッチ 192
- Amazon Redshift パッチ 191
- Amazon Redshift パッチ 190
- Amazon Redshift パッチ 189
- Amazon Redshift パッチ 188
- Amazon Redshift パッチ 187
- Amazon Redshift パッチ 186
- Amazon Redshift パッチ 185
- Amazon Redshift パッチ 184
- Amazon Redshift パッチ 183
- Amazon Redshift パッチ 182
- Amazon Redshift パッチ 181
- Amazon Redshift パッチ 180
- Amazon Redshift パッチ 179
- Amazon Redshift パッチ 178
- Amazon Redshift パッチ 177
- Amazon Redshift パッチ 176
- Amazon Redshift パッチ 175
- Amazon Redshift パッチ 174
- Amazon Redshift パッチ 173
- Amazon Redshift パッチ 172
- Amazon Redshift パッチ 171
- Amazon Redshift パッチ 170
- Amazon Redshift パッチ 169
- Amazon Redshift パッチ 168

Amazon Redshift パッチ 205

このパッチのクラスターバージョン:

- 1.0.477953 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2026 年 10 月 6 日にリリース

- 1.0.434008 – 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 9 月 16 日にリリース

このパッチの新機能と改善点

- Iceberg MV – 増分更新および完全な再計算が用意された、Iceberg に保存されているマテリアライズドビュー。
- システムテーブルに、各クエリに関連付けられた永続的な IAM または IAM Identity Center ID を記録する `external_user_id` 列が含まれるようになりました。これにより、クラスターライフサイクルイベント全体でユーザーごとのフィルタリングが可能になります。
- 外部テーブルに対する Amazon Redshift Spectrum クエリで、gzip および bzip2 圧縮テキスト/CSV ファイルの読み取りがサポートされるようになりました。
- COPY および UNLOAD に、キーワード IAM_ROLE 'SESSION' のサポートが追加されました。これにより、お客様は Amazon S3 アクセスに、明示的な IAM ロール ARN を指定する代わりに、セッションバインド FAS 認証情報を使用できます。
- `ALTER DATABASE ... INTEGRATION REFRESH REMEDIABLE TABLES` を追加しました。これにより、オンデマンドで重複日付警告でフラグ付けされたゼロ ETL テーブルが再同期され、オプションでスキーマを絞り込むことができます。
- ゼロ ETL 統合で `IntegrationLatestSourceCommit` Amazon CloudWatch メトリクスを追加し、観測された最新のソースコミット時間をレポートして、アイドル状態のソースと停止したパイプラインを区別できるようにしました。
- Amazon Redshift で、管理者が `ALTER SYSTEM SET passwd_min_length = N` (8 ~ 32) を使用して、データベースユーザーパスワードに必要な最小長を引き上げることができるようになりました。
- `ALTER IDENTITY PROVIDER ... ENABLE` が失敗し、自動的にロールを作成するように設定された Azure ID プロバイダーで誤ったパラメータエラーが発生する問題を修正しました。
- Amazon Redshift が、TLS 接続のポスト量子ハイブリッドキー交換 (`X25519MLKEM768`) をサポートするようになりました。
- Amazon Redshift は、フェデレーティッドアクセス許可カタログデータベースに対する `DEBUG` アクセス許可をサポートするようになりました。これにより、グローバル ID、IAM アイデンティティセンター (IdC) グループ、またはコンシューマーアカウントの管理者が、きめ細かなアクセス制御 (FGAC) で保護されたデータの未編集のセキュアなログ記録を表示できるようになります。
- Amazon Redshift は、メタデータセキュリティによるスコープ付きアクセス許可をサポートするようになりました。

- Iceberg テーブルを NOT NULL 列制約で作成できるようになりました。
- Amazon Redshift が開始した API コールの Amazon CloudTrail 監査イベントに、システムテーブルとの相関関係の queryUuid フィールドが含まれるようになりました。
- 一貫性を保つため、システムテーブルとビューで user_query_uuid 列の名前を query_uuid に変更しました。
- クエリ同時実行数が多い場合に、ゼロ ETL 統合を使用するクラスターのクエリパフォーマンスと可用性が向上しました。
- 列数が多く、列レベルのアクセス許可が少ないテーブルに対するアクセス許可チェックが迅速化され、テーブル権限を一覧表示するメタデータクエリとクライアントツールが高速化されました。
- 集計で Lambda UDF を使用するクエリの安定性を向上させました。
- 自動肥大化除去により、スーパーブロックがライブヘッダーセットに合わせて縮小するようになり、スーパーブロックバキューム (SBV) を使用して肥大化を再利用する必要がなくなりました。
- 自動テーブルソートが改善され、列数が多いテーブルのストレージの増加が減少しました。
- クラスターでの自動バキュームの実行が、クラスターが再起動されるまで停止することがある問題を修正しました。
- 同時実行スケーリングを使用する自動ワークロード管理を使用したクラスターのバキュームオペレーション中に、クラスターが再起動することがある問題を修正しました。
- 文字列リテラルを含む再帰的 CTE のタイプ推論でエラーが発生することがあるバグを修正しました。
- フェデレーティッド PostgreSQL クエリが、応答しないリモートデータベースでハングし、キャンセルリクエストを無視することがある問題を修正しました。
- メモリプレッシャー下で COPY 負荷状態管理がダブルフリーになることによるクラスターのまれな再起動を修正しました。
- LIKE 句に長いパターンを指定したときに SHOW 検出コマンドで結果が見つからないことがある問題を修正しました。
- データ共有の作成または公開が、データベースロールの最大数に達したクラスターで失敗することがある問題を修正しました。
- DROP DATABASE コマンドをキャンセルすると、予期しないクラスターの再起動を引き起こすのではなく、エラーでコマンドが終了するようになりました。
- 同時実行スケーリングが使用されているときに、Iceberg テーブルを複数回クエリし、その後ローカルテーブルでクエリを実行するストアドプロシージャの信頼性が向上しました。
- 同時実行スケーリングが有効になっているときに、Iceberg カーソルと Iceberg 以外のクエリを混在させるセッションの信頼性が向上しました。

- fdisk fetch-block IPC 待機をキャンセル可能にする機能を追加しました。これにより、失われた ACK が、強制終了できないハングではなく、強制終了可能なクエリになります。
- オンラインでの伸縮自在なサイズ変更中に使用される内部プロセスの信頼性が向上し、プロセスがより確定的で堅牢になりました。

Amazon Redshift パッチ 204

このパッチのクラスターバージョン:

- 1.0.436211 – 後続トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 9 月 22 日にリリース
- 1.0.436211 – 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 9 月 10 日にリリース
- 1.0.416217 – 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 8 月 25 日にリリース
- 1.0.394035 – CURRENT Track Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 8 月 11 日にリリース

このパッチの新機能と改善点

- 単一ノードの rg.large クラスターのサポートを追加しました。これは、単一ノードの rg.large クラスターを作成するために必要な最小クラスターバージョンです。
- システムテーブルとビューに query_uuid 列を追加し、クラスターの一時停止/再開サイクルにわたって保持されるグローバルに一意のクエリ識別子を提供しました。
- user_query_id を stl_udf_log テーブルに追加し、sys_ ビュー定義の stl_user_query_map 結合への依存関係を削除しました。
- ゼロ ETL 統合のレプリケーションの状態をモニタリングできるように、IntegrationLatestDetectedChange と IntegrationLatestAppliedChange という 2 つの新しい Amazon CloudWatch メトリクスを追加しました。
- ゼロ ETL 統合データベースの CREATE DATABASE と ALTER DATABASE に AUTO_REMEDIATION オプションを追加しました。AUTO_REMEDIATION を設定すると、Amazon Redshift はレプリケーションの問題が発生したテーブルを自動的に再同期します。詳細は svv_integration_table_state.reason に表示されます。

- Amazon Redshift ネイティブエンジンを使用した Apache Iceberg v3 テーブルの読み取り、作成、および書き込みのサポートが追加されました。これには、以前からサポートされていたすべてのデータ型におけるデフォルト値、行リネージュ、削除ベクトルのサポートが含まれます。Variant、Geography、Geometry データ型のサポートは、このリリースには含まれておらず、今後のパッチで発表される予定です。
- Iceberg テーブルを v3 にアップグレードすると、Iceberg の `timestamptz` タイプは Amazon Redshift の `TIMESTAMPTZ` タイプにマッピングされます。Iceberg v2 テーブルでは、`timestamptz` が Amazon Redshift の `TIMESTAMP` タイプにマッピングされます。v3 の場合、これはクエリがタイムゾーンに基づいてタイムスタンプを出力することを意味します。
- インスタンスタイプの最大結果セットサイズを超える `CURSOR` に対するサポートを追加しました。カーソルの結果セットが使用可能なディスク容量を超えると、Amazon Redshift は自動的にオーバーフローデータを Amazon S3 にスピルします。この機能は、AWS サポートへのリクエストに応じてクラスターごとに有効にできます。
- キャッシュアクティビティをモニタリングするための新しいビンパッキングメトリクス列を `stl_fasma_data_cache_stats` システムテーブルに追加しました。
- 遅延バインディングビュー (LBV) を参照するクエリの解析とセマンティック分析がコンシューマークラスターで実行されるようになりました。これにより、LBV 処理におけるプロデューサー側のボトルネックが解消されます。
- Amazon Redshift ML は、Amazon SageMaker に対して `CreateAutoMLJob` および `CreateProcessingJob` リクエストを行うときに、コンテナ間のトラフィック暗号化オプションを有効にするようになりました。これは、`BuildTrainingRequest` に既に適用されている設定と一致し、セキュリティスキャナーアラートを防止します。
- 高頻度ワークロードのゼロ ETL 統合パフォーマンスが向上し、多数の同時更新が発生するテーブルでのレプリケーション遅延が短縮されました。
- ゼロ ETL 統合では、ターゲット列の長さを超えるソースデータが取得不可能なエラーとして正しく報告されるようになりました。
- クラスター再起動後のゼロ ETL 統合の安定性が向上しました。
- ゼロ ETL 統合でテーブルフィルターが変更されてスキーマからすべてのテーブルが削除されると、空のスキーマが自動的にクリーンアップされるようになりました。
- アクティブなゼロ ETL 統合が設定されているデータベースで DROP DATABASE を試行する際のエラーメッセージを改善しました。
- DynamoDB のゼロ ETL 統合では、最初のシードと継続的なレプリケーションの両方で ACCEPTINVCHARS および TRUNCATECOLUMNS オプションが優先されるようになりました。ソースに無効な UTF-8 文字が含まれている場合、テーブルのレプリケーションは継続します。

- ゼロ ETL 統合のクラスター、特に多くのデータベースやテーブルをレプリケートするワークロードでのクエリパフォーマンスが向上しました。
- 一時テーブルを使用するワークロードのパフォーマンスが向上し、カタログアクセスのオーバーヘッドが軽減され、全体的なスループットが向上しました。
- SHOW TABLES、SHOW SCHEMAS、SHOW COLUMNS、SHOW GRANTS などの SHOW 検出コマンドは、最大 133,500 個のオブジェクトを返すようになりました。そのため、オブジェクト数が非常に多いスキーマやデータベースのメタデータの一覧表示が、行制限エラー 10,000 で失敗することなく、正常に実行されるようになりました。
- ユーザーがオブジェクトを所有しているか権限を保持しているために DROP USER コマンドが拒否された場合、エラーメッセージは、特定のブロッキングオブジェクトの名前と、残りの依存関係の数を報告するようになりました。
- Amazon Redshift は、IAM ロール認証情報をリーダーノードでキャッシュするようになりました。これにより、COPY、UNLOAD、Amazon Redshift Spectrum、Lambda UDF クエリの取得オーバーヘッドが削減されます。頻繁に使用される認証情報はクエリ間で再利用されるため、クエリの起動時間が短縮されます。
- AWS IAM アイデンティティセンター認証が、拡張 VPC ルーティングが有効になっているクラスターとワークグループで機能するようになりました。
- RG ノードタイプのディスクヘルス検出が改善され、通常のディスク I/O で動作するクラスターでの、不要な自動ハードウェア修復が防止されるようになりました。
- バックエンドプロセスで余分なコピーをスキップすることで、データ共有のパフォーマンスが向上しました。
- マルチウェアハウス書き込みクエリにおけるコンシューマーサイズのサポートを拡張しました。
- 結合操作における、引数を持たない Lambda ユーザー定義関数の安定性を強化しました。
- 述語に IDF が含まれている場合の MDDL ソートキー計算で発生していた問題を修正しました。
- BEGIN READ ONLY トランザクション内で実行された際、外部テーブル、データ共有オブジェクト、およびクロスデータベース参照に対するクエリが「トランザクションは読み取り専用です」となって失敗する問題を修正しました。
- 同じテーブルで COPY または INSERT が自動テーブル最適化と同時に実行された場合に、クラスターが再起動する可能性がある、まれな状態を修正しました。
- 古いクラスターバージョンで作成されたデータ共有を介してアクセスされたテーブルで、自動テーブル最適化が失敗する可能性がある、まれなエンコードの不一致エラーを修正しました。
- データ分散が歪んだテーブルの自動バキュームによってクラスターが再起動する可能性がある、まれな状態を修正しました。

- Auto COPY ジョブが進行を停止し、同じクラスターでの他の Auto COPY オペレーションをブロックする問題を修正しました。
- クエリのクリーンアップ中の競合状態が原因で COPY コマンドによってクラスターが再起動する可能性がある、まれな状態を修正しました。
- 外部テーブルの LIMIT 句を持つクエリに対して svl_s3query_summary.s3query_returned_rows が 1 行余分に報告する場合がある問題を修正しました。
- ゼロ ETL 統合が設定されており、Auto COPY ジョブが設定されていないクラスターにおいて、ブートストラップ中に発生する可能性のある、まれなクラスター再起動の問題を修正しました。
- ESCAPE オプションで複数文字の区切り文字を使用する場合の COPY フィールド解析の問題を修正しました。
- IAM アイデンティティセンター経由で接続されたユーザーが CREDENTIALS パラメータを使用して、明示的で一時的な認証情報を提供したときに、COPY コマンドと UNLOAD コマンドが失敗する問題を修正しました。
- SHOW CONSTRAINTS において、クラスターの再起動を引き起こす可能性がある問題を修正しました。
- ロールに付与された CONNECT アクセス許可によって、マッピングされたグループの AWS IAM アイデンティティセンターユーザーが、Amazon Redshift フェデレーティッドアクセス許可が有効になっているデータベースに接続できない問題を修正しました。
- SUPER 配列で UNNEST を使用するクエリが、同時実行スケーリングクラスターで実行される際に、誤ってブロックされる問題を修正しました。
- ネストされた SUPER 構造に対して ARRAY_SORT 関数を適用した際、パスナビゲーションエラーにより失敗する場合がある問題を修正しました。
- 深くネストされた PartiQL 式または SUPER 式によってクラスターが再起動する可能性がある問題を修正しました。
- 形式文字列で精度が 38 を超えると TO_NUMBER 関数が失敗することがある問題を修正しました。
- 暗号化キーのローテーションにより、グラフトされたテーブルのキーが破損し、データアクセスが失敗する可能性がある問題を修正しました。
- クエリがハングしたり、ノードが応答しなくなる可能性があるストレージオペレーションのデッドロック状態を修正しました。
- ディスクブロック管理中にまれにクラッシュし、ノードが再起動する可能性がある問題を修正しました。

- ブロックの割り当て中にメモリ不足が発生し、ノードが再起動する原因となるクラッシュを修正しました。
- ブロックレプリケーションの割り当て中にディスクがいっぱいの状態でクラッシュし、ノードが再起動する可能性がある問題を修正しました。

Amazon Redshift パッチ 203

このパッチのクラスターバージョン:

- 1.0.417950 – 後続トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 8 月 26 日にリリース
- 1.0.417950 – 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 8 月 25 日にリリース
- 1.0.400034 – 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 8 月 17 日にリリース
- 1.0.384821 – CURRENT Track Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 8 月 4 日にリリース
- 1.0.377293 – 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 7 月 25 日にリリース
- 1.0.365190 – 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 7 月 21 日にリリース
- 1.0.347559 – 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 7 月 6 日にリリース

このパッチの新機能と改善点

- 新しい列が stl_alter_dist_sort に追加され、ALTER DISTKEY オペレーションと ALTER SORTKEY オペレーションがメインクラスター、データ共有コンシューマー、または同時実行スケーリングクラスターで実行されたかどうかが示されます。
- 特定のユーザーグループに付与されたアクセス許可を検出する SHOW GRANTS FOR GROUP 構文を追加しました。
- チェックポイントの詳細な成功ステータスを追加することでゼロ ETL 統合オブザーバビリティを強化し、システムテーブルのレプリケーションの進行状況をよりきめ細かくモニタリングできるようになりました。

- ゼロ ETL 統合内のソーステーブルスキーマと DDL メタデータでサポートされていない UTF-8 文字の適切な処理を導入しました。これにより、統合の失敗を防ぎます。
- Amazon Redshift は、Amazon Kinesis Data Streams (KDS) に接続されたストリーミングマテリアライズドビュー (MV) の更新の同時実行スケーリングをサポートするようになりました。
- パッチ 203 以降、Amazon Redshift は既存の Python UDF のサポート終了の適用を開始します。新しい Python UDF の作成はパッチ 198 で廃止され、Python UDF のすべてのサポートは 2026 年 6 月 30 日をもって終了します。既存の Python UDF を Lambda UDF に移行して、継続的な機能利用を確保してください。移行ガイダンスについては、「Amazon Redshift Python user-defined functions will reach end of support after June 30, 2026」および AWS Big Data Blog の「Best practices for Amazon Redshift Lambda user-defined functions」を参照してください。動作の変更の詳細については、「スカラー Python UDFは 2026 年 6 月 30 日にサポートが終了します」を参照してください。
- Amazon Redshift Spectrum から COPY でロードされたテーブル、明示的な列エンコーディングを使用したテーブル、または空でないテーブルにロードされたテーブルは、テーブルにプライマリキーがある場合に DISTKEY の自動割り当ての対象となり、プライマリキーでの結合のクエリパフォーマンスが向上します。
- データ共有テーブルの ALTER TABLE … ALTER DISTKEY エラーメッセージを更新し、お客様向けのテーブル名を表示するようにしました。
- テーブル数が非常に多いクラスターの自動バキュームによるバックグラウンドアクティビティを減らし、以前に過剰なメンテナンスアクティビティが発生していたクラスターの安定性を改善しました。
- 特定のケースにおいて、不一致のフィルター、集計、ウィンドウ関数を含むサブクエリを組み合わせることで、クエリのパフォーマンスを最適化しました。
- テーブルの再同期後に Amazon DynamoDB 履歴モードを使用したゼロ ETL テーブルで、タイムトラベルクエリのタイムスタンプ精度を改善しました。
- 特定のプライマリキー設定を持つテーブルで CDC チェックポイントを処理する際に、ゼロ ETL 統合が再同期ループに入ることがなくなりました。
- 無効なチャネルフィルター式に関するゼロ ETL 統合のエラーレポートを改善しました。
- 元のクエリテキストはセキュアなログ記録でマスクされなくなりました (外部の AWS Lake Formation ビューまたはフェデレーティッド Amazon Redshift アクセス許可を持つデータベースに対するクエリに適用されます)。
- 大きなテーブルでの ANALYZE のパフォーマンスを最大 30% 向上させました。
- ROLLUP 集計で AWS Lambda ユーザー定義関数を使用するクエリのパフォーマンスを強化しました。

- ハッシュ結合の外部テーブルで不要なデータスキャンをスキップすることで、LIMIT 句を使用した ORDER BY のクエリパフォーマンスを強化しました。
- バックグラウンドのツリースイープオペレーション中のメモリ消費量を削減し、フリート全体でのノードクラッシュを排除しました。サバイバーカウントのバッチドレインを実装することで、コンピューティングノードのツリースイープのメモリフットプリントを制限し、高いメタデータチャーン下での無制限のメモリ増加を防止しました。ツリースイープパスは、バックグラウンドツリー再生の完了時に制限されるようになりました。
- Amazon S3 の読み取りに失敗した後の障害復旧を改善し、ノードのクラッシュを排除しました。Amazon S3 の読み取りが失敗した場合、その後にトリクルツリーイテレーターが適切にリセットされるようになりました。これにより、古いデータや不完全なデータをサイレントに返すのではなく、後続の読み取りが正しく再試行されます。
- アダプティブクラスターのサイズ変更後に ALTER TABLE … ALTER DISTSTYLE AUTO で変更されたテーブルが、sys_restore_state で無期限に保留状態のままになる問題を修正しました。
- SYS_QUERY_TEXT システムビューに空白がないクエリテキストが表示されたり、チャンク境界で単語が不正確に分割されたりする問題を修正しました。
- Amazon Redshift Catalog メタデータの不整合が原因で、2023 年 8 月 ~ 2024 年 3 月に作成されたクラスターでゼロ ETL 統合によるスタートアップの失敗が発生する問題を修正しました。
- SHOW GRANTS で、identity_type として「role」を持つユーザーグループに付与されたアクセス許可が誤って表示される問題を修正しました。
- セッションの中断処理中にクエリが無期限にハングする問題を修正しました。
- SYSLOG ACCESS UNRESTRICTED を持つスーパーユーザー以外のユーザーが、すべてのユーザーのセッションを想定どおりに表示できず、sys_session_history で自分のセッションデータしか表示できない問題を修正しました。
- データベースまたはユーザー接続の制限が 1000 以上に設定されていると切り捨てられ、正しい接続数が表示されない問題を修正しました。
- メモリプレッシャー下でクエリが無期限にハングする問題を修正しました。
- 一時停止操作中に Amazon Redshift Serverless エンドポイントが短時間使用できなくなる問題を修正しました。
- クエリのコンパイル中に、特定のパラメータの組み合わせを持つ COPY コマンドが予期せず失敗する問題を修正しました。
- APPROXIMATE PERCENTILE_DISC と GROUP BY を使用するクエリがコンパイルエラーで失敗する問題を修正しました。
- COLLATE 式を使用するクエリが誤った列値を返す可能性がある、誤った結果の問題を修正しました。

- SUPER 型の値を持つ LEAST 関数と GREATEST 関数でクエリが失敗する問題を修正しました。
- REGEXP_REPLACE で、キャプチャグループ参照を持つ置換文字列が切り捨てられた出力を生成する可能性がある、誤った結果の問題を修正しました。
- 再レプリケーション読み取りの中止中に、データが破損する可能性がある問題を修正しました。再レプリケーション読み取りが実行中に中止された場合に発生する可能性があるダブルフリー条件を解決しました。これにより、メモリの破損やクラッシュを防ぎます。

Amazon Redshift パッチ 202

このパッチのクラスターバージョン:

- 1.0.369760 – TRAILING Track Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 7 月 29 日にリリース
- 1.0.377292 – TRAILING Track Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 7 月 25 日にリリース
- 1.0.377292 – 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 7 月 25 日にリリース
- 1.0.358853 – TRAILING Track Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 7 月 21 日にリリース
- 1.0.358853 – 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 7 月 14 日にリリース
- 1.0.343555 – 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 6 月 29 日にリリース
- 1.0.340728 – 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 6 月 22 日にリリース
- 1.0.331549 – 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 6 月 16 日にリリース
- 1.0.323213 – 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 6 月 3 日にリリース
- 1.0.314775 – 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 5 月 28 日にリリース

このパッチの新機能と改善点

- RG ノードタイプ (rg.large および rg.12xlarge マルチノード) のサポートが追加されました。これは、rg.large と rg.12xlarge に移行するために必要な最小クラスターバージョンです。
- Lakehouse タイプの無効な状態にある IAM Identity Center ID プロバイダーを削除するスーパーユーザーのサポートを追加
- STL_QUERY および STL_QUERYTEXT システムテーブルに「JIT ANALYZE」というラベルが付いた内部メンテナンスクエリを表示しました。このクエリは Amazon Redshift によって自動的に実行され、外部テーブル統計を収集し、クエリプランを最適化します。お客様の操作は必要ありません。
- Kinesis ストリーム ID 検証を最適化して、DescribeStreamSummary API コールの数を減らしました。これにより、多くの Kinesis ストリーム MV で、Redshift のお客様が Kinesis API クオータに達することを回避できます。詳細については、Redshift Streaming のドキュメントセクションを参照してください。
- GROUP BY 句、ORDER BY 句、および LIMIT 句を持つクエリに Top K 集約を使用することで、クエリ実行のパフォーマンスが向上し、集計中に処理されるデータ量が減りました。
- 複合フィルター述語 (OR、IN、NOT IN) を Glue GetPartitions API にプッシュすることで、パーティション化された AWS Glue データカタログテーブルに対するクエリのパフォーマンスを強化し、取得されるパーティションの数を減らしました。
- ウィンドウ関数で Lambda ユーザー定義関数を使用するクエリのパフォーマンスを強化しました。
- ハッシュ結合の結合フィルターで Lambda ユーザー定義関数を使用する特定のクエリのパフォーマンスを強化しました。
- 結合で Lambda UDF を使用するクエリの安定性を向上させました。
- Lambda ユーザー定義関数を作成するときに、AWS Lambda の名前として ARN の使用を許可します。
- COALESCE 式と CASE WHEN 式のメモリ使用量を最適化することで、クエリのパフォーマンスが向上しました。
- 一度評価されたサブプランを含むクエリに高度なフィルタリングを適用し、残りの実行全体で結果を定数として再利用します。
- ST_Transform を使用した初回実行クエリのパフォーマンスの向上
- AWS Glue 圧縮によって生成されることがある重複行インデックスを含む位置削除ファイルを受け入れることで、Apache Iceberg の互換性が向上しました。

- 削除ファイルの参照で s3a:// パスを受け入れることで、Apache Iceberg の互換性が向上しました。
- 正しく設定されていないテーブルを処理する際の Spectrum パーティションレベルの取得エラーの処理を改善
- USE を介してデータベース間でネストされた SQL UDF を呼び出すストアドプロシージャの安定性を改善
- データ共有への追加や削除など、ストアドプロシージャでサポートされていないオペレーションのエラーメッセージを改善しました。
- EVR の使用時にお客様の VPC 情報を検出できないバグを修正しました。この情報はローカル Redshift ML モデルのトレーニングにのみ必要であり、他の Redshift 機能には影響しません。これにより、既存のトレーニング済み Redshift ML モデルを使用できなくなることはありませんでした。
- 内部クエリ ID カウンターが最大値に近づいたときにクラスターが一時停止して再開した後に、重複したクエリ ID がシステムテーブルに表示されることがある問題を修正しました。
- 内部キャッシュの不整合により、マルチ AZ フェイルオーバー後に STL システムテーブルのクエリが不完全な結果を返すことがある問題を修正しました。
- 拡張行の説明の isAutoIncremented フィールドが IDENTITY 列に対して誤った値を返すことがある問題を修正しました。
- ARRAY_CONTAINS 関数と ARRAY_POSITION 関数が NULL 入力を指定したときに期待どおりに NULL を返さない問題を修正しました。
- ベクトル化されたスキャン中に複数の結合フィルター述語を持つクエリが予期せず失敗することがある、まれな問題を修正しました。
- 同時 COPY オペレーションによってデータロードプロセスでクラスターが不安定になることがある、まれな問題を修正しました。
- バキュームや分析などの自動メンテナンス操作により、プロセスのシャットダウン中にクラスターが不安定になることがある問題を修正しました。
- ワークロード管理 (WLM) のショートクエリアクセラレーションが動的にオンとオフに切り替わった後に、クエリが失敗することがある問題を修正しました。
- 履歴モードのゼロ ETL テーブルでタイムトラベルクエリが発生する問題を修正しました。
- クライアントの同時シャットダウン中に、ゼロ ETL 統合が応答しなくなることがある問題を修正しました
- ソースシャードを共有するゼロ ETL テーブルが、不要な再同期をトリガーすることがある問題を修正しました

- 内部理由フィールドが許可された長さを超えた場合に、ゼロ ETL 統合テーブルの状態の更新が失敗することがある問題を修正しました。
- Aurora PostgreSQL パーティションテーブルの列変更中に、ゼロ ETL 統合でエラーが発生する問題を修正しました

Amazon Redshift パッチ 201

このパッチのクラスターバージョン:

- 1.0.331388 – 後続トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 6 月 16 日にリリース
- 1.0.331388 – 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 6 月 15 日にリリース
- 1.0.326225 – 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 6 月 8 日にリリース
- 1.0.319797 – 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 6 月 3 日にリリース
- 1.0.313685 – 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 5 月 27 日にリリース
- 1.0.300094 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2026 年 5 月 12 日にリリース
- 1.0.298697 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2026 年 5 月 7 日にリリース
- 1.0.292316 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2026 年 5 月 4 日にリリース
- 1.0.280974 – 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2026 年 4 月 22 日にリリース

このパッチの新機能と改善点

- RG ノードタイプ (rg.xlarge および rg.4xlarge マルチノード) のサポートが追加されました。これは、rg.xlarge と rg.4xlarge に移行するために必要な最小クラスターバージョンです。
- Iceberg テーブルでの ALTER のサポートを追加しました。
- AwsDataCatalog にマウントされた Iceberg テーブルへの書き込みのサポートを追加しました。

- USE コマンドを使用したデータベースへの接続時の SHOW GRANTS コマンドのサポートを追加しました。
- フェデレーティッドアクセス許可を持つ ID の SHOW GRANTS コマンドと SHOW COLUMN GRANTS コマンドのサポートを追加しました。
- Amazon Redshift Serverless においてバーストキャパシティを使用し、マテリアライズドビューの自動更新をユーザーワークロードとして扱うサポートを追加しました。
- ワイドテーブルへの書き込みクエリに対して、より多くのクラスターメモリを割り当てることを AutoWLM に許可します。
- Amazon Redshift は、SVV_ATTACHED_MASKING_POLICY システムビューのクエリパフォーマンスを向上させ、多数のアクセス許可と動的データマスキングポリシーアタッチメントを持つインスタンスの実行時間を短縮しました。
- librdkafka ライブラリをバージョン 2.8.x から 2.13.x にアップグレードし、最新のバグ修正とパフォーマンスの強化を行いました。
- 集計の HAVING 句で Lambda ユーザー定義関数を使用するクエリのパフォーマンスを強化しました。
- ウィンドウ関数で Lambda ユーザー定義関数を使用するクエリのパフォーマンスを強化しました。
- 引数を持たない Lambda ユーザー定義関数の安定性を強化しました。
- ORDER BY 句と LIMIT 句を使用する場合、不要なデータスキャンを排除することで、データレイクテーブルのクエリ実行パフォーマンスを強化しました。
- ゼロ ETL 統合のエラーメッセージを改善し、サポートされていない制約、サポートされていない DDL ステートメント、日付精度損失、マルチバイト文字長の不一致、スキップされたソーステーブルの詳細をより明確にしました。
- レプリケーションステータスチェック時の認証情報取得の失敗を適切に処理することで、ゼロ ETL 統合の耐障害性を改善しました。
- 同時レプリケーションタスクを実行する際のゼロ ETL 統合のパフォーマンスを改善しました。
- 行が頻繁に更新されるゼロ ETL テーブルのタイムトラベルクエリ結果を改善し、各履歴行バージョンに個々の削除タイムスタンプが反映されるようにしました。
- AI 駆動スケーリングによってメインクラスターのサイズが適切に変更されなかったエッジケースを修正しました。
- ワークロード管理 (WLM) によってクエリが削除されて再試行されたときに、Amazon CloudWatch メトリクスが失敗したクエリ数を重複して報告する場合がある問題を修正しました。

- Redshift Table Copy を使用した COPY で、分散スタイルの不一致を引き起こす可能性がある問題を修正しました。
- 範囲が制限されたスキャン結果に影響を与える可能性のあるパーティションソート終了行の計算が正しくない問題を修正しました。
- テーブルが同時に削除されると自動バキューム処理が失敗する可能性がある問題を修正しました。
- VACUUM がリレーションの取得時に古いテーブルバージョンを使用してエラーを引き起こす可能性がある問題を修正しました。
- バーストクラスターにおいて、バキュームのトランザクションマップが適切にクリーンアップされるようにしました。
- データ共有を介してアクセスすると、ゼロ ETL でレプリケートしたテーブルに対するクエリが失敗する可能性がある問題を修正しました。
- クラスターの再起動を引き起こす可能性があるゼロ ETL 統合データレプリケーションの問題を修正しました。
- 更新マージオペレーション中に DynamoDB 履歴モードとのゼロ ETL 統合が失敗する問題を修正しました。
- Amazon Redshift コンソールでゼロ ETL 統合テーブルに誤った行数が表示される問題を修正しました。
- Amazon Redshift Serverless ワークグループがゼロ ETL 統合オペレーション中に一時停止できない問題を修正しました。
- 空の Avro ファイルが原因で COPY 取り込みが失敗する問題を修正しました。空の Avro ファイルをスキップし、エラーなしで取り込みが続行されるようになりました。
- 位置に依存しないスーパーブロックを使用する場合のイプシロンツリーのリプレイ中における古いブロックヘッダーのチェックサム問題を修正しました。

Amazon Redshift パッチ 200

このパッチのクラスターバージョン:

- 1.0.314426 – 後続トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン – 2026 年 5 月 27 日にリリース
- 1.0.293472 - 後続トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2026 年 5 月 7 日にリリース
- 1.0.293472 – 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2026 年 4 月 30 日にリリース

- 1.0.266483 – 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2026 年 4 月 9 日にリリース
- 1.0.257586 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2026 年 3 月 27 日にリリース

このパッチの新機能と改善点

- 多次元データレイアウトソートキーを使用しているテーブルで INSERT オペレーションが失敗する場合がある問題を修正しました。
- ワークロードベースのソートによってメリットが得られるテーブルに対する、Amazon Redshift Advisor による多次元データレイアウト (MDDL) ソートキーの推奨事項を改善しました。
- Redshift Serverless の AI 駆動のスケーリングと最適化を強化して、ディスク使用量シグナルを改善し、より正確で応答性の高いスケーリングの決定を可能にしました。
- Redshift Serverless AI 主導のスケーリングと最適化を強化し、プロビジョニングされたクラスタースナップショットをサーバーレス名前空間に移行するときに初期容量をインテリジェントに決定しました。
- キャッシュされた ID トークンがコンシューマーで期限切れになると、IAM アイデンティティセンターの ID 伝播を使用したデータ共有クエリが失敗する可能性がある問題を修正しました。
- IAM アイデンティティセンターのグループの同期により、後続のバックグラウンドメンテナンスオペレーションが失敗する可能性がある問題を修正しました。
- IAM アイデンティティセンターの ID プロバイダー設定を別のアプリケーション ARN で更新すると、クラスターが再起動する可能性がある問題を修正しました。
- 特定の AWS リージョンで IAM アイデンティティセンターの ID 伝播によるデータ共有クエリが失敗する可能性がある問題を修正しました。
- ID プロバイダーの設定変更が一貫して伝播されるようにすることで、IAM アイデンティティセンターの統合の信頼性を改善しました。
- 一部の相関 NOT EXISTS サブクエリをアンチジョインに非相関化することで、クエリ計画を最適化しました。
- Apache Iceberg テーブルで DROP TABLE IF EXISTS を実行したときに、対象のテーブルが存在しない場合、成功ではなくクエリ失敗を誤って報告する可能性がある問題を修正しました。
- ユーザーがカタログテーブルに対して ANALYZE を実行しようとした場合の適切なエラーメッセージを追加しました。
- Spectrum COPY の一般提供に伴い、Spectrum ベースの COPY オペレーションにおける一時テーブルへの依存がなくなりました。

- 耐久性の高い永続性タスクが追いつかない場合に S3 通知キューの読み取りをスロットリングすることで Auto COPY の信頼性を向上させ、COPY JOB の状態を調整するための S3 リストオペレーションの必要性を削減しました。
- 同時実行スケーリングが、ゼロ ETL テーブルでの読み取りおよび書き込みオペレーションで一般利用可能になりました。
- 同時ディスクコミットが、コミットされていないテーブル状態の追跡に干渉した場合に、クラスターが再起動する可能性があるゼロ ETL CDC レプリケーションの競合状態を修正しました。
- テーブル状態とクライアント状態の両方で最新の取り込み位置を使用して、既に適用されたデータをフィルタリングすることで、ゼロ ETL CDC 取り込みの信頼性を向上させました。
- 再同期ライフサイクル全体でチェックポイントの下限と再同期状態を正しく追跡することで、既に適用された古いチェックポイントがテーブルの障害後に再び取り込まれる可能性があるゼロ ETL テーブルの再同期におけるデータ信頼性の問題を修正しました。
- ゼロ ETL エラーメッセージを改善し、列の長さの超過や無効な UTF-8 文字エラーの詳細など、障害の理由をより明確にしました。
- SYS_LUDF_DETAIL システムビューを強化し、ミリ秒単位ではなくマイクロ秒単位で期間をレポートすることで、オブザーバビリティを向上させました。
- 自動 COPY ジョブの作成がゼロ ETL データベースで誤って許可されていた問題を修正しました。
- 追加の組み込み関数を RLS セーフとして分類することで、行レベルセキュリティ (RLS) で保護されたテーブルのクエリパフォーマンスを改善し、以前にパフォーマンスが低下していた一般的なクエリパターンの述語プッシュダウンを可能にしました。
- Apache Iceberg テーブルにおける DELETE、UPDATE、MERGE オペレーションのサポートを追加しました。
- 位置に依存しないスーパーブロックのサポートを追加し、データ量に関わらず、予測可能なクラスターの再起動時間を実現しました。
- GUID ベースのディスクキャッシュを有効にし、ディスクキャッシュのスーパーブロック位置への依存をなくし、データ量に依存しない予測可能なクラスターの再起動時間を実現しました。
- 自動テーブル分散の最適化を改善しました。1 つのプライマリキーを持つテーブルは、ALL ディストリビューションから KEY ディストリビューションへの自動変換の対象となり、クエリのパフォーマンスが向上し、ストレージのオーバーヘッドが削減されました。
- バキュームオペレーションと同時 DML オペレーションが同じストレージオブジェクトにアクセスしたときに、競合により VACUUM オペレーションが失敗したり、誤った結果を生成したりする問題を修正しました。
- 特殊なパーティションチェーンを処理するときに、範囲外メモリアクセスにより、VACUUM FULL が特定のテーブルで失敗する可能性がある問題を修正しました。

- ALTER TABLE OWNER オペレーションにより、誤ってオブジェクトの所有権が内部ブートストラップユーザーに変更されてしまう可能性がある問題を修正しました。
- VACUUM オペレーションとステージングテーブルの同時再利用によってクラスターが不安定になる可能性がある競合状態を修正しました。
- 処理されたパーティションの数が内部制限を超えた場合に、大きなテーブルの VACUUM FULL または VACUUM SORT が内部エラーで失敗する問題を修正しました。
- リモートクラスターから統計を集約している間にテーブルが削除されると、バックグラウンドパーティション統計の収集が失敗する可能性がある問題を修正しました。
- 不要なデータ更新を回避し、ソートされたテーブルのバキュームに要する時間とリソースを削減することで、VACUUM SORT オペレーションのパフォーマンスを改善しました。
- バキュームオペレーション中に内部パーティションデータに正しくアクセスしないと、予期しないエラーやクラスターの不安定性が発生する問題を修正しました。
- ALTER TABLE オペレーションにより、テーブルの所有権が非特権ユーザーに誤って変更される可能性がある問題を修正しました。ALTER オペレーション中の所有者の変更は、承認されたユーザーのみに正しく制限されるようになりました。
- バキュームオペレーション中の内部制限チェックにより、パーティションの数が多いテーブルで不要なクエリ失敗が発生するという、まれな問題を修正しました。
- 削除されたテーブルの検出時にバックグラウンド統計の収集が失敗したり、エラーが発生したりする問題を修正し、クラスターの安定性を向上させました。
- バキュームオペレーションで古いテーブルメタデータを使用し、スキーマが同時に変更されるテーブルのバキューム中に誤った動作が発生する可能性がある問題を修正しました。
- CURRENT メンテナンストラックのAmazon Redshift パッチ P198において、新しいクエリ (クエリの最初の実行) のパフォーマンスを向上させる新しいクエリコンパイル機能強化をリリースしました。現在、この機能は、TRAILING トラックの Amazon Redshift パッチ P200 から始まるすべてのリリースで有効になっています。

Amazon Redshift パッチ 199

このパッチのクラスターバージョン:

- 1.0.266409 - 後続トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2026 年 4 月 20 日にリリース
- 1.0.252420 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2026 年 3 月 25 日にリリース

- 1.0.232773 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2026 年 3 月 5 日にリリース

このパッチの新機能と改善点

- Redshift CURRENT メンテナンストラックの新しいクエリ (クエリの最初の実行) のパフォーマンスを向上させる新しいクエリコンパイル機能強化をリリースしました。TRAILING トラックは次回のパッチリリースで配信されます。
- プロデューサー側の新しい被付与者に対する Redshift フェデレーションアクセス許可を使用して、コンシューマー側における列レベルの権限のサポートを有効にしました。
- Amazon Redshift フェデレーションアクセス許可は、遅延バインドビュー (LBV) による外部テーブルのクエリをサポートしています。
- ALTER EXTERNAL SCHEMA を KINESIS 外部スキーマタイプで使用して、既存の外部スキーマの IAM_ROLE と REGION を変更できるようになりました。
- Redshift RA3 および Serverless データウェアハウスでの NO-BACKUP テーブルの復元サポートを追加しました。これらのインスタンスタイプは NO-BACKUP 機能をサポートしていないため、NO-BACKUP とマークされたテーブルは常にスナップショットに含まれ、永続テーブルとして適切に復元されるようになりました。
- Amazon Redshift では、IAM ロールまたは IAM ユーザーを使用して接続するときに、外部テーブルへの書き込みをサポートするようになりました。以前は、外部テーブルに対する読み取りオペレーションのみが IAM ロールベースの認証でサポートされていました。この更新により、Redshift は外部テーブルのデータを Amazon S3 に書き込むときに IAM セッション認証情報を正しく使用し、完全な読み取りおよび書き込みアクセスを有効にします。
- Kafkaストリームの MV を自動更新する際、取り込むべき新しいレコードがあるかどうかを判定するために、より効率的なヒューリスティック (発見的問題解決法) を採用するようになりました。これにより、このチェックのために Kafka ブローカーから転送されるデータの量が減少し、その後の MV 更新を開始する際のレイテンシーが短縮されます。
- FIPS 準拠の SHA384 暗号化を使用するようにバックアップおよび復元オペレーションをアップグレードし、レガシー MD5 ハッシュを置き換えました。
- Lambda ユーザー定義関数と UNNEST ステートメントを組み合わせてクエリのパフォーマンスを向上させました。
- システムテーブルおよびビューに対する Lambda ユーザー定義関数を使用してクエリのパフォーマンスを向上させました。

- 不要なデータスキャンを排除することで、ORDER BY 句と LIMIT 句を使用する場合のクエリ実行パフォーマンスを強化しました。
- Redshift Serverless の AI 駆動スケーリングと最適化を強化して、サポートされる RPU の範囲を拡張し、8 RPU 以上をサポートするようになりました。
- Amazon Redshift マテリアライズドビュー (MV) の自動 REFRESH クエリは、バックグラウンドの自律プロセスとしてではなく、ユーザークエリとして扱われるようになりました。自動 REFRESH クエリは、他のユーザークエリと同じ優先度で実行され、自律バックグラウンドプロセスによって延期されなくなりました。MV の自動 REFRESH の動作変更機能は、Amazon Redshift でプロビジョニングされたクラスターにおいて、パッチリリース P198 以降の CURRENT トラックでのみ有効になります。Serverless では現在無効になっています。
- 自動タスクのスケジュールが改善され、ストリーミングインジェスト、自動コピー (COPY JOB)、マテリアライズドビューの自動更新など、ユーザーにとって不可欠なタスクが優先的に実行されるようになりました。
- pg_last_query_id() が、マルチ AZ クラスターのユーザークエリ ID ではなく、書き換えられたクエリ ID を誤って返す問題を修正しました。
- SHOW GRANTS が 1 部構成のオブジェクト参照を処理できない問題を修正しました。
- pg_class_extended の同時更新により、DROP TABLE がカタログ再構築ワーカーのタイムスタンプ自動ワーカーと競合していた問題を、DROP TABLE 時に削除をスキップし、タイムスタンプワーカーが次の実行時にクリーンアップできるようにすることで修正しました。
- データベース、スキーマ、またはテーブルの名前に include/exclude キーワードが含まれていた場合のゼロ ETL チャネルフィルター解析の問題を修正しました。
- ゼロ ETL テーブルオペレーションと同時にバキュームオペレーションが実行された場合に発生する可能性がある競合状態を修正しました。
- ゼロ ETL 統合が、KMS 暗号化コンテキストの変更時にエラー状態に移行せずに、再同期オペレーションを誤って試行する問題を修正しました。
- データレプリケーションがアクティブである場合、クラスターの再起動中に発生する可能性があるゼロ ETL 統合の競合状態を修正しました。
- テーブルの所有者が削除されたときに SHOW TABLES コマンドがエラーを返す問題を修正しました。

Amazon Redshift パッチ 198

このパッチのクラスターバージョン:

- 1.0.239402 - 後続トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2026 年 3 月 18 日にリリース
- 1.0.227967 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2026 年 2 月 25 日にリリース
- 1.0.224252 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2026 年 2 月 19 日にリリース
- 1.0.219674 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2026 年 2 月 11 日にリリース
- 1.0.212832 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2026 年 2 月 10 日にリリース
- 1.0.204436 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2026 年 2 月 4 日にリリース

このパッチの新機能と改善点

- Redshift CURRENT メンテナンストラックの新しいクエリ (クエリの最初の実行) のパフォーマンスを向上させる新しいクエリコンパイル機能強化をリリースしました。TRAILING トラックは次回のパッチリリースで配信されます。
- 列を定数値と比較する際に適切な照合ルールを適用することで、INTERSECT クエリと EXCEPT クエリを改善しました。
- 手動で作成されたマテリアライズドビューの自動更新を有効にして、同時実行スケーリングクラスターで実行します。
- メインクラスターと同時実行スケーリングクラスター間の直接接続を有効にして、レイテンシーを減らし、接続スループットの制限を排除することで、接続パフォーマンスを向上させます。
- ゼロ ETL 統合のエラー分類が改善され、根本原因の特定と問題のデバッグが容易になりました。
- 正常に実行されたクエリのエラーメッセージを SYS_QUERY_HISTORY から削除し、よりクリーンなクエリ履歴レコードを提供します。
- SYS_QUERY_HISTORY のバーストクエリのコンパイル時のログ記録の精度が向上し、パフォーマンスに関するインサイトが向上します。
- ARRAY_CONTAINS、ARRAY_POSITION、ARRAYS_OVERLAP、ARRAY_INTERSECTION、ARRAY_S の配列関数のサポートが追加されました。
- SUPER オブジェクト内の属性の数を返す GET_NUMBER_ATTRIBUTES 関数のサポートが追加されました。

- 失敗したトランザクションのアイドルトランザクションタイムアウトを有効にして、リソース管理を改善します。
- 再利用可能な形式パラメータを定義できる TEMPLATE オブジェクトタイプのサポートが追加され、オペレーションごとに手動で指定する必要がなくなりました。
- 共通テーブル式 (CTE) で Lambda ユーザー定義関数 (LUDF) を呼び出すクエリのパフォーマンスが向上しました。
- CASE 式の THEN 句で Lambda ユーザー定義関数 (LUDF) を呼び出すクエリのパフォーマンスが向上しました。
- 新しい Python ユーザー定義関数の作成は廃止されました。既存の Python UDF は、2026 年 6 月 30 日までサポートされます。継続的なサポートを受けるには、Lambda ユーザー定義関数 (LUDF) に移行します。
- 新しいシステムテーブル SYS_LUDF_DETAIL により、Lambda ユーザー定義関数 (LUDF) のオブザーバビリティを改善しました。このテーブルには、クエリで使用される LUDF の情報とメトリクスが記録されます。
- Amazon Redshift マテリアライズドビュー (MV) の自動 REFRESH クエリは、バックグラウンドの自律プロセスとしてではなく、ユーザークエリとして扱われるようになりました。自動 REFRESH クエリは、他のユーザークエリと同じ優先度で実行され、自律バックグラウンドプロセスによって延期されなくなりました。MV の自動 REFRESH の動作変更機能は、Amazon Redshift でプロビジョニングされたクラスターにおいて、パッチリリース P198 以降の CURRENT トラックでのみ有効になります。Serverless では現在無効になっています。

Amazon Redshift パッチ 197

このパッチのクラスターバージョン:

- 1.0.237947 - 後続トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2026 年 3 月 10 日にリリース
- 1.0.198462 - 後続トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2026 年 1 月 28 日にリリース
- 1.0.198462 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2026 年 1 月 21 日にリリース
- 1.0.194394 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2026 年 1 月 14 日にリリース

- 1.0.187928 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2026 年 1 月 9 日にリリース
- 1.0.179517 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 12 月 16 日にリリース
- 1.0.176066 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 12 月 9 日にリリース
- 1.0.166219 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 11 月 21 日にリリース

このパッチの新機能と改善点

- メッシュ内のクラスターのグローバル自律管理である Autonomics は、メッシュ全体のワークロードを考慮して、Vacuum、Analyze、ATO を最適化するようになりました。
- 複数の Amazon Redshift データウェアハウスからマテリアライズドビューを作成および更新します。
- SUPER データ型内に最大 16,000,000 バイトの文字列リテラルを保存するサポートを追加
- SUPER オブジェクトに含まれる属性の総数を返す GET_NUMBER_ATTRIBUTES(super_object) 関数を追加しました。
- テーブル復元オペレーションには、リリースパッチの互換性の制限が適用されます。テーブルの復元を試みる場合、使用できるバックアップは、バックアップの作成時と同じパッチバージョン、バックアップより 1 つ前のパッチバージョン、またはバックアップより上位のパッチバージョンを実行しているクラスターでのみとなります。
- CHAR データ型を使用する場合、パターンで末尾の空白を適切に処理するように LIKE 演算子を強化しました。
- セル階層で動作する 4 つの新しい空間 H3 関数が導入されました。H3_Resolution、H3_ToParent、H3_ToChildren、H3_IsValid です。
- Amazon Redshift フェデレーティッドアクセス許可を有効にすると、データアクセス許可を 1 回定義し、AWS アカウント内のすべてのウェアハウスに自動的に適用できるため、複数の Redshift データウェアハウスのアクセス許可管理を簡素化できます。
- Amazon Redshift フェデレーティッドアクセス許可カタログオブジェクトに対して、オブジェクトレベルの許可、きめ細かな列レベルのアクセスコントロール、スコープ付きアクセス許可の付与、データベースレベルの許可を有効にします。
- Amazon Redshift フェデレーティッドアクセス許可カタログで RLS ポリシーの CREATE、ALTER、ATTACH、DETACH、DROP 操作を有効にします。

- Amazon Redshift フェデレーティッドアクセス許可カタログのリレーションの行レベルセキュリティを有効または無効にする機能を有効にします。
- Amazon Redshift フェデレーティッドアクセス許可カタログオブジェクトで、CREATE、ALTER、ATTACH、DETACH、DROP の各マスキングポリシー操作を有効にします。
- SHOW POLICIES を有効にして、接続されたデータベースに RLS ポリシーとマスキングポリシーを表示し、Amazon Redshift フェデレーティッドアクセス許可カタログオブジェクトのリレーションに添付ファイルを表示します。
- データベースユーザーが Amazon Redshift フェデレーティッドアクセス許可カタログオブジェクトでクエリする IAM ロールを引き受けることを可能にします。
- 同じ Redshift クラスター内のクロスデータベース許可の可視性をサポートするように SHOW GRANTS コマンドを強化しました。
- 列レベルのアクセスコントロールの検出を簡素化するために、SHOW COLUMN GRANTS コマンドのサポートを追加しました。
- ストアドプロシージャメタデータの検出を簡素化する SHOW PROCEDURES コマンドのサポートを追加しました。
- ユーザー定義の関数メタデータの検出を簡素化する SHOW FUNCTIONS コマンドのサポートが追加されました。
- 関数とストアドプロシージャのパラメータメタデータを表示する SHOW PARAMETERS コマンドが導入されました。
- 関数とストアドプロシージャの制約メタデータを表示する SHOW CONSTRAINTS コマンドが導入されました。
- Amazon Redshift フェデレーティッドアクセス許可カタログおよび同じ Redshift クラスター内のクロスデータベースでの CREATE/CREATE OR REPLACE/DROP VIEW オペレーションのサポートが追加されました。
- 一部の INTERSECT または EXCEPT クエリでの定数文字列照合処理が改善されました。
- SHOW TABLES が拡張され、FAS 認証情報が AWS Glue Data Catalog アクセスにない場合に、失敗するのではなく警告メッセージが表示されます。
- ソートキー、分散スタイル、エンコーディング、照合情報のメタデータ列を追加した SHOW COLUMNS コマンドを強化しました。
- grantor_name 列を SHOW GRANTS 出力に追加し、すべての SHOW GRANTS 結果セットで database_name が含まれるよう標準化しました。

- table_owner、last_altered_time、last_modified_time、dist_style、table_sub_type 列を追加して SHOW TABLES コマンドを強化しました。
- アイデンティティセンター認証のサポート: Amazon Redshift は、新しい `GetIdentityCenterAuthToken` API オペレーションによる IAM アイデンティティセンター認証をサポートするようになりました。この機能を使用すると、既存のアイデンティティセンター ID を使用した Amazon Redshift クラスターとサーバーレスワークグループへのフェデレーションアクセスが可能になります。主なベネフィットには、アイデンティティセンターの認証情報を使用したシングルサインオンアクセス、暗号化された認証トークンによるセキュリティの強化、コンプライアンスのための包括的な監査ログ記録、既存のアイデンティティセンターのセットアップとのシームレスな統合などがあります。API オペレーションは、特定のクラスター識別子またはワークグループ名を対象とする認証トークンを生成します。要件には、アイデンティティセンターのアイデンティティ情報が組み込まれたアイデンティティ拡張認証情報と、API アクセスに適切な IAM アクセス許可が含まれます。

Amazon Redshift パッチ 196

このパッチのクラスターバージョン:

- 1.0.179872 - 後続トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2026 年 1 月 8 日にリリース
- 1.0.163480 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 11 月 20 日にリリース
- 1.0.160807 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 11 月 14 日にリリース
- 1.0.155905 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 11 月 11 日にリリース

このパッチの新機能と改善点

- 相関 IN サブクエリと EXISTS サブクエリのプランを解除して半結合にすることで最適化しました。
- SHOW TABLE コマンドを拡張して、すべての列の照合情報を含めました。

Amazon Redshift パッチ 195

このパッチのクラスターバージョン:

- 1.0.172541 - 後続トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 12 月 8 日にリリース
- 1.0.162991 - 後続トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 11 月 20 日にリリース
- 1.0.151179 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 11 月 4 日にリリース
- 1.0.148180 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 10 月 31 日にリリース
- 1.0.142459 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 10 月 24 日にリリース

このパッチの新機能と改善点

- CREATE TABLE 、CREATE TABLE AS SELECT、INSERT INTO、SQL コマンドを使用した Iceberg テーブルへの書き込みのサポートが追加されました。
- 大文字と小文字を区別しない照合順序のデータベースで SUPER データ型のサポートを追加しました。
- IANA タイムゾーンデータベースのバージョンを 2025b に更新しました。
- TZDB_VERSION 関数のサポートを追加しました。この関数は、使用中の現在の IANA タイムゾーンデータベースのバージョンを表示します。
- Amazon Redshift でパッチ互換性の制限が適用されるようになりました。現在のクラスターより 2 つ以上前のパッチバージョンで作成されたバックアップは、テーブル復元オペレーションを実行できなくなります。
- UNNEST 機能を拡張して JOIN 式をサポートします。以前は、UNNEST は単一テーブル参照に限定されていました。
- JIT (Just In Time) ANALYZE 機能のサポートが追加されました。JIT ANALYZE は、Amazon Redshift のクエリオプティマイザが効率的なクエリ実行プランを生成するために必要な最小限の統計セットを自動的、インテリジェントかつ効率的に収集し、Apache Iceberg out-of-box (OOB) クエリとワークロードに最適な価格パフォーマンスを実現できるようにします。

Amazon Redshift パッチ 194

このパッチのクラスターバージョン:

- 1.0.148169 - 後続トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 11 月 5 日にリリース
- 1.0.138443 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 10 月 18 日にリリース
- 1.0.129791 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 10 月 8 日にリリース

このパッチの新機能と改善点

- Amazon Redshift は、`AuthorizedColumns` フィールドにバックスラッシュが含まれる AWS Lake Formation が管理するリレーションにアクセスするクエリに失敗しなくなりました。
- SUPER 列の COPY コマンドで NULL AS オプションのサポートが追加され、NULL 値を表すカスタム文字列を指定できるようになりました。
- 日付とタイムスタンプとともに使用する場合、EXTRACT 関数と DATE_PART 関数で ISOWEEK と ISOYEAR の日付部分のサポートが追加されました。
- グループ化された変数に対する PartiQL ナビゲーションを修正し、GROUP BY クエリのネストされたフィールドアクセスの解析と評価が正しく行われるようにしました。
- SHOW CONSTRAINTS コマンドを使用してプライマリキーと外部キーの制約を表示するサポートが追加されました。
- データ共有マテリアライズドビューから作成されたローカルマテリアライズドビューのサポートが追加されました。
- さまざまなテーブルからデータを取得し、結合条件で Lambda ユーザー定義関数を呼び出すクエリのパフォーマンスが向上しました。
- データ共有 MVでの MVのサポートを追加
- DROP DATABASE コマンドは、バックグラウンドワーカーを使用してコミット後にリーダーノードカタログディレクトリを削除するようになりました。ワーカーを使用すると、カタログディレクトリのクリーンアップに関連するまれなバグを回避できます。

Amazon Redshift パッチ 193

このパッチのクラスターバージョン:

- 1.0.136890 - 後続トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 10 月 17 日にリリース
- 1.0.127211 - 最新トラック Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 9 月 30 日にリリース

このパッチの新機能と改善点

- CREATE TABLE LIKE を使用してテーブルを作成し、リモートデータベースから (同じクラスター内および異なるクラスター間で) 1 つの親テーブルのクローンを作成するサポートが追加されました。
- 最大 1 MB のジオメトリをサポートする、GeoJSON 形式の SUPER データ型からジオメトリを作成する ST_GeomFromGeoJSON オーバーロードが追加されました。
- 自動ワークロード管理 (WLM) の設定ミスが発生した場合のデフォルトのフォールバックオプションを変更しました。デフォルトのフォールバックオプションは、手動 WLM ではなく自動 WLM になりました。
- 最新の IANA タイムゾーンデータベースパッチを採用してタイムゾーン計算を更新しました。この変更により、特定のタイムゾーンと期間における日付と時刻の変換の仕組みが変更されます。詳細については、「Amazon Redshift は最新の IANA タイムゾーンデータベースを使用します」を参照してください。
- 多次元データレイアウト (MDDL) ソートのサポートが追加されました。MDDL ソートキーは、実際のクエリフィルターに基づいてデータを動的にソートし、クエリのパフォーマンスを加速します。詳細については、「Amazon Redshift データベースデベロッパーガイド」の「多次元データレイアウトのソート」を参照してください。

Amazon Redshift パッチ 192

このパッチのクラスターバージョン:

- 1.0.124422 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 9 月 22 日にリリース

- 1.0.122914 - 後続の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 9 月 16 日にリリース
- 1.0.121035 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 8 月 22 日にリリース
- 1.0.119650 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 8 月 7 日にリリース
- 1.0.118447 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 7 月 24 日にリリース

このパッチの新機能と改善点

- 照合された大文字小文字を区別しない列と定数文字列列の JOIN で不一致のハッシュ関数が選択される問題を修正しました。

Amazon Redshift パッチ 191

このパッチのクラスターバージョン:

- 1.0.117891 - 後続の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 7 月 29 日にリリース
- 1.0.117891 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 7 月 22 日にリリース
- 1.0.117458 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 7 月 15 日にリリース
- 1.0.116415 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 6 月 26 日にリリース
- 1.0.116263 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 6 月 25 日にリリース
- 1.0.115865 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 6 月 18 日にリリース

このパッチの新機能と改善点

- SELECT ステートメントの FROM 句に UNNEST オペレーションのサポートを追加し、お客様が配列要素を個々の行に変換できるようにしました。
- 共通テーブル式 (CTE) と再帰的な CTE で SUPER データ型を使用する場合、特に同じ名前のテーブルエイリアスを解決する場合に、PartiQL ネスト解除構文のエラー処理を改善しました。
- 大きなテーブルの増分自動分析のサポートを追加することで、自動分析のパフォーマンスを改善しました。
- Amazon Redshift は、ネストされたマテリアライズドビュー (MV) のカスケード更新をサポートするようになりました。
- Amazon Redshift は、Apache Iceberg テーブルで定義されたマテリアライズドビュー (MV) の自動更新をサポートするようになりました。
- Amazon RDS for Oracle と Amazon Redshift のゼロ ETL 統合のサポートを追加しました。
- Amazon RDS for PostgreSQL と Amazon Redshift のゼロ ETL 統合のサポートを追加しました。

Amazon Redshift パッチ 190

このパッチのクラスターバージョン:

- 1.0.115140 - 後続の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 6 月 16 日にリリース
- 1.0.115564 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 6 月 14 日にリリース
- 1.0.113436 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 5 月 15 日にリリース
- 1.0.112895 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 5 月 7 日にリリース

このパッチの新機能と改善点

- Amazon Redshift Serverless の AI 駆動型スケーリングおよび最適化のための改善モデルを追加しました。
- 拡張された VPC ルーティングを有効にして Amazon Redshift を使用する場合の自動コピーのサポートを追加しました。

- 複数の自動テーブル最適化オペレーションを異なるテーブル間で同時に実行するためのサポートを追加しました。
- ストリーミング取り込みにおいて、キューに入れた自動更新クエリによって、同じくキューに入れた UNDO クエリがトリガーされ、ロックの競合が発生するというまれな状況を修正しました。
- Oracle Database@AWS と Amazon Redshift のゼロ ETL 統合のサポートを追加しました。

Amazon Redshift パッチ 189

このパッチのクラスターバージョン:

- 1.0.112790 - 後続の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 5 月 14 日にリリース
- 1.0.112790 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 5 月 6 日にリリース
- 1.0.112086 – 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 4 月 29 日にリリース
- 1.0.111040 – 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 4 月 16 日にリリース
- 1.0.109768 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 3 月 22 日にリリース
- 1.0.109284 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 3 月 13 日にリリース
- 1.0.108971 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 3 月 7 日にリリース

このパッチの新機能と改善点

- SUPER、GEOMETRY、および GEOGRAPHY データ型を使用した書き込みクエリの同時実行スケーリングのサポートが追加されました。
- CREATE TABLE AS を使用したテーブルの作成中に、SUPER、GEOMETRY、または GEOGRAPHY の各列を DISTKEY 列または SORTKEY 列として設定できる問題を修正しました。
- TRY_CAST 関数のサポートを追加しました。
- 異なるテーブル間での複数のバキュームコマンドの同時実行が可能になります。

Amazon Redshift パッチ 188

このパッチのクラスターバージョン:

- 1.0.109616 – 後続の Amazon Redshift Serverless ワークグループバージョン – 2025 年 3 月 27 日にリリース
- 1.0.109616 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 3 月 24 日にリリース
- 1.0.108470 – 後続の Amazon Redshift Serverless ワークグループバージョン – 2025 年 3 月 13 日にリリース
- 1.0.108470 – 現在の Amazon Redshift Serverless ワークグループバージョン – 2025 年 3 月 11 日にリリース
- 1.0.108950 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 3 月 10 日にリリース
- 1.0.108790 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 3 月 9 日にリリース
- 1.0.108470 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 3 月 4 日にリリース
- 1.0.107910 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 2 月 20 日にリリース
- 1.0.107360 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 2 月 13 日にリリース
- 1.0.106767 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 2 月 5 日にリリース

このパッチの新機能と改善点

- Amazon S3 Tables カタログの自動マウントのサポートが追加され、Amazon S3 Tables で管理される Apache Iceberg テーブルに対してクエリを簡単に実行できるようになりました。必要なアクセス許可については、「AWS Lake Formation デベロッパーガイド」の「Prerequisites for managing Amazon Redshift namespaces in the AWS Glue Data Catalog」を参照してください。
- データマスキングチェックをバイパスして、動的データマスキング (DDM) がアタッチされたテーブルをデータ共有に追加できるようになりました。

- クエリブロックをスキップする相関参照 (「スキップレベル相関参照」とも呼ばれる) の使用が禁止されました。詳細については、Amazon Redshift データベースデベロッパーガイドの「サポートされていない相関サブクエリパターン」を参照してください。
- データ共有および 3 部構成の表記を使用した相関 SUPER データ型クエリのサポートを追加しました。
- EXCLUDE SQL キーワードのサポートを追加しました。
- 新しい GROUP BY ALL SQL キーワードのサポートを追加しました。
- ロールで設定されたワークロード管理 (WLM) キューのパフォーマンスが向上し、ロックの競合が減少しました。
- 履歴モードがオンになっているゼロ ETL テーブルでマテリアライズドビューの自動更新のサポートを追加しました。
- ゼロ ETL レプリケーション中に文字列に埋め込まれた null 文字のサポートを追加しました。
- AI 主導のスケーリングと最適化のモデルが改良されました。
- `sys_query_history` ビューに `username` 列を追加しました。

Amazon Redshift パッチ 187

このパッチのクラスターバージョン:

- 1.0.107351 - Amazon Redshift でプロビジョニングされたクラスターの最終トラックバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 2 月 13 日にリリース
- 1.0.106452 - Amazon Redshift でプロビジョニングされたクラスターの最終トラックバージョン - 2025 年 2 月 5 日にリリース
- 1.0.106980 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 2 月 3 日にリリース
- 1.0.106452 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 1 月 24 日にリリース
- 1.0.106073 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 1 月 21 日にリリース
- 1.0.105722 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 1 月 10 日にリリース
- 1.0.105373 - 現在の Amazon Redshift でプロビジョニングされたクラスターバージョンおよび Amazon Redshift Serverless ワークグループバージョン - 2025 年 1 月 8 日にリリース

- 1.0.104930 - Amazon Redshift Serverless ワークグループバージョン - 2024 年 12 月 20 日にリリース

このパッチの新機能と改善点

- 2 つの新しい空間 H3 関数 (H3_center および H3_boundary) が導入されました。
- 同時実行スケーリングクラスターで動的ディスクキャッシュを有効にします。
- ソートされたテーブルの取り込み中のメモリ消費が非効率になり、パフォーマンスに支障をきたす問題を修正しました。
- SELECT ARRAY_FLATTEN((SELECT FROM ...)) など、SELECT 句内にサブクエリがある特定のクエリが、単一の SUPER 値を含む特定のシナリオで size >= min_partiql_size XCHECK エラーを誤ってトリガーする問題を修正しました。
- null 不可の SUPER 式 (json_serialize()、json_size()、json_typeof()is_object() など) を CASE ... END や COALESCE() などの引数式と組み合わせると、誤った結果が生成されることがある問題を修正しました。
- 遅延バインディングビューで行レベルのセキュリティを有効にした後、そのビューからの SELECT で ERROR が発生する問題を修正しました。
- ゼロ ETL 統合に関する CDC でリーダーノードの CPU 使用率が高くなる問題を修正しました。
- 更新時を含むすべての状態のテーブルをクエリする機能が、ゼロ ETL 統合に追加されました。
- 大きなテキスト/文字列を切り捨てて、Amazon Redshift における最大 VARCHAR サイズに対応させる機能が、ゼロ ETL 統合に追加されました。
- 無効な UTF-8 文字を任意の指定された文字に置き換える機能が、ゼロ ETL 統合に追加されました。
- Amazon RDS for MySQL とのゼロ ETL 統合の「更新間隔」に関する問題を修正しました。
- Aurora PostgreSQL とのゼロ ETL 統合のため、JSON/JSONB データ型が新たにサポートされるようになりました。
- IS [NOT] {TRUE|FALSE|UNKNOWN} を含む式があるクエリでアサーションエラーが発生する問題を修正しました。
- 存在しない配列要素または属性、JSON の 'null' 値、または空の JSON 文字列にアクセスする場合など、json_extract_path_text() 関数や json_extract_array_element_text() 関数で NULL の代わりに空の文字列 '' が生成されたり、その逆になったりする問題を修正しました。

- 同じテーブルに書き込む同時トランザクションでの INSERT/COPY ステートメントについて、ロックを共有し、テーブルへの書き込みが必要になるまで処理を進められるようにすることで、パフォーマンスを向上させました。

Amazon Redshift パッチ 186

このパッチのクラスターバージョン:

- 1.0.82096 – 前のトラックバージョン – 2025 年 1 月 10 日にリリース
- 1.0.82000 – Amazon Redshift Serverless バージョン – 2025 年 1 月 10 日にリリース
- 1.0.81981 – 最新トラックバージョン – 2025 年 1 月 10 日にリリース
- 1.0.81475 – 前のトラックバージョン – 2025 年 1 月 6 日にリリース
- 1.0.81473 – Amazon Redshift Serverless バージョン – 2025 年 1 月 6 日にリリース
- 1.0.81462 – 最新トラックバージョン – 2025 年 1 月 6 日にリリース
- 1.0.80643 - トレーリングトラックバージョン – 2024 年 12 月 17 日にリリース
- 1.0.80583 - Amazon Redshift Serverless バージョン – 2024 年 12 月 17 日にリリース
- 1.0.80560 - 最新トラックバージョン - 2024 年 12 月 17 日にリリース
- 1.0.80498 - Amazon Redshift Serverless バージョン – 2024 年 12 月 13 日にリリース
- 1.0.80491 - 最新トラックバージョン - 2024 年 12 月 13 日にリリース
- 1.0.80036 - Amazon Redshift Serverless バージョン – 2024 年 12 月 6 日にリリース
- 1.0.80009 - 最新トラックバージョン - 2024 年 12 月 6 日にリリース
- 1.0.79372 - Amazon Redshift Serverless バージョン – 2024 年 11 月 26 日にリリース
- 1.0.79237 - Amazon Redshift Serverless バージョン – 2024 年 11 月 24 日にリリース
- 1.0.79229 - 最新トラックバージョン - 2024 年 11 月 24 日にリリース
- 1.0.79003 - Amazon Redshift Serverless バージョン – 2024 年 11 月 19 日にリリース
- 1.0.78987 - 最新トラックバージョン - 2024 年 11 月 19 日にリリース
- 1.0.78890 - Amazon Redshift Serverless バージョン – 2024 年 11 月 18 日にリリース
- 1.0.78881 - 最新トラックバージョン - 2024 年 11 月 18 日にリリース
- 1.0.78646 - Amazon Redshift Serverless バージョン – 2024 年 11 月 14 日にリリース
- 1.0.78641 - 最新トラックバージョン - 2024 年 11 月 14 日にリリース
- 1.0.78178 - Amazon Redshift Serverless バージョン – 2024 年 11 月 12 日にリリース

- 1.0.78160 - 最新トラックバージョン - 2024 年 11 月 12 日にリリース
- 1.0.77809 - Amazon Redshift Serverless バージョン - 2024 年 10 月 31 日にリリース
- 1.0.77777 - 最新トラックバージョン - 2024 年 10 月 31 日にリリース
- 1.0.77292 - Amazon Redshift Serverless バージョン - 2024 年 10 月 24 日にリリース
- 1.0.77272 - 現在のトラックバージョン - 2024 年 10 月 24 日にリリース
- 1.0.77040 - Amazon Redshift Serverless バージョン - 2024 年 10 月 22 日にリリース
- 1.0.77028 - 現在のトラックバージョン - 2024 年 10 月 22 日にリリース

このパッチの新機能と改善点

- Amazon Aurora MySQL、Amazon Aurora PostgreSQL、Amazon RDS for MySQL、および Amazon DynamoDB とのゼロ ETL 統合からのテーブルのマテリアライズドビューの自動更新と増分更新のサポートが追加されました。
- Amazon Redshift が `SELECT … WHERE key = (SELECT … correlated subquery)` などの相関する単一行サブクエリでクエリを書き換える方法を改善しました。このようなクエリは、サブクエリが 1 行を生成する場合にのみ有効であることに注意してください。書き換えが改善されたことにより、この条件に違反する特定のクエリは、以前は許可されていた場合でも、エラー: 単一行のサブクエリが 1 行以上を返すというエラーで失敗する可能性があります。これを回避するには、MIN() または MAX() による集計を追加するなどして、このようなサブクエリを 1 行返すように修正する必要がある場合があります。
- Amazon Redshift に SQL 識別子「KAFKA」を追加し、直接ストリーミング取り込みによる外部 Kafka ソースから Amazon Redshift へのストリーミングに対応しました。Confluent Managed Cloud や Apache Kafka などの外部 Kafka ソースが該当します。
- データ共有を使用したマルチウェアハウス書き込みに対応しました。これにより、ワークロードのニーズに応じてタイプとサイズの異なるさまざまなウェアハウスを使用して、書き込みワークロードをスケールし、抽出、変換、ロード (ETL) ワークロードのパフォーマンスを向上させることができます。
- Amazon Redshift を使用して、Amazon S3 Tables で管理される Apache Iceberg テーブルで `SELECT` クエリを実行できるようになります。詳細については、「Amazon Simple Storage Service ユーザーガイド」の「Amazon Redshift を使用した Amazon S3 テーブルへのアクセス」を参照してください。
- Amazon Redshift でクロスデータベースの書き込みクエリに対応しました。Amazon Redshift クラスター内の複数のデータベースにまたがって書き込めるようになります。詳細については、「データベース間でのデータのクエリ」を参照してください。

Amazon Redshift パッチ 185

このパッチのクラスターバージョン:

- 1.0.79878 - トレーリングトラックバージョン – 2024 年 12 月 3 日にリリース
- 1.0.79868 - Amazon Redshift Serverless バージョン – 2024 年 12 月 3 日にリリース
- 1.0.79845 - 最新トラックバージョン - 2024 年 12 月 3 日にリリース
- 1.0.78946 - トレーリングトラックバージョン - 2024 年 11 月 19 日にリリース
- 1.0.78354 - トレーリングトラックバージョン - 2024 年 11 月 12 日にリリース
- 1.0.78130 - Amazon Redshift Serverless バージョン – 2024 年 11 月 12 日にリリース
- 1.0.78125 - 最新トラックバージョン - 2024 年 11 月 12 日にリリース
- 1.0.78016 - Amazon Redshift Serverless バージョン – 2024 年 11 月 5 日にリリース
- 1.0.78014 - 最新トラックバージョン - 2024 年 11 月 5 日にリリース
- 1.0.77707 - Amazon Redshift Serverless バージョン – 2024 年 11 月 4 日にリリース
- 1.0.77687 - 最新トラックバージョン - 2024 年 11 月 4 日にリリース
- 1.0.77467 - Amazon Redshift Serverless バージョン – 2024 年 11 月 4 日にリリース
- 1.0.77433 - 最新トラックバージョン - 2024 年 11 月 4 日にリリース
- 1.0.77467 - Amazon Redshift Serverless バージョン - 2024 年 10 月 29 日にリリース
- 1.0.77433 - 最新トラックバージョン - 2024 年 10 月 29 日にリリース
- 1.0.76991 - Amazon Redshift Serverless バージョン - 2024 年 10 月 21 日にリリース
- 1.0.76913 - 現在のトラックバージョン - 2024 年 10 月 21 日にリリース
- 1.0.76645 - Amazon Redshift Serverless バージョン - 2024 年 10 月 14 日にリリース
- 1.0.76642 - 現在のトラックバージョン - 2024 年 10 月 14 日にリリース
- 1.0.76242 - Amazon Redshift Serverless バージョン – 2024 年 10 月 10 日にリリース
- 1.0.76230 - 現在のトラックバージョン – 2024 年 10 月 10 日にリリース

このパッチの新機能と改善点

- データレイクテーブルに作成されたマテリアライズドビュー (MV) の増分更新のサポートが追加されました。
- マテリアライズドビューの分散キーとソートキーを変更するサポートを導入しました。

- Amazon Redshift の機械学習 (ML) と Amazon Bedrock の統合のサポートを追加し、シンプルな SQL コマンドからの大規模言語モデル (LLM) と Amazon Redshift のデータを活用します。
- エラーをスローせずに Amazon Redshift 空間データ内の空のシェープファイルの処理を許可するバグを修正しました。
- ゼロ ETL 統合で、varbit および varbinary データ型が空の文字列値を NULL ではなく "" として取り込むことを許可するバグを修正しました。
- ゼロ ETL 統合において、クロスデータベースクエリが古い結果を返す原因となっていた、結果キャッシュを持つ競合状態を修正しました。
- ゼロ ETL 統合の再同期プロセスを最適化したことで、再同期時間が短縮されました。
- 復旧およびシステムメンテナンスオペレーション後のゼロ ETL ブートストラップ時間が改善されました。これで、システムは最新の利用可能な CDC ポイントまで回復する代わりに、最後のクエリオペレーション後に CDC を回復できるようになりました。
- 新しいシステムテーブル `sys_integration_table_activity` とのゼロ ETL 統合のオブザーバビリティが改善されました。このテーブルは、ゼロ ETL 統合テーブルの挿入、更新、削除を追跡します。
- フェデレーションロールの自動作成が改善されました。Amazon Redshift 管理者は、フェデレーションユーザーのログイン中のフェデレーションロールの自動作成をより細かく制御できるようになりました。これで、各 ID プロバイダーについて自動作成設定の有効 / 無効化、フィルターの適用、設定ができるようになりました。
- IAM アイデンティティセンターのユーザーは、S3 アクセス許可を使用し、IAM アイデンティティセンターの認証情報を使用して COPY、UNLOAD、CREATE LIBRARY を実行できます。
- 自動コピー (COPY JOB) の一般提供を開始しました。この機能を使用して、Amazon S3 からファイルを継続的に取り込むことができます。

Amazon Redshift パッチ 184

このパッチのクラスターバージョン:

- 1.0.77706 - トレーリングトラックバージョン - 2024 年 11 月 12 日にリリース
- 1.0.76832 - 現在のトラックバージョン - 2024 年 10 月 17 日にリリース
- 1.0.76169 - Amazon Redshift Serverless バージョン – 2024 年 10 月 10 日にリリース
- 1.0.76142 - 現在のトラックバージョン – 2024 年 10 月 10 日にリリース
- 1.0.75677 - Amazon Redshift Serverless バージョン – 2024 年 9 月 27 日にリリース

- 1.0.75672 - 現在のトラックバージョン – 2024 年 9 月 27 日にリリース
- 1.0.75504 - Amazon Redshift Serverless バージョン – 2024 年 9 月 23 日にリリース
- 1.0.75449 - 現在のトラックバージョン – 2024 年 9 月 24 日にリリース
- 1.0.74765 - Amazon Redshift Serverless バージョン – 2024 年 9 月 12 日にリリース
- 1.0.74754 - 現在のトラックバージョン – 2024 年 9 月 12 日にリリース

このパッチの新機能と改善点

- ストリーミングデータインジェスト用の Amazon Redshift ストリーミングマテリアライズドビュー (MV) では、VARBYTE 列サイズが 1,024,000 バイトから増加しました。Amazon Redshift は Amazon Kinesis Data Streams から最大 1,048,576 バイト、または 1MiB のレコードを取り込むことができます。Redshift は、Amazon Managed Streaming for Apache Kafka から最大 16,777,216 バイト、または 16 MiB のレコードを取り込むことができます。ATA SQL コマンド `ALTER TABLE <target_tbl> APPEND FROM <streaming_mv>` を使用している場合は、<target_tbl> を削除して、対応するより大きな VARBYTE 列サイズになるように再作成してください。
- Amazon Redshift データ共有に、Lake Formation によって管理されるテーブルを含む AWS Glue Data Catalog を参照する Amazon S3 データレイクテーブルとビューを含めることができるようになりました。
- DynamoDB とのゼロ ETL 統合におけるテーブルのマテリアライズドビューの自動更新と増分更新のサポートが追加されました。
- Amazon Redshift のレプリケーション更新レートを指定するゼロ ETL 統合テーブルで「更新間隔」を設定するサポートが追加されました。新しい統合用のデータベースの作成時に設定することも、既存の統合のデータベースを変更することもできます。
- Amazon Managed Streaming for Apache Kafka の Amazon Redshift ストリーミング取り込みでの Mutual Transport Layer Security (mTLS) 認証のサポートが追加されました。
- クエリのテキスト表現とそのパラメータの値に基づく SQL クエリの一意の識別子であるクエリハッシュのサポートが追加されました。同様のクエリの識別、グループ化、分析に使用できます。クエリハッシュが SYS_QUERY_HISTORY ビューで検索できるようになりました。次の 2 つの新しい列が追加されました。
  - `user_query_hash` – クエリリテラルを含む、ユーザーが送信したハッシュ。
  - `generic_query_hash` - クエリリテラルを含まない、ユーザーが送信したハッシュ。
- ゼロ ETL が CDC レプリケーションとスキャンを実行し、結果キャッシュでテーブルクエリを実行していたまれなケースにおけるシステムのデッドロックを修正しました。

- Python ユーザー定義関数 (UDF) クエリのワークロード管理 (WLM) リソースが使用できなかった場合に、Python UDF クエリが他のクエリを優先する WLM の問題に対処しました。
- ワークロード管理 (WLM) が、新しく作成されたユーザーロールにマッピングされたキューにクエリをルーティングできない問題に対処しました。
- 大規模なプロデューサーテーブルをクエリする小規模なデータ共有コンシューマーのディスク使用率が向上しました。
- プロビジョニングされたクラスターがより大きなサイズに伸縮自在にサイズ変更した場合の INSERT ステートメントのパフォーマンスが向上しました。

Amazon Redshift パッチ 183

このパッチのクラスターバージョン:

- 1.0.75655 - 現在のトラックバージョン - 2024 年 9 月 30 日にリリース
- 1.0.75388 - Amazon Redshift Serverless バージョン – 2024 年 9 月 25 日にリリース
- 1.0.75379 - 現在のトラックバージョン – 2024 年 9 月 25 日にリリース
- 1.0.74967 - Amazon Redshift Serverless バージョン – 2024 年 9 月 17 日にリリース
- 1.0.74927 - 現在のトラックバージョン – 2024 年 9 月 17 日にリリース
- 1.0.74518 - Amazon Redshift Serverless バージョン – 2024 年 9 月 11 日にリリース
- 1.0.74503 - 現在のトラックバージョン – 2024 年 9 月 11 日にリリース
- 1.0.74223 - Amazon Redshift Serverless バージョン – 2024 年 9 月 5 日にリリース
- 1.0.74159 - 現在のトラックバージョン – 2024 年 9 月 5 日にリリース
- 1.0.74126 - Amazon Redshift Serverless バージョン – 2024 年 8 月 30 日にリリース
- 1.0.74097 - 現在のトラックバージョン – 2024 年 8 月 30 日にリリース
- 1.0.73016 - Amazon Redshift Serverless バージョン – 2024 年 8 月 8 日にリリース
- 1.0.72982 - 現在のトラックバージョン – 2024 年 8 月 8 日にリリース

このパッチの新機能と改善点

- SVV_DATABASE_PRIVILEGES および SVV_SCHEMA_PRIVILEGES を介したスコープ設定アクセス許可の検出のサポートを追加しました。また、SVV_DATABASE_PRIVILEGES および SVV_SCHEMA_PRIVILEGES に privilege_scope 列も導入しました。

- 列のグループ化に異なる値 (NDV) が少ない場合に異なる集計オペレーションを実行するクエリのパフォーマンスが向上しました。
- プロビジョニングされたデータウェアハウスの INSERT/COPY ステートメントが 2 倍以上のサイズで伸縮自在にサイズ変更されるようにパフォーマンスが向上しました。
- 動的データマスキングポリシー内のセッションコンテキスト変数をサポートしました。
- MERGE ステートメントのソースとしてサブクエリとビューのサポートを追加しました。
- プロビジョニングされた同時実行スケーリングとサーバーレス自動スケーリングコンピューティングに関する MERGE ステートメントを含むストアドプロシージャをサポートします。
- COPY コマンドのワークロード管理とサイズ変更を行うウェアハウスのリソース予測が改善され、クエリのパフォーマンスが向上しました。
- 使用可能なメモリが限られているクラスターのメモリ不足エラーに対する耐障害性を向上しました。
- COPY コマンドにフィールド区切り文字として ASCII 以外の文字のサポートを追加しました。
- COPY コマンドを使用して ISO-8859-1 文字セットにエンコードされたデータを取り込むためのサポートを追加しました。
- URI を指定する場合、MSK 外部スキーマ定義で CLUSTER_ARN を指定する要件を削除しました。
- ゼロ ETL 統合テーブルのスキャン中に範囲スキャンフィルターの適用をサポートします。
- ゼロ ETL 統合テーブルへのソートキーの追加をサポートします。
- ゼロ ETL 統合データベースの作成時に CREATE DATABASE ステートメントで指定される `serializable` や `collation` などのデータベースオプションをサポートします。
- ゼロ ETL 統合でデータフィルターが 2 KB を超えたときにクラスターが再起動する問題を修正しました。

Amazon Redshift パッチ 182

このパッチのクラスターバージョン:

- 1.0.73589 — 後続トラックバージョン — 2024 年 8 月 22 日にリリース
- 1.0.73359 - Amazon Redshift Serverless バージョン – 2024 年 8 月 15 日にリリース
- 1.0.73348 - 現在のトラックバージョン – 2024 年 8 月 15 日にリリース
- 1.0.72917 - Amazon Redshift Serverless バージョン – 2024 年 8 月 12 日にリリース

- 1.0.72899 - 現在のトラックバージョン – 2024 年 8 月 12 日にリリース
- 1.0.72528 - Amazon Redshift Serverless バージョン – 2024 年 8 月 7 日にリリース
- 1.0.72503 - 現在のトラックバージョン – 2024 年 8 月 8 日にリリース
- 1.0.72239 - Amazon Redshift Serverless バージョン – 2024 年 8 月 1 日にリリース
- 1.0.71714 - Amazon Redshift Serverless バージョン – 2024 年 7 月 24 日にリリース
- 1.0.71629 - 現在のトラックバージョン — 2024 年 7 月 24 日にリリース
- 1.0.70953 - Amazon Redshift Serverless バージョン – 2024 年 7 月 11 日にリリース
- 1.0.70890 - 現在のトラックバージョン — 2024 年 7 月 11 日にリリース
- 1.0.70716 - Amazon Redshift Serverless バージョン – 2024 年 7 月 8 日にリリース
- 1.0.70695 - 現在のトラックバージョン — 2024 年 7 月 8 日にリリース
- 1.0.69945 - Amazon Redshift Serverless バージョン – 2024 年 6 月 27 日にリリース
- 1.0.69938 - 現在のトラックバージョン — 2024 年 6 月 27 日にリリース

このパッチの新機能と改善点

- LISTAGG、MEDIAN、PERCENTILE_CONT、PERCENTILE_DISC を更新して、ユーザー定義テーブルを不要にしました。カタログテーブルを参照するクエリやテーブルを一切参照しないクエリでも、これらの関数を使用できます。
- 単一セッション内で複数のクエリにまたがる一時テーブルを統合して、同時実行性の高いワークロードを実現することで、データ共有の読み取りクエリのクエリ計画時間を短縮しました。
- Redshift 機械学習 (ML) と Amazon SageMaker AI Jumpstart の統合の一般提供により、独自の大規模言語モデルを使用できるようにしました。
- Redshift ML での SUPER 入出力データ型のサポートを導入しました。
- ターゲットテーブルが動的データマスキングポリシーによって保護され、JOIN 句で参照されている場合、JOIN 句を含む UPDATE ステートメントのサポートを有効にしました。
- 統合がソースから削除された後でも、Redshift でのゼロ ETL データベースのクエリができるようにしました。
- ゼロ ETL 統合が失敗する原因となるレプリケーションエラーを修正しました。これにより、統合の耐障害性が向上します。
- ゼロ ETL 統合を作成したユーザー以外のユーザーが、GRANT アクセス許可の後にデータをクエリすることを許可しました。

- ゼロ ETL 統合が有効になっている、Amazon Redshift でプロビジョニングされたクラスターでクラスターが再起動する場合があるメモリ不足の問題を修正しました。
- ソース RDS マルチ AZ DB クラスターから、RDS for MySQL と Redshift のゼロ ETL 統合を作成できるようにしました。マルチ AZ DB クラスターは、2 つの読み取り可能なレプリカ DB インスタンスを持つ、Amazon RDS の準同期および高可用性のデプロイモードです。
- Amazon Redshift ストリーミングマテリアライズドビューを Amazon MSK トピックに関連付けるために必要な外部スキーマ定義で Amazon MSK クラスターのブローカー URI を指定することで、ユーザーが Amazon Redshift ストリーミングコンシューマークライアントから Amazon MSK クラスターに接続できるようにしました。この機能により、インターネットゲートウェイを介して Amazon MSK クラスターで GetBootStrapBroker API を呼び出し、Amazon MSK ブートストラップブローカーノード名を取得する必要がなくなります。
- Amazon Redshift Serverless インスタンスを再開し、IAM アイデンティティセンターの認証方法を使用して Amazon Redshift クエリエディタ v2 のデータベースに接続するときに、既存のデータベースユーザーが Serverless インスタンスを再開できるようになる問題を解決しました。
- テーブルベースのシャーディングに移行することで、CDC レプリケーションを最適化し、Redshift コンピューティングのリソース使用率を削減しました。
- ワークロード管理 (WLM) のリソース予測の強化により、クエリのパフォーマンスが向上しました。
- 同時実行スケーリングクラスターに対するクエリが失敗して「ran out of WLM queues for restart」というメッセージが表示される問題を修正しました。
- ユーザーが無効なワークロード管理 (WLM) 設定を適用しようとしたときに Amazon Redshift が手動 WLM にフォールバックするという WLM 問題を修正しました。
- 定期メンテナンスや予期しない停止のためにプロデューサーが停止した場合でも、データ共有コンシューマーが読み取りクエリを実行できるようにしました。
- COUNT (DISTINCT) 集計関数など、データを集約するクエリで ANY_VALUE 関数が使用されている場合に発生するまれなクラスター再起動の問題を修正しました。

Amazon Redshift パッチ 181

このパッチのクラスターバージョン:

- 1.0.72031 - 現在のトラックバージョン – 2024 年 8 月 1 日にリリース
- 1.0.71912 — 後続トラックバージョン — 2024 年 8 月 1 日にリリース
- 1.0.70665 - Amazon Redshift Serverless バージョン – 2024 年 7 月 8 日にリリース

- 1.0.70634 - 現在のトラックバージョン — 2024 年 7 月 8 日にリリース
- 1.0.69954 - Amazon Redshift Serverless バージョン – 2024 年 6 月 26 日にリリース
- 1.0.69952 - 現在のトラックバージョン — 2024 年 6 月 26 日にリリース
- 1.0.69497 - Amazon Redshift Serverless バージョン – 2024 年 6 月 18 日にリリース
- 1.0.69451 - 現在のトラックバージョン — 2024 年 6 月 18 日にリリース
- 1.0.69076 - Amazon Redshift Serverless バージョン – 2024 年 6 月 14 日にリリース
- 1.0.69065 - 現在のトラックバージョン — 2024 年 6 月 14 日にリリース
- 1.0.68555 - Amazon Redshift Serverless バージョン — 2024 年 5 月 31 日にリリース
- 1.0.68540 - 現在のトラックバージョン — 2024 年 5 月 31 日にリリース
- 1.0.68328 - Amazon Redshift Serverless バージョン — 2024 年 5 月 23 日にリリース
- 1.0.68205 - 現在のトラックバージョン — 2024 年 5 月 23 日にリリース
- 1.0.67796 - Amazon Redshift Serverless バージョン — 2024 年 5 月 15 日にリリース
- 1.0.67788 - 現在のトラックバージョン — 2024 年 5 月 15 日にリリース
- 1.0.67308 - Amazon Redshift Serverless バージョン — 2024 年 5 月 1 日にリリース
- 1.0.67305 - 現在のトラックバージョン — 2024 年 5 月 1 日にリリース

このパッチの新機能と改善点

- SUPER オブジェクト値の属性名の大文字と小文字を変更する「lower_attribute_names()」関数と「upper_attribute_names()」関数のサポートを導入しました。
- ID 列を使用する場合の CREATE TABLE LIKE の問題を修正しました。以前、新しいテーブルはソーステーブルから識別子を継承していました。これにより、ソーステーブルが後で削除された場合、新しいテーブルで識別子が無効になるため、問題が発生していました。
- 一部の外部テーブルが SVV_ALL_TABLES に表示されない問題を修正しました。
- クラスターのブートストラップ時間を短縮し、同時実行ワークロード数が多い場合のクエリの初期化を高速化しました。
- RDS および Aurora MySQL のフェデレーションソースに split_part() 関数を渡すときにエラーが発生するというフェデレーティッドクエリの問題を修正しました。
- プロビジョニングされた同時実行スケーリングクラスターとサーバーレス自動スケーリングコンピューティングで、ALTER TABLE...ALTER DISTSTYLE KEY DISTKEY コマンドを通じてユーザーが開始した分散キーの変更がサポートされるようになりました。

- プロビジョニングされた同時実行スケーリングクラスターとサーバーレス自動スケーリングコンピューティングにおける集計を伴うマテリアライズドビューの手動更新がサポートされるようになりました。
- 最大サイズが 16 MB のレコードを処理するゼロ ETL と、最大 16 MB の SUPER 値のサポートを追加しました。
- スキーマやテーブル名などの追加の詳細を提供することで、Aurora MySQL からのゼロ ETL の初回同期中のエラーメッセージを強化しました。
- Amazon Redshift ML CREATE MODEL によるタグ付けのサポートを導入しました。この改善により、Amazon Redshift ML で使用する Amazon SageMaker AI リソースにタグ付けできるようになりました。タグ付けは、リソースの管理、識別、整理、検索、フィルタリングに役立ちます。
- AWS Lambda でデータ処理を最適化することで、Lambda ユーザー定義関数 (UDF) を含むクエリのパフォーマンスを向上させました。
- 伸縮自在にサイズ変更されたクラスターとサーバーレスクラスターのソートされたテーブルにおけるデータインジェスト中のメモリ使用量を削減しました。
- SYS_QUERY_HISTORY ビューの `query_text` 列と SYS_QUERY_TEXT ビューの `text` 列の改行 (\n) のサポートを追加しました。

Amazon Redshift パッチ 180

このパッチのクラスターバージョン:

- 1.0.68520 — 後続トラックバージョン — 2024 年 5 月 28 日にリリース
- 1.0.67699 — 後続トラックバージョン — 2024 年 5 月 15 日にリリース
- 1.0.66960 — 後続トラックバージョン — 2024 年 4 月 21 日にリリース
- 1.0.66954 - 現在のトラックバージョン — 2024 年 4 月 21 日にリリース
- 1.0.66276 - 現在のトラックバージョン — 2024 年 4 月 12 日にリリース
- 1.0.66290 - Amazon Redshift Serverless バージョン — 2024 年 4 月 10 日にリリース
- 1.0.63590 — 現在のトラックバージョン — 2024 年 2 月 19 日にリリース
- 1.0.63567 — Amazon Redshift Serverless バージョン — 2024 年 2 月 16 日にリリース
- 1.0.63282 — Amazon Redshift Serverless バージョン — 2024 年 2 月 13 日にリリース
- 1.0.63269 — 現在のトラックバージョン — 2024 年 2 月 13 日にリリース
- 1.0.63215 — Amazon Redshift Serverless バージョン — 2024 年 2 月 12 日にリリース

- 1.0.63205 — 現在のトラックバージョン — 2024 年 2 月 12 日にリリース
- 1.0.63030 — Amazon Redshift Serverless バージョン — 2024 年 2 月 7 日にリリース
- 1.0.62913 — 現在のトラックバージョン — 2024 年 2 月 7 日にリリース
- 1.0.62922 — Amazon Redshift Serverless バージョン — 2024 年 2 月 5 日にリリース
- 1.0.62878 — 現在のトラックバージョン — 2024 年 2 月 5 日にリリース
- 1.0.62698 — Amazon Redshift Serverless バージョン — 2024 年 1 月 31 日にリリース
- 1.0.62614 — 現在のトラックバージョン — 2024 年 1 月 31 日にリリース
- 1.0.61687 — Amazon Redshift Serverless バージョン — 2024 年 1 月 5 日にリリース
- 1.0.61678 — 現在のトラックバージョン — 2024 年 1 月 5 日にリリース
- 1.0.61567 - Amazon Redshift Serverless バージョン – 2023 年 12 月 31 日にリリース
- 1.0.61559 - 現在のトラックバージョン — 2023 年 12 月 31 日にリリース
- 1.0.61430 - Amazon Redshift Serverless バージョン – 2023 年 12 月 29 日にリリース
- 1.0.61395 - 現在のトラックバージョン — 2023 年 12 月 29 日にリリース

このパッチの新機能と改善点

- CURRENT_USER が変更され、返されるユーザー名が 64 文字に切り捨てられなくなりました。
- 標準ビューと遅延バインディングビューにデータマスキングポリシーを適用する機能が追加されました。
- SUPER データ型の列のスカラー属性に動的データマスキング (DDM) を適用する機能が追加されました。
- OBJECT_TRANSFORM SQL 関数が追加されました。詳細については、「Amazon Redshift データベースデベロッパーガイド」の「OBJECT_TRANSFORM 関数」を参照してください。
- ネスト化されたデータに AWS Lake Formation のきめ細かいアクセスコントロールを適用し、Amazon Redshift データレイク分析によるクエリを実行できるようになりました。
- INTERVAL データ型が追加されました。
- CONTINUE_HANDLER が追加されました。ストアドプロシージャのフローを制御する例外ハンドラーの一種です。これを使用すると、既存のステートメントブロックを終了せずに例外をキャッチして処理できます。
- 個々のオブジェクトだけでなく、スコープ (スキーマまたはデータベース) にもアクセス許可を定義できるようになりました。そのおかげで、スコープ内の現在および将来のすべてのオブジェクトに対するアクセス許可をユーザーやロールに付与できます。

- データ共有からデータベースを作成する機能が追加され、コンシューマー側の管理者が共有データベースのオブジェクトに対する個別のアクセス許可をコンシューマー側のユーザーとロールに付与できるようになりました。
- リモート BYOM モデルからの SUPER 戻りデータ型がサポートされるようになりました。持ち込める SageMaker AI モデルの範囲が広がり、戻り値の形式がより複雑なモデルも対象となりました。
- 外部関数が変更され、小数部の有無にかかわらず、数値が列の数値データ型に暗黙的にキャストされるようになりました。int2、int4、int8 の列の場合、数値が範囲外でない限り、小数点以下を切り捨てて受け入れられます。float4 と float8 の列では、小数点以下の桁数を除いた数値が受け入れられます。
- H3 階層型地理空間インデックスグリッドシステムで使用できる 3 つの空間関数、H3_FromLongLat、H3_FromPoint、H3_Polyfill が追加されました。

Amazon Redshift パッチ 179

このパッチのクラスターバージョン:

- 1.0.62317 — Amazon Redshift Serverless バージョン — 2024 年 1 月 29 日にリリース
- 1.0.62312 — 現在のトラックバージョン — 2024 年 1 月 29 日にリリース
- 1.0.61631 — Amazon Redshift Serverless バージョン — 2024 年 1 月 5 日にリリース
- 1.0.61626 — 現在のトラックバージョン — 2024 年 1 月 5 日にリリース
- 1.0.61191 - 現在のトラックバージョン — 2023 年 12 月 16 日にリリース
- 1.0.61150 - Amazon Redshift Serverless バージョン – 2023 年 12 月 16 日にリリース
- 1.0.60982 - Amazon Redshift Serverless バージョン – 2023 年 12 月 13 日にリリース
- 1.0.60854 - 現在のトラックバージョン — 2023 年 12 月 10 日にリリース
- 1.0.60354 - Amazon Redshift Serverless バージョン – 2023 年 11 月 22 日にリリース
- 1.0.60353 - 現在のトラックバージョン – 2023 年 11 月 21 日にリリース
- 1.0.60293 - Amazon Redshift Serverless バージョン – 2023 年 11 月 21 日にリリース
- 1.0.60292 - 現在のトラックバージョン — 2023 年 11 月 22 日にリリース
- 1.0.60161 - Amazon Redshift Serverless バージョン – 2023 年 11 月 18 日にリリース
- 1.0.60140 - 現在のトラックバージョン — 2023 年 11 月 18 日にリリース
- 1.0.60139 - Amazon Redshift Serverless バージョン – 2023 年 11 月 18 日にリリース

- 1.0.59947 - Amazon Redshift Serverless バージョン – 2023 年 11 月 16 日にリリース
- 1.0.59945 - 現在のトラックバージョン — 2023 年 11 月 16 日にリリース
- 1.0.59118 - Amazon Redshift Serverless バージョン – 2023 年 11 月 9 日にリリース
- 1.0.59117 - 現在のトラックバージョン — 2023 年 11 月 9 日にリリース

このパッチの新機能と改善点

- 適切なアクセス許可を持つフェデレーティッドユーザーが、次のような行レベルのセキュリティビューと動的データマスキングシステムビューを表示できるようになりました。
  - SVV_ATTACHED_MASKING_POLICY
  - SVV_MASKING_POLICY
  - SVV_RLS_ATTACHED_POLICY
  - SVV_RLS_POLICY
  - SVV_RLS_RELATION
- FROM 句にスカラー関数のみを含むクエリがエラーになるような機能が追加されました。
- 永続的なターゲットテーブル機能がある CREATE TABLE AS (CTAS) ステートメントが同時実行スケーリングクラスターに追加されました。同時実行スケーリングクラスターがより多くのクエリをサポートするようになりました。
- RA3 クラスターで従来のサイズ変更を実行した後のテーブル再配布のステータスを追跡するために、次のシステムテーブルが追加されました。
  - SYS_RESTORE_STATE システムテーブルには、テーブルレベルの再配布の進行状況が表示されます。
  - SYS_RESTORE_LOG システムテーブルには、データ再配布のスループットの履歴が表示されます。
- RA3 ノードタイプで従来のサイズ変更を実行した後の EVEN テーブルでのスライススキューの最小化が改善されました。これは、従来のサイズ変更を実行したパッチ 178 クラスターにも適用されます。
- 同時実行スケーリングクラスターで、EXTENSION を指定した UNLOAD がサポートされるようになりました。
- ハッシュ結合と NestLoop 結合に Λ UDF が含まれるクエリのパフォーマンスが向上しました。
- RA3 ノードタイプの伸縮自在なサイズ変更のパフォーマンスが向上しました。
- データ共有のクエリのパフォーマンスが向上しました。

- 伸縮自在にサイズ変更され、プロビジョニングされたクラスターとサーバーレスワークグループで、手動で開始した分析クエリのパフォーマンスが向上しました。
- ワークロード管理におけるリソース予測が向上し、自動 WLM クエリのパフォーマンスが向上しました。
- 専有テナンシー VPC でクラスターを起動する機能が削除されました。この変更は VPC 内の EC2 インスタンスのテナンシーには影響しません。VPC のテナンシーを `modify-vpc-tenancy` AWS CLI コマンドでデフォルト設定に変更できます。
- マテリアライズドビューの手動更新が、プロビジョニング済みの同時実行スケーリングクラスターとサーバーレス自動スケーリングコンピューティングでサポートされるようになりました。
- EXTRACT 関数に INTERVAL リテラルのサポートが追加されました。例えば、50 時間は 2 日と 2 時間だと解釈され、2 の時間成分が抽出されるため、`EXTRACT('hours' from Interval '50 hours')` は 2 を返します。

Amazon Redshift パッチ 178

このパッチのクラスターバージョン:

- 1.0.63327 - 現在のトラックバージョン — 2024 年 2 月 9 日にリリース
- 1.0.63313 — 現在のトラックバージョン — 2024 年 2 月 9 日にリリース
- 1.0.60977 - 現在のトラックバージョン – 2023 年 12 月 15 日にリリース
- 1.0.59596 - 現在のトラックバージョン — 2023 年 11 月 9 日にリリース
- 1.0.58593 - Amazon Redshift Serverless バージョン – 2023 年 10 月 23 日にリリース
- 1.0.58558 - 現在のトラックバージョン — 2023 年 10 月 23 日にリリース
- 1.0.57864 - 現在のトラックバージョン – 2023 年 10 月 12 日にリリース
- 1.0.57850 - Amazon Redshift Serverless バージョン – 2023 年 10 月 12 日にリリース
- 1.0.56952 - 現在のトラックバージョン – 2023 年 9 月 25 日にリリース
- 1.0.56970 - Amazon Redshift Serverless バージョン –2023 年 9 月 25 日にリリース

このパッチの新機能と改善点

- Amazon Redshift は、プロデューサーインスタンスで同時にデータ変更が進行中に、コンシューマーインスタンスでのメタデータの更新を高速化することで、データ共有クエリのパフォーマンスを向上させました。

- マテリアライズドビューのベーステーブルが共有データを参照する場合に、Amazon Redshift データ共有コンシューマーインスタンスでのマテリアライズドビューの自動更新と増分更新のサポートを追加しました。
- SUPER データ型に最大 16 MB のサイズのラージオブジェクトを格納するためのサポートを追加しました。JSON、PARQUET、TEXT、CSV のソースファイルから取り込む場合、半構造化データまたはドキュメントを最大 16 MB の SUPER データ型の値として読み込むことができます。
- 単一ノードの Amazon Redshift RA3 クラスターとの間でスケーリングするための伸縮自在なサイズ変更のサポートを追加しました。
- シングルノードの Amazon Redshift RA3 クラスターでは、暗号化機能の強化の利点を活用し、全体的な暗号化時間を短縮して、暗号化プロセス中のデータウェアハウスの可用性を向上させることができるようになりました。
- SUPER データ型に保存されているデータをネスト解除したり、ピボット解除したりするときのクエリのサポートを改善しました。
- SUPER データ型によるマテリアライズドビューの更新のパフォーマンスを改善しました。
- ANY_VALUE 関数による INTERVAL リテラルを集約するためのサポートを追加しました。
- ストリーミング取り込みで、ストリーミングデータを消去する新しい SQL コマンドとして `DELETE FROM streaming_materialized_views WHERE <where filter clause>` をサポートするようになりました。
- DECODE 関数は、等価条件の結果に応じて、特定の値を別の特定の値またはデフォルト値に置換します。DECODE には次の 3 つのパラメータが必要になりました。
  - expression
  - search
  - 結果
- ストアドプロシージャに機能を追加して、データオーバーフローのデータ型変換エラーをキャッチしたり、例外処理ブロック内で処理したりできるようにしました。
- enable_case_sensitive_identifier をセッションのデフォルト設定と異なるものに変更すると、行レベルのセキュリティまたは動的データマスキングで保護された関係のクエリ時にエラーが表示されるようになりました。さらに、プロビジョニングされたクラスターまたはサーバーレス名前空間で行レベルのセキュリティまたは動的データマスキングポリシーが適用されている場合、次の設定はブロックされます:

  ```
  ALTER USER <current_user> SET case-sensitive identifier.
  ```

- MERGE コマンドは、ターゲットテーブルとソーステーブルのみを必要とする簡略化された構文をサポートするようになりました。詳細については、「Amazon Redshift データベース開発者ガイド」の「MERGE」を参照してください。
- 同一の動的データマスキングポリシーを、同じ優先順位を持つ、または優先順位を指定せずに、複数のユーザーやロールにアタッチするサポートを追加しました。
- ALTER TABLE ADD COLUMN を使用して新しい列を追加するときに COLLATION を指定できるようになりました。
- 同時実行スケーリングクラスターと Amazon Redshift Serverless での QMR ルールの適用が遅延する問題を修正しました。
- Amazon Redshift フェデレーテッドクエリは、Amazon RDS for PostgreSQL と Amazon Aurora PostgreSQL でのタイムスタンプ付きのタイムゾーンのプッシュダウンサポートを拡張しました。
- フェデレーテッドクエリで、Amazon RDS for MySQL と Aurora MySQL の数字で始まるデータベース名を使用できるようになりました。
- SYS_ANALYZE_HISTORY ビューを追加しました。このビューには、ANALYZE オペレーションのレコードの詳細が含まれます。
- SYS_ANALYZE_COMPRESSION_HISTORY ビューを追加しました。このビューには、COPY コマンドまたは ANALYZE COMPRESSION コマンドの実行時の圧縮分析オペレーションに関するレコードの詳細が含まれます。
- SYS_SESSION_HISTORY ビューを追加しました。このビューには、アクティブなセッション、過去のセッション、再開されたセッションに関するレコードの詳細が含まれます。
- SYS_TRANSACTION_HISTORY ビューを追加しました。このビューには、コミットにかかった時間、コミットされたデータ共有のブロック数、および分離レベルを示すトランザクションレベル分析に関するレコードの詳細が含まれます。
- SVV_REDSHIFT_SCHEMA_QUOTA ビューを追加しました。このビューには、データベース内の各スキーマのクォータと現在のディスク使用量に関するレコードが含まれます。
- SYS_PROCEDURE_CALL ビューを追加しました。このビューには、開始時刻、終了時刻、ストアドプロシージャ呼び出しのステータス、ネストされたストアドプロシージャ呼び出しの呼び出し階層など、ストアドプロシージャ呼び出しに関するレコードが含まれます。
- SYS_CROSS_REGION_DATASHARING_USAGE ビューを追加しました。このビューには、クロスリージョンデータ共有の使用状況の追跡に関するレコードが含まれます。
- SYS_PROCEDURE_MESSAGES ビューを追加しました。このビューには、ログに記録されたストアドプロシージャメッセージの追跡情報に関するレコードが含まれます。

- SYS_UDF_LOG ビューを追加しました。このビューには、ユーザー定義関数の呼び出し、エラー、警告、またはトレース (該当する場合) のシステムログメッセージの追跡に関するレコードが含まれます。
- 新しい列 IS_RECURSIVE、IS_NESTED、S3LIST_TIME、GET_PARTITION_TIME を SYS_EXTERNAL_QUERY_DETAIL に追加しました。
- Redshift Serverless の新しいコンピューティングコスト管理設定である MaxRPU が追加されました。MaxRPU では、Redshift Serverless がワークグループごとにスケーリングできる最大コンピューティングレベルを選択することで、必要に応じてコンピューティングの上限しきい値を指定して、さまざまな時点でのデータウェアハウスのコストを制御できます。
- INTERVAL リテラルの出力が数値間隔文字列で修正されました。例えば、`INTERVAL '1' YEAR` として指定された間隔が、`"00:00:00` の代わりに `1 YEAR` を返すようになりました。さらに、INTERVAL リテラルの出力は、指定された最小の INTERVAL コンポーネントに切り捨てられます。例えば、`INTERVAL '1 day 1 hour 1 minute 1.123 seconds' HOUR TO MINUTE` は `1 day 01:01:00` に切り捨てられます。

Amazon Redshift パッチ 177

このパッチのクラスターバージョン:

- 1.0.57922 - 現在のトラックバージョン – 2023 年 10 月 12 日にリリース
- 1.0.57799 - Amazon Redshift Serverless バージョン – 2023 年 10 月 10 日にリリース
- 1.0.57798 - 現在のトラックバージョン – 2023 年 10 月 10 日にリリース
- 1.0.57085 - 現在のトラックバージョン – 2023 年 9 月 26 日にリリース
- 1.0.56899 - Amazon Redshift Serverless バージョン – 2023 年 9 月 21 日にリリース
- 1.0.56754 - 現在のトラックバージョン – 2023 年 9 月 21 日にリリース
- 1.0.56242 - 現在のトラックバージョン – 2023 年 9 月 11 日にリリース
- 1.0.55539 - Amazon Redshift Serverless バージョン – 2023 年 8 月 28 日にリリース
- 1.0.55524 - 現在のトラックバージョン – 2023 年 8 月 28 日にリリース
- 1.0.54899 - 現在のトラックバージョン — 2023 年 8 月 15 日にリリース
- 1.0.54899 - 現在のトラックバージョン – 2023 年 8 月 14 日にリリース
- 1.0.54899 - 現在のトラックバージョン — 2023 年 8 月 15 日にリリース
- 1.0.54239 - 現在のトラックバージョン — 2023 年 8 月 3 日にリリース
- 1.0.54321 - Amazon Redshift Serverless バージョン – 2023 年 8 月 3 日にリリース

このパッチの新機能と改善点

- SYS_MV_STATE ビューを追加しました。このビューには、マテリアライズドビューのすべての状態遷移の行が含まれています。SYS_MV_STATE は、Amazon Redshift Serverless および Amazon Redshift のプロビジョンドインスタンスの MV 更新モニタリングに使用できます。
- SYS_USERLOG ビューを追加しました。このビューには、ユーザーの作成、ユーザーの削除、ユーザーの変更 (名前変更)、ユーザーの変更 (プロパティの変更) に関するデータベースユーザー変更の詳細が記録されます。
- SYS_COPY_REPLACEMENTS ビューを追加しました。このビューは、無効な UTF-8 文字を COPY コマンドの ACCEPTINVCHARS オプションで置き換えたときのログ記録を表示します。
- SYS_SPATIAL_SIMPLIFY ビューを追加しました。このビューには、COPY コマンドを使用して簡略化した空間ジオメトリオブジェクトに関する情報が表示されます。
- SYS_VACUUM_HISTORY ビューを追加しました。このビューを使用して VACUUM 操作の詳細と結果を確認できます。
- SYS_SCHEMA_QUOTA_VIOLATIONS ビューを追加しました。このビューは、スキーマのクォータを超過したときのオカレンス、タイムスタンプ、XID、その他の有用な情報を記録します。
- SYS_RESTORE_STATE ビューを追加しました。このビューを使用すると、非同期のクラシックサイズ変更中にクラスター内の各テーブルの再分散の進行状況をモニタリングできます。
- SYS_EXTERNAL_QUERY_ERROR ビューを追加しました。このビューは、Redshift Spectrum スキャンエラーに関する情報を返します。
- タグパラメータを CREATE MODEL コマンドに追加し、オートパイロットトレーニングジョブのトレーニングコストを追跡できるようにしました。
- Amazon Redshift クラスターにカスタムドメイン名 (CNAME) を追加しました。
- Apache Iceberg のプレビューサポートを追加し、Amazon Redshift 内で Apache Iceberg テーブルに分析クエリを実行できるようにしました。
- ワークロード管理 (WLM) のパラメータグループでユーザーロールを使用するためのサポートを追加しました。
- AWS Glue Data Catalog の自動マウントのサポートを追加し、データレイクでクエリを実行しやすくしました。
- GROUP BY 句なしでグループ化関数を使用したり、WHERE 句でグループ化演算を使用したりするとエラーが発生するなどの機能を追加しました。
- ストアドプロシージャに機能を追加して、ゼロ除算エラーをキャッチしたり、例外処理ブロック内で処理したりできるようにしました。

- ソーステーブルがデータ共有テーブルである場合に、クエリが同時実行スケーリングを使用してテーブルにデータを書き込めないというバグを修正しました。
- enable_case_sensitive_identifier に記載の大文字と小文字を区別する識別子を修正し、MERGE ステートメントを使用できるようにしました。
- 関数 pg_get_late_binding_view_cols() に対するクエリが時々無視されることがあるというバグを修正しました。このようなクエリは、いつでもキャンセルできるようになりました。
- プロデューサーでバキュームジョブを実行するときに、コンシューマーで実行するデータ共有クエリのパフォーマンスが向上しました。
- データ共有と同時実行スケーリングクエリのパフォーマンスが向上しました。特に、プロデューサーで同時にデータを変更する場合や、コンシューマーにアタッチされた同時実行スケーリングインスタンスにオフロードする場合に役立ちます。

Amazon Redshift パッチ 176

このパッチのクラスターバージョン:

- 1.0.56738 - 現在のトラックバージョン – 2023 年 9 月 21 日にリリース
- 1.0.55837 - 現在のトラックバージョン – 2023 年 9 月 11 日にリリース
- 1.0.54776 - 現在のトラックバージョン — 2023 年 8 月 15 日にリリース
- 1.0.54052 - 現在のトラックバージョン — 2023 年 7 月 26 日にリリース
- 1.0.53642 - Amazon Redshift Serverless バージョン – 2023 年 7 月 20 日にリリース
- 1.0.53301 - 現在のトラックバージョン — 2023 年 7 月 20 日にリリース
- 1.0.52943 - Amazon Redshift Serverless バージョン – 2023 年 7 月 7 日にリリース
- 1.0.52931 - 現在のトラックバージョン — 2023 年 7 月 7 日にリリース
- 1.0.52194 - Amazon Redshift Serverless バージョン – 2023 年 6 月 21 日にリリース
- 1.0.51986 - 現在のトラックバージョン — 2023 年 6 月 16 日にリリース
- 1.0.51594 - 現在のトラックバージョン — 2023 年 6 月 9 日にリリース

このパッチの新機能と改善点

- 空のグループ化セットに GROUP BY () を書き込む際のエラー処理が改善されました。これは以前は無視されていましたが、現在はパーサーエラーを返します。

- SUPER 列を使用してマテリアライズドビューを段階的に更新する場合のパフォーマンスの拡張。
- ALTER TABLE <target_tbl> APPEND FROM <streaming_mv> – (ATA) SQL コマンドは、ソースとしてのテーブルに加えて、ソースとしてのストリーミングマテリアライズドビュー (MV) からターゲットテーブルへのすべてのレコードの移動をサポートするようになりました。ストリーミング MV での ATA のサポートにより、ユーザーはストリーミング MV 内のレコードをすべて別のテーブルに移動してすばやく消去し、データの増加を管理できます。
- TRUNCATE <streaming_mv> – SQL コマンドは、テーブルに加えて、ストリーミングマテリアライズドビュー (MV) のすべてのレコードの切り詰めをサポートするようになりました。TRUNCATE は、ストリーミング MV 構造はそのままに、ストリーミング MV のすべてのレコードを削除します。ストリーミング MV で TRUNCATE を実行すると、ストリーミング MV のすべてのレコードを迅速に消去して、データの増加を管理できます。
- SELECT コマンドに QUALIFY 句の機能を追加しました。
- Amazon Forecast との統合による、Redshift の機械学習での時系列予測のサポート。
- AWS Glue Data Catalog の自動マウントがサポートされているため、外部スキーマ参照を作成する余分な手順を行うことなく、データレイクへのクエリを簡単に行うことができます。
- RLS ポリシーの変更がサポートされるようになりました。詳細については、「ALTER RLS POLICY」を参照してください。
- Lambda UDF が、CREATE FUNCTION ステートメントの STABLE 関数ボラティリティパラメータをサポートするようになりました。CREATE FUNCTION ステートメントで STABLE パラメータを使用し、同じ引数で Lambda UDF が複数回呼び出されると、Lambda UDF 関数の予想呼び出し回数が減少します。STABLE 関数のボラティリティカテゴリについては、「CREATE FUNCTION」の「パラメータ」で詳しく説明しています。
- 複数の Lambda UDF のパフォーマンスの改善。具体的には、行レベルセキュリティ (RLS) ポリシーで保護されているテーブルをクエリするときのレコードバッチ処理サポートが改善されました。
- Amazon Redshift RA3 クラスター全体の暗号化時間の短縮、および暗号化中のデータウェアハウスの可用性の向上。詳細については、「Amazon Redshift データベースの暗号化」を参照してください。
- 新しいシステムビュー SYS_MV_REFRESH_HISTORY が Redshift に追加されました。SYS_MV_REFRESH_HISTORY ビューには、マテリアライズドビューの更新作業用の行が含まれています。SYS_MV_REFRESH_HISTORY を使用すると、マテリアライズドビューの更新履歴を確認できます。SYS_MV_REFRESH_HISTORY はすべてのユーザーに表示されます。スーパーユーザーはすべての行を表示できますが、通常のユーザーは自分のデータのみを表示できます。

システムビュー SYS_QUERY_DETAIL に新しい列 SPILLED_BLOCK_LOCAL_DISK が追加されました。新しい列 SPILLED_BLOCK_LOCAL_DISK は、ローカルディスクに流出したブロックをお客様が特定するのに役立ちます。SYS_QUERY_DETAIL を使用して、ステップレベルでクエリの詳細を表示できます。SYS_QUERY_DETAIL はすべてのユーザーが表示可能です。スーパーユーザーはすべての行を表示できます。通常のユーザーはアクセスできるメタデータのみを表示できます。

- 新しいシステムビュー SYS_QUERY_TEXT が Amazon Redshift Serverless と Amazon Redshift プロビジョニングに追加されました。SYS_QUERY_TEXT ビューは、プロビジョニングされたクラスターの SVL_STATEMENTTEXT に似ています。SYS_QUERY_TEXT ビューの sequence 列を使用すると、SQL ステートメントのテキスト全体を取得できます。

Amazon Redshift パッチ 175

このパッチのクラスターバージョン:

- 1.0.53064 - 現在のトラックバージョン — 2023 年 7 月 7 日にリリース
- 1.0.51973 - 現在のトラックバージョン — 2023 年 6 月 16 日にリリース
- 1.0.51781 - 現在のトラックバージョン — 2023 年 6 月 10 日にリリース
- 1.0.51314 - Amazon Redshift Serverless バージョン – 2023 年 6 月 3 日にリリース
- 1.0.51304 - 現在のトラックバージョン — 2023 年 6 月 2 日にリリース
- 1.0.50708 - 現在のトラックバージョン — 2023 年 5 月 19 日にリリース
- 1.0.50300 - 現在のトラックバージョン — 2023 年 5 月 8 日にリリース
- 1.0.49710 - Amazon Redshift Serverless バージョン — 2023 年 4 月 28 日にリリース
- 1.0.49676 - 現在のトラックバージョン — 2023 年 4 月 28 日にリリース

このパッチの新機能と改善点

- 軽微なバグを修正。
- Amazon Redshift ストリーミング取り込みでは、クロスリージョンストリーミング取り込みがサポートされるようになりました。この場合、ソース Amazon Kinesis Data Streams (KDS) または Amazon Managed Streaming for Apache Kafka (MSK) のトピックは、Amazon Redshift データウェアハウスが配置されている AWS リージョンとは異なる AWS リージョンに配置できます。

「Amazon Kinesis Data Streams からストリーミング取り込みを開始する方法」のドキュメントが改訂され、REGION キーワードの使用方法についての説明が追加されました。

- エジプトの夏時間の調整。
- RA3 クラスターの暗号化にかかる全体的な時間が短縮されました。

Amazon Redshift パッチ 174

1.0.51296 – 2023 年 6 月 2 日にリリース

トレーリングトラックにリリース。リリースノートはありません。

1.0.50468 - 2023 年 5 月 12 日にリリース

メンテナンスリリース。リリースノートはありません。

1.0.49780、1.0.49868、1.0.49997 — 2023年4月28日にリリース

このバージョンのリリースノート:

- Lambda UDF のバッチ処理サポートが改善されました。
- Lambda UDF の増分バッチ処理。
- ソースデータの変更を Amazon Redshift テーブルに適用するための新しい MERGE SQL コマンド。
- Amazon Redshift データウェアハウス内の機密データを保護するプロセスを簡素化する新しい動的データマスキング機能。
- Lake Formation とのデータ共有のための新しい集中型アクセス制御。これによって、Lake Formation API と AWS コンソールを使用して Amazon Redshift データ共有内のテーブルとビューに対するアクセス許可付与の管理、アクセス制御の表示、およびアクセス許可の監査が可能になります。
- エジプトの夏時間の調整。

1.0.49087 — 2023 年 4 月 12 日にリリース

メンテナンスリリース。リリースノートはありません。

1.0.48805 — 2023 年 4 月 5 日にリリース

このバージョンのリリースノート:

- Amazon Redshift では、Amazon Redshift の新しい圧縮エンコーディングである BYTEDICT を使用して、文字列を多用するクエリのパフォーマンスをさらに強化しました。これにより、LZO や ZSTD などの代替圧縮エンコーディングと比較して、文字列ベースのデータ処理が 5 倍から 63 倍に高速化されます。詳細については、「Amazon Redshift データベースデベロッパーガイド」の「圧縮エンコード」を参照してください。

1.0.48004 — 2023 年 3 月 17 日にリリース

メンテナンスリリース。リリースノートはありません。

1.0.47470 — 2023 年 3 月 11 日にリリース

このバージョンのリリースノート:

- `pg_catalog.svv_table_info` でのクエリパフォーマンスが向上しました。新しい列 `create_time` も追加されました。テーブルを作成すると、この列に日付/タイムスタンプが UTC で保存されます。
- フェデレーテッドクエリでセッションレベルのタイムアウトを指定するためのサポートを追加しました。

Amazon Redshift パッチ 173

1.0.49788 — 2023 年 4 月 28 日にリリース

このバージョンのリリースノート:

- エジプトの夏時間の調整。

1.0.49074 — 2023 年 4 月 12 日にリリース

このバージョンのリリースノート:

- タイムゾーン設定が IANA ライブラリリリース 2022g に更新されました。

1.0.48766 — 2023 年 4 月 5 日にリリース

メンテナンスリリース。リリースノートはありません。

1.0.48714 — 2023 年 4 月 5 日にリリース

メンテナンスリリース。リリースノートはありません。

1.0.48022 — 2023 年 3 月 17 日にリリース

メンテナンスリリース。リリースノートはありません。

1.0.47357 - 2023 年 3 月 7 日にリリース

メンテナンスリリース。リリースノートはありません。

1.0.46987 — 2023 年 2 月 24 日にリリース

メンテナンスリリース。リリースノートはありません。

1.0.46806 — 2023 年 2 月 18 日にリリース

メンテナンスリリース。リリースノートはありません。

1.0.46607 — 2023 年 2 月 13 日にリリース

このバージョンのリリースノート:

- 分散スタイルが DISTSTYLE KEY に設定されている場合、テーブルの手動設定のインターリーブソートキーを自動的に複合ソートキーに変換できるようになり、テーブルのパフォーマンス向上に役立っています。これは、Amazon Redshift Serverless にスナップショットを復元するときに行われます。

1.0.45698 — 2023 年 1 月 20 日にリリース

このバージョンのリリースノート:

- UNLOAD コマンドにファイル拡張子パラメータを追加して、ファイル拡張子がファイル名に自動的に追加されるようにします。

- RLS で保護されたオブジェクトをデータ共有に追加するとき、または既にデータ共有の一部になっている場合に、デフォルトで保護をサポートします。管理者はデータ共有の RLS をオフにして、コンシューマーが保護されたオブジェクトにアクセスできるようにできるようになりました。
- モニタリング向けに、新しいシステムテーブルとして、SVV_ML_MODEL_INFO、SVV_MV_DEPENDENCY、SYS_LOAD_DETAIL を追加します。また、data_skewness、time_skewness 列をシステムテーブル SYS_QUERY_DETAIL に追加します。

Amazon Redshift パッチ 172

このパッチのクラスターバージョン:

- 1.0.46534 — 2023 年 2 月 18 日にリリース
- 1.0.46523 — 2023 年 2 月 13 日にリリース
- 1.0.46206 — 2023 年 2 月 1 日にリリース
- 1.0.45603 — 2023 年 1 月 20 日にリリース
- 1.0.44924 – 2022 年 12 月 19 日にリリース
- 1.0.44903 – 2022 年 12 月 18 日にリリース
- 1.0.44540 – 2022 年 12 月 13 日にリリース
- 1.0.44126 – 2022 年 11 月 23 日にリリース
- 1.0.43980 – 2022 年 11 月 17 日にリリース

このパッチの新機能と改善点

- CTAS によって作成されたテーブルは、デフォルトで AUTO です。
- マテリアライズドビューでの行レベルセキュリティ (RLS) のサポートを追加しました。
- S3 タイムアウトを増やして、クロスリージョンのデータ共有を改善しました。
- 新しい空間関数 ST_GeomFromGeohash を追加しました。
- 複合プライマリキーからの分散キーの自動選択が改善され、追加設定不要のパフォーマンスが向上しました。
- 複合プライマリキーを持つテーブルの分散キーに自動プライマリキーが追加され、追加設定不要のパフォーマンスが向上しました。

- 同時実行スケーリングが改善され、データ変更があってもより多くのクエリをスケーリングできるようになりました。
- データ共有のクエリのパフォーマンスが向上しました。
- 分類モデルに機械学習確率メトリクスを追加しました。
- モニタリング向けに次の新しいシステムテーブルを追加しました: SVV_USER_INFO、SVV_MV_INFO、SYS_CONNECTION_LOG、SYS_DATASHARE_USAGE_PRODUCER、SY
- Parquet および ORC ファイル形式の外部テーブルで VARBYTE 列をクエリするためのサポートが追加されました。

Amazon Redshift パッチ 171

このパッチのクラスターバージョン:

- 1.0.43931 – 2022 年 11 月 16 日にリリース
- 1.0.43551 – 2022 年 11 月 5 日にリリース
- 1.0.43331 – 2022 年 9 月 29 日にリリース
- 1.0.43029 – 2022 年 9 月 26 日にリリース

このパッチの新機能と改善点

- CONNECT BY サポート: CONNECT BY SQL コンストラクトのサポートが追加され、データセット内の親子関係に基づいて、データウェアハウス内の階層データに再帰的にクエリを実行できるようになりました。

Amazon Redshift パッチ 170

このパッチのクラスターバージョン:

- 1.0.43922 – 2022 年 11 月 21 日にリリース
- 1.0.43573 – 2022 年 11 月 7 日にリリース
- 1.0.41881 – 2022 年 9 月 20 日にリリース
- 1.0.41465 – 2022 年 9 月 7 日にリリースしました。
- 1.0.40325 - 2022 年 7 月 27 日にリリース

このパッチの新機能と改善点

- ST_GeomFromGeoJSON: GeoJSON 表現の VARCHAR から Amazon Redshift 空間ジオメトリオブジェクトを構築します。

Amazon Redshift パッチ 169

このパッチのクラスターバージョン:

- 1.0.41050 – 2022 年 9 月 7 日にリリース
- 1.0.40083 - 2022 年 7 月 16 日にリリース
- 1.0.39734 - 2022 年 7 月 7 日にリリース
- 1.0.39380 - 2022 年 6 月 23 日にリリース
- 1.0.39251 - 2022 年 6 月 15 日にリリース
- 1.0.39009 - 2022 年 6 月 8 日にリリース

このパッチの新機能と改善点

- ロールベースのアクセス制御をサポートするため、デフォルト権限の変更コマンドのパラメータとしてロールを追加します。
- Parquet と ORC ファイルからコピーする際に無効な UTF-8 文字の置き換えをサポートするため、ACCEPTINVCHARS パラメータを追加します。
- OBJECT (k,v) 関数を追加して、キーと値のペアから SUPER オブジェクトを構築します。

Amazon Redshift パッチ 168

このパッチのクラスターバージョン:

- 1.0.38698 – 2022 年 5 月 25 日にリリース
- 1.0.38551 – 2022 年 5 月 20 日にリリース
- 1.0.38463 – 2022 年 5 月 18 日にリリース
- 1.0.38361 – 2022 年 5 月 13 日にリリース
- 1.0.38199 - 2022 年 5 月 9 日にリリース
- 1.0.38112 — 2022 年 5 月 6 日にリリース

- 1.0.37684 — 2022 年 4 月 20 日にリリース

このパッチの新機能と改善点

- Amazon Redshift ML の線形学習モデルタイプのサポートを追加しました。
- SQL トランザクション分離レベルの SNAPSHOT オプションを追加しました。
- VARBYTE および VARCHAR データの新しいハッシュアルゴリズムとして `farmhashFingerprint64` を追加しました。
- マテリアライズドビューの増分更新で AVG 関数をサポートします。
- Redshift Spectrum の外部テーブルの相関サブクエリをサポートします。
- 初期状態のクエリパフォーマンスを向上させるために、Amazon Redshift は、特定のテーブルにおける単一列のプライマリキーを分散キーとして自動的に選択します。

Amazon Redshift の動作の変更

Amazon Redshift が進化し、改善が進むにつれ、パフォーマンス、セキュリティ、ユーザーエクスペリエンスを向上させるために、動作に一定の変更が加えられます。このページは、これらの重要な更新に関する情報を入手して、アクションを実行し、ワークロードが中断する可能性を回避するための包括的なリソースとして利用できます。

今後予定されている動作変更

今後予定されている動作変更について、以下に説明します。

トピック

- Amazon Redshift は、2026 年 9 月 30 日以降、一時停止中のデータ共有プロデューサークラスターの自動スナップショットを作成
- SUPER データ型は、パッチ 206 以降、より大きな個々の文字列をサポート
- 2026 年 8 月 17 日以降の Amazon Redshift Serverless API における AWS KMS キーアクセス許可の適用
- Amazon Redshift では、パッチ 208 以降、ログイン試行が複数回失敗した後にユーザーロックアウトを適用します
- 2026 年 6 月 8 日に、Amazon Redshift Serverless および Amazon Redshift RG インスタンスの手動スナップショットの請求モデルを強化
- Lake Formation テーブルの Iceberg DELETE には、パッチ 202 で始まる DELETE アクセス許可が必要です
- Amazon Redshift Serverless は、パッチ 202 以降のスナップショット復元でゼロ ETL および S3 イベント統合を保持
- 2026 年 12 月 31 日の Amazon Redshift ODBC 1.x ドライバーのサポート終了
- スカラー Python UDFは 2026 年 6 月 30 日にサポートが終了します
- 2026 年 2 月 27 日以降のマテリアライズドビュー (MV) の自動 REFRESH の動作変更
- Amazon Redshift は、2026 年 2 月 16 日以降、データ共有を通じてコンシューマー情報にアクセスする関数をサポートしません
- 2026 年 10 月 31 日以降に有効となる Transport Layer Security (TLS) の最小バージョンに関する変更
- Amazon Redshift は、2025 年 10 月 30 日以降、新しいスカラー Python UDF の作成をサポートしません

Amazon Redshift は、2026 年 9 月 30 日以降、一時停止中のデータ共有プロデューサークラスターの自動スナップショットを作成

2026 年 9 月 30 日以降、Amazon Redshift は、一時停止中のデータ共有プロデューサークラスターの自動スナップショットを作成します。コンシューマーは一時停止したプロデューサーに書き込むことができるため、これらのスナップショットはそれらのデータ共有の書き込みをキャプチャし、復旧ポイントを提供します。

データ共有プロデューサークラスターを一時停止すると、この影響を受ける可能性があります。

以前は、Amazon Redshift はこれらのクラスターの一時停止中に自動スナップショットを作成しなかったため、復元に使用できる最新の自動スナップショットは、クラスターの一時停止時に作成されたものでした。自動スナップショットが、一時停止期間中も引き続き作成されるようになりました。

これらのスナップショットは、クラスターの既存の自動スナップショットのスケジュールと保持期間に従い、アクションは必要ありません。データ共有プロデューサーではない一時停止中のクラスターの場合、Amazon Redshift はクラスターの一時停止中に自動スナップショットを作成しません。

詳細については、「自動スナップショット」を参照してください。

SUPER データ型は、パッチ 206 以降、より大きな個々の文字列をサポート

パッチ 206 以降、Amazon Redshift は SUPER データ型 (半構造化データの保存に使用) 内で最大 16,000,000 バイトの個々の文字列をサポートします。この制限は、インメモリ文字列オペレーションにも適用されます。この機能を使用するには、データベースパラメータグループを作成し、`enable_large_strings_opt_in` パラメータを `Yes` に設定します。次に、パラメータグループをデータウェアハウスにアタッチします。プロビジョニングされたクラスターの場合は、クラスターを再起動してパラメータの変更を適用する必要があります。

この変更は、SUPER データ型を使用して大きな JSON ドキュメントを保存する場合に影響する可能性があります。また、65,535 バイトを超える文字列を処理するビュー、ユーザー定義関数 (UDF)、またはストアドプロシージャ内で `VARCHAR(MAX)` に依存しているワークロードがある場合、これらにも影響する可能性があります。

`enable_large_strings_opt_in` を `Yes` に設定すると、インメモリオペレーションは `VARCHAR(MAX)` を `VARCHAR(16000000)` に解決します。これらのオペレーションには、CAST、ビュー、UDF、ストアドプロシージャが含まれます。これにより、65,535 バイトを超える文字列を処理できます。この変更は、VARCHAR 列の永続的ストレージには影響しません。`CREATE TABLE`

と VARCHAR(MAX) は引き続き VARCHAR(65535) 列を作成します。パラメータを No に設定すると、Amazon Redshift はデータウェアハウスの既存の動作を保持します。データウェアハウスの現在の状態を確認するには、SELECT is_large_string_enabled(); を実行します。

enable_large_strings_opt_in を Yes に設定すると、次のようになります。

- SUPER 列への COPY および INSERT は、以前の 65,535 バイトの制限を超えるサイズをサポートします。JSON_PARSE、JSON_SERIALIZE、LISTAGG などの文字列関数も、これらのより大きなサイズをサポートします。
- 既存のビュー、UDF、ストアドプロシージャは、再作成するまで 65,535 バイトに制限されます。
- 既存のビューが VARCHAR(MAX) を使用しており、このビューに他のオブジェクト (他のビューなど) が依存している場合、CREATE OR REPLACE VIEW は失敗します。エラーとして、ビュー列のデータ型を変更できないというメッセージが表示されます。ビューを更新するには、依存オブジェクトを削除し、ビューを削除して再作成してから、依存オブジェクトを再作成します。VARCHAR(MAX) を使用していて依存オブジェクトを持たないビューは、これらの追加の手順なしで置換できます。
- この設定は、データを共有するプロデューサー側とコンシューマー側のデータウェアハウス間で一致させておきます。

enable_large_strings_opt_in を No に設定すると、次のようになります。

- SUPER 列に 65,535 バイトを超える文字列を COPY または INSERT することはできません。SUPER を通じて既存の文字列を読み取ることは引き続き可能です。ただし、VARCHAR(MAX) にキャストすると、値は 65,535 バイトに切り詰められます。
- JSON_PARSE、JSON_SERIALIZE、LISTAGG などの文字列関数は、65,535 バイトを超える VARCHAR 値を処理できません。
- 65,535 バイトを超える文字列リテラルを処理するビュー、ストアドプロシージャ、および UDF に対してクエリを実行することはできません。
- パラメータを Yes に設定した状態で VARCHAR(MAX) を使用してビューを作成すると、そのビューに他のオブジェクト (他のビューなど) が依存している場合、CREATE OR REPLACE VIEW は失敗し、ビュー列のデータ型を変更できないという同じエラーメッセージが表示されます。前のリストで説明したのと同じ修復手順に従ってください。
- インスタンス間のスナップショットの復元では、長い文字列の取り込みに失敗する可能性があります。この失敗は、スナップショットの SUPER 列に 65,535 バイトを超える文字列が含まれている場合に発生します。このパラメータが No に設定されているクラスターにスナップショットを復元すると、この失敗が発生します。

パッチバージョンの詳細については、「Amazon Redshift のクラスターバージョン」を参照してください。

2026 年 8 月 17 日以降の Amazon Redshift Serverless API における AWS KMS キーアクセス許可の適用

Amazon Redshift Serverless は、特定の Amazon Redshift Serverless API を呼び出す IAM プリンシパルに対し、ターゲット名前空間を暗号化するカスタマーマネージドキーへの明示的な AWS KMS アクセス許可を持つことを要求する新しい認可チェックを実行するようになりました。以前は、Amazon Redshift は Amazon Redshift サービスロールを使用してユーザーに代わって AWS KMS オペレーションを実行していましたが、呼び出し元の IAM プリンシパル自体にそのキーを使用する権限があるかどうかは確認されていませんでした。

新しい認可チェックが有効になると、必要な AWS KMS アクセス許可がないプリンシパルからの API 呼び出しは `ValidationException: "Insufficient KMS permissions."` となって失敗します。カスタマーマネージドキーで暗号化された Amazon Redshift Serverless 名前空間を使用しており、IAM プリンシパルを使用して以下にリストされている影響を受ける API を呼び出すと、この影響を受ける可能性があります。

中断を回避するには、アカウント内のどの IAM プリンシパルがカスタマーマネージドキーで暗号化された名前空間に対して以下に示す API を呼び出すかを確認し、それらのプリンシパルに該当するキーに対する必要な AWS KMS アクセス許可があることを確認します。これらのアクセス許可は、AWS KMS キーポリシー、IAM アイデンティティベースのポリシー、または AWS KMS グラントを通じて付与できます。

この変更は、デフォルトの AWS 所有キーで暗号化された名前空間には影響しません。IAM プリンシパルに以下に示す必要な AWS KMS アクセス許可が既にある場合、アクションは必要ありません。

影響を受ける API と、名前空間のカスタマーマネージドキーに必要なアクセス許可を以下に示します。

- `CreateWorkgroup`、`RestoreFromSnapshot`、`RestoreFromRecoveryPoint` – `kms:Encrypt`、`kms:GenerateDataKey`、`kms:CreateGrant`、および `kms:Decrypt` が必要です。
- `RestoreTableFromSnapshot`、`RestoreTableFromRecoveryPoint` – `kms:Decrypt` が必要です。

詳細については、「AWS Key Management Service デベロッパーガイド」の「AWS KMS キーへのアクセスの制御」を参照してください。Amazon Redshift データベースの暗号化の詳細については、「Amazon Redshift データベース暗号化」を参照してください。

Amazon Redshift では、パッチ 208 以降、ログイン試行が複数回失敗した後にユーザーロックアウトを適用します

Amazon Redshift パッチ 208 以降、データウェアハウスのセキュリティ体制を強化するために、設定された回数を超えて連続してログインに失敗した場合、Amazon Redshift はデータベースユーザーを自動的にロックするようになりました。この機能は、デフォルトで失敗のしきい値 5 回 (2～50 の範囲で設定可能) で有効になっており、プロビジョニングされたクラスターとサーバーレスワークグループの両方に適用されます。

データベース接続 (パスワードベース) を使用して接続すると、この影響を受ける可能性があります。ロックされると、管理者またはスーパーユーザーがロックを解除するまで、正しいパスワードを使用してもログイン試行が拒否されます。IAM または AWS IAM アイデンティティセンターを介して認証するフェデレーティッド ID は、静的パスワードではなく有効期間の短いトークンを使用するため、影響を受けません。

この動作を管理するには、次の操作を行います。

- `ALTER SYSTEM SET max_failed_login_attempts TO value;` (2～50、デフォルト 5) でしきい値を設定します。
- `SHOW USER LOCKOUT;` を使用して、すべてのユーザーのロックアウトステータスを確認します。
- `ALTER USER username LOGIN PASSWORD 'password';` でユーザーをロック解除します。これにより、試行失敗カウンターもリセットされます。

詳細については、「SHOW USER LOCKOUT」、「ALTER USER」、および「max_failed_login_attempts」を参照してください。

2026 年 6 月 8 日に、Amazon Redshift Serverless および Amazon Redshift RG インスタンスの手動スナップショットの請求モデルを強化

2026 年 6 月 8 日に、Amazon Redshift は、Amazon Redshift Serverless および Amazon Redshift RG インスタンスでの手動スナップショットの請求モデルの強化を発表しました。この強化により、手動スナップショットは、アカウント内のすべてのアクティブな手動スナップショットに保存されて

いる一意のデータブロックに基づいて計測されるようになりました。複数のスナップショットに表示される共有データブロックは 1 回だけカウントされ、リージョンの既存の手動スナップショットレートで請求されます。

Amazon Redshift Serverless ワークグループまたは Amazon Redshift RG クラスターで手動スナップショットを使用している場合、この影響を受ける可能性があります。これにより、複数のスナップショットを維持しているお客様は、手動スナップショットのコストが削減されます。

アクションは必要ありません。強化された請求モデルは、既存の手動スナップショットと新しい手動スナップショットの両方に自動的に適用されます。

スナップショット料金の詳細については、Amazon Redshift 料金表を参照してください。

Lake Formation テーブルの Iceberg DELETE には、パッチ 202 で始まる DELETE アクセス許可が必要です

Amazon Redshift Patch 202 以降、Lake Formation (LF) マネージドテーブルに対する Iceberg の DELETE オペレーションには、Lake Formation の DELETE アクセス許可が必要です。UPDATE および MERGE オペレーションには、INSERT と DELETE の両方のアクセス許可が必要です。すべての Iceberg DML オペレーションには、ALTER アクセス許可が必要です。

AWS Lake Formation が管理する Iceberg テーブルに対して DELETE、UPDATE、または MERGE オペレーションを実行すると、この影響を受ける可能性があります。

以前は、INSERT アクセス許可のみ (DELETE アクセス許可なし) を持つプリンシパルが、LF マネージド Iceberg テーブルに対して DELETE オペレーションを実行できました。パッチ 202 以降、DELETE オペレーションには Lake Formation の DELETE アクセス許可が必要です。削除を実行するために INSERT アクセス許可のみに依存するプリンシパルは、DELETE アクセス許可が明示的に付与されるまで、アクセス許可エラーを受け取ります。S3 テーブルはこの変更の影響を受けません。

この変更後も DELETE オペレーションを引き続き実行するには、Iceberg テーブルに対する Lake Formation アクセス許可の付与を確認し、DELETE オペレーションを実行するプリンシパルに DELETE Lake Formation アクセス許可が付与されていることを確認します。AWS Lake Formation コンソールまたは `aws lakeformation list-permissions` AWS CLI コマンドを使用して、既存の付与を確認できます。

一時的に前の動作に戻す必要がある場合は、AWS サポートに連絡して、コードのデプロイを要求することなくクラスターでこの変更を無効にしてください。

パッチバージョンの詳細については、「Amazon Redshift のクラスターバージョン」を参照してください。

Amazon Redshift Serverless は、パッチ 202 以降のスナップショット復元でゼロ ETL および S3 イベント統合を保持

Amazon Redshift パッチ 202 から、スナップショットまたはリカバリポイントから同じサーバーレス名前空間に Amazon Redshift Serverless 名前空間を復元すると、名前空間とスナップショット/復旧ポイントに関連付けられたゼロ ETL および S3 イベント統合が自動的に維持されます。追加のアクションは不要です。

ゼロ ETL または S3 イベント統合が設定された Amazon Redshift Serverless 名前空間にスナップショットまたは復旧ポイントを復元すると、この影響を受ける可能性があります。

以前は、スナップショットまたは復旧ポイントを、関連付けられたゼロ ETL および S3 イベント統合が失敗したとマークされたサーバーレス名前空間に復元していて、復元完了後に手動の再作成が必要でした。パッチ 202 から、これらの統合はデフォルトで維持され、復元完了後にオペレーションが再開されます。

この機能は、同じサーバーレス名前空間に復元された場合にのみ Amazon Redshift Serverless に適用されます。スナップショットを別の名前空間に復元しても、統合は維持されません。プロビジョニングされたクラスターのスナップショット復元では、ゼロ ETL または S3 イベント統合は維持されません。

復元中に統合の維持をオプトアウトするには、AWS マネジメントコンソールの復元ページの [統合の維持] チェックボックスをオフにします。AWS CLI を使用している場合は、restore-from-snapshot または restore-from-recovery-point API オペレーションを呼び出すときに `--no-maintain-integration` パラメータを設定します。オプトアウトすると、統合は復元後に `FAILED` 状態になります。その後、失敗した統合を削除して再作成できます。

パッチバージョンの詳細については、「Amazon Redshift のクラスターバージョン」を参照してください。

2026 年 12 月 31 日の Amazon Redshift ODBC 1.x ドライバーのサポート終了

2026 年 12 月 31 日以降、Amazon Redshift は ODBC 1.x ドライバーのサポートを終了します。お客様からのフィードバックに基づいて、元のサポート終了日を 2026 年 9 月 30 日から 2026 年 12 月

31 日に延長し、移行のための追加の時間を確保しました。これは、Amazon Redshift でプロビジョニングされたクラスターとサーバーレスワークグループの両方に適用されます。

ODBC 1.x ドライバーのいずれかのバージョンを使用して Amazon Redshift に接続すると、この影響を受ける可能性があります。ODBC 1.x ドライバーを使用しているかどうかを確認するには、次のクエリを実行します。

```
SELECT *
FROM SYS_CONNECTION_LOG
WHERE
(driver_version ilike 'Amazon Redshift ODBC Driver 1%'
OR driver_version ilike 'Redshift ODBC Driver 01%'
OR driver_version ilike 'Redshift ODBC Driver 1,%'
)
OR
(application_name ilike 'Amazon Redshift ODBC Driver 1%');
```

Amazon Redshift ODBC ドライバー接続のテクニカルサポートを引き続き受けるには、2026 年 12 月 30 日までに最新の Amazon Redshift ODBC 2.x ドライバーに移行してください。

本番環境で ODBC 2.x ドライバーに移行する前に、概念実証を徹底的に実施して、新しいドライバーがすべての機能要件を満たしていることを確認することをお勧めします。

Amazon Redshift ODBC 2.x ドライバーの最新バージョンを使用し、Amazon Redshift に接続するときにアプリケーションを識別するように ApplicationName プロパティを設定することをお勧めします。

スカラー Python UDFは 2026 年 6 月 30 日にサポートが終了します

Amazon Redshift は、2026 年 6 月 30 日に Python UDF のサポートを終了します。別の方法として、Lambda UDF を使用することをお勧めします。

Lambda UDF には、Python UDF に比べて以下の利点があります。

- Lambda UDF は、UDF ロジック内から外部サービスと API に接続できます。
- Lambda UDF は Lambda コンピューティングリソースを使用します。コンピューティングやメモリを大量に消費する Lambda UDF は、Amazon Redshift のクエリパフォーマンスやリソースの同時実行には影響しません。

- Lambda UDF は Python コードの実行をサポートしています。Lambda UDF は特定のユースケースに応じて複数の Python ランタイムをサポートします。詳細については、「AWS Lambda デベロッパーガイド」の「Python で構築」を参照してください。
- カスタムコード実行は、別のサービス境界に分離できます。これにより、メンテナンス、モニタリング、予算編成、アクセス許可管理が簡素化されます。

Lambda UDF の作成と使用については、「Amazon Redshift データベース開発者ガイド」の「スカラー Lambda UDF」を参照してください。既存の Python UDF を Lambda UDF に変換する方法については、ブログ記事を参照してください。

2026 年 2 月 27 日以降のマテリアライズドビュー (MV) の自動 REFRESH の動作変更

2026 年 2 月 27 日以降、Amazon Redshift マテリアライズドビューの自動 REFRESH クエリは、バックグラウンドの自律プロセスとしてではなく、ユーザークエリとして実行されます。その結果、自動 REFRESH クエリは、他のユーザークエリと同じ優先度で実行されるようになりました。

この変更により、AUTO REFRESH を有効にしてマテリアライズドビューの鮮度を向上させることで、以前の動作と比較して、ベーステーブルに対する最新の変更が反映されるため、ビューを最新の状態に保つことができます。

注: MV の AUTO REFRESH の動作変更機能は、Amazon Redshift でプロビジョニングされたクラスターにおいて、パッチリリース P198 以降の CURRENT トラックでのみ有効になります。Serverless では現在無効になっています。

Amazon Redshift は、2026 年 2 月 16 日以降、データ共有を通じてコンシューマー情報にアクセスする関数をサポートしません

2026 年 2 月 16 日以降、Amazon Redshift は、データ共有を通じてコンシューマーユーザー、ロール、またはグループ情報にアクセスする `user_is_member_of` および関連関数の使用をサポートしなくなります。

2026 年 10 月 31 日以降に有効となる Transport Layer Security (TLS) の最小バージョンに関する変更

2026 年 10 月 31 日以降、Amazon Redshift では Transport Layer Security (TLS) の最小バージョン 1.2 が必須となります。TLS バージョン 1.0 または 1.1 を使用する着信接続は拒否されます。これは、Amazon Redshift でプロビジョニングされたクラスターとサーバーレスワークグループの両方に

適用されます。TLS を使用しない Amazon Redshift データウェアハウスは、この変更の影響を受けません。

TLS バージョン 1.0 または 1.1 を使用して Amazon Redshift に接続している場合、この更新の影響を受ける可能性があります。

現在使用している TLS バージョンを確認するには、次の操作を行います。

Amazon Redshift プロビジョニングの場合: STL_CONNECTION_LOG システムテーブル [1] の sslversion 列を確認します。

Amazon Redshift Serverless ワークグループの場合: SYS_CONNECTION_LOG システムテーブル [2] の ssl_version 列を確認します。

この変更後も Amazon Redshift データウェアハウスへの中断のないアクセスを維持するには、以下の手順に従ってください。

1. TLS 1.2 以降をサポートするようにクライアントを更新する
2. TLS 1.2 以降をサポートする最新のドライバーバージョンをインストールする

可能であれば、最新バージョンの Amazon Redshift ドライバー [3] を使用することをお勧めします。

[1] https://docs.aws.amazon.com/redshift/latest/dg/r_STL_CONNECTION_LOG.html

[2] https://docs.aws.amazon.com/redshift/latest/dg/SYS_CONNECTION_LOG.html

[3] https://docs.aws.amazon.com/redshift/latest/mgmt/configuring-connections.html

Amazon Redshift は、2025 年 10 月 30 日以降、新しいスカラー Python UDF の作成をサポートしません

Amazon Redshift は、2025 年 10 月 30 日以降、新しい Python UDF の作成をサポートしなくなります。既存の Python UDF は引き続き正常に機能します。この日付より前に既存の Python UDF を Lambda UDF に移行することを強くお勧めします。

Lambda UDF には、Python UDF に比べて以下の利点があります。

- Lambda UDF は、UDF ロジック内から外部サービスと API に接続できます。
- Lambda UDF は Lambda コンピューティングリソースを使用します。コンピューティングやメモリを大量に消費する Lambda UDF は、Amazon Redshift のクエリパフォーマンスやリソースの同時実行には影響しません。

- Lambda UDF は Python コードの実行をサポートしています。Lambda UDF は特定のユースケースに応じて複数の Python ランタイムをサポートします。詳細については、「AWS Lambda デベロッパーガイド」の「Python で構築」を参照してください。
- カスタムコード実行は、別のサービス境界に分離できます。これにより、メンテナンス、モニタリング、予算編成、アクセス許可管理が簡素化されます。

Lambda UDF の作成と使用については、「Amazon Redshift データベース開発者ガイド」の「スカラー Lambda UDF」を参照してください。既存の Python UDF を Lambda UDF に変換する方法については、ブログ記事を参照してください。

最近の動作の変更

トピック

- 2025 年 8 月 26 日以降、Amazon Redshift は最新の IANA タイムゾーンデータベースを使用します
- 2025 年 8 月 15 日以降に有効になる Amazon Redshift Serverless RPU の変更
- 2025 年 8 月 10 日以降に有効となるデータベース監査ログ記録に関する変更
- 2025 年 6 月 27 日以降に有効になるサーバーレスワークグループの仮想プライベートクラウドエンドポイントの変更
- 2025 年 5 月 2 日以降に有効になる、クエリモニタリングの変更
- 2025 年 1 月 10 日以降に有効になるセキュリティ変更

2025 年 8 月 26 日以降、Amazon Redshift は最新の IANA タイムゾーンデータベースを使用します

2025 年 8 月 26 日以降、Amazon Redshift は最新の IANA タイムゾーンデータベースパッチを採用してタイムゾーンを計算します。この変更により、特定のタイムゾーンと期間における日付と時刻の変換の仕組みが変更されます。この更新は、CONVERT_TIMEZONE 関数または TIMEZONE および AT TIME ZONE コマンドで実行されるような明示的なタイムゾーン変換や、特に TIMESTAMP 形式と TIMESTAMPTZ 形式間の型のキャスト操作中に発生する暗黙的な変換に影響します。

以下は、タイムゾーンと期間の組み合わせの更新のリストです。

- タイムゾーンで、2038 年以降の夏時間 (DST) が正しく反映されるようになりました。以前は、2038 年以降に DST が反映されたタイムゾーンはありませんでした。

- America/Toronto タイムゾーンとこれに連なるタイムゾーンでは、1947 年から 1950 年にかけて DST が、現地時間の午前 0 時ではなく午前 2 時に切り替わりました。
- Amazon Redshift は、すべてのタイムゾーンの標準化前の期間のローカル平均時間 (LMT) を正しく反映するようになりました。この期間はタイムゾーンに固有であり、ほとんどのタイムゾーンは 19 世紀半ばより前に標準化に切り替わりました。
- EET、CET、WET および MET は、略語ではなく通常のタイムゾーンとして扱われるようになりました。
- Amazon Redshift に次のタイムゾーン名が存在しなくなりました。
  - Asia/Riyadh87
  - Asia/Riyadh88
  - Asia/Riyadh89
  - Mideast/Riyadh87
  - Mideast/Riyadh88
  - Mideast/Riyadh89
  - US/Pacific-New

IANA タイムゾーンデータベースの詳細については、IANA タイムゾーンデータベースウェブサイトの「タイムゾーンデータベース」を参照してください。

2025 年 8 月 15 日以降に有効になる Amazon Redshift Serverless RPU の変更

2025 年 8 月 15 日以降、Amazon Redshift Serverless のベース Redshift 処理ユニット (RPU) の AWS アカウントクォータは、3,200 RPU または過去 6 か月の 最大集計ベース RPU の 1.5 倍のいずれか大きい方になります。

2025 年 8 月 10 日以降に有効となるデータベース監査ログ記録に関する変更

2025 年 8 月 10 日以降、Amazon Redshift はデータベース監査ログ記録を変更します。この変更に伴い、お客様側の対応が必要となります。Amazon Redshift は、データベースの接続とユーザーアクティビティに関する情報を Amazon S3 バケットと CloudWatch に記録します。2025 年 8 月 10 日以降、Amazon Redshift は Redshift IAM USER を指定するバケットポリシーを持つ Amazon S3 バケットへのデータベース監査ログ記録を中止します。監査ログ記録用の S3 バケットポリシー内で、代わ

りに Redshift SERVICE-PRINCIPAL を使用するようにポリシーを更新するようお勧めします。監査ログ記録の詳細については、「Amazon Redshift 監査ログ作成のためのバケットのアクセス許可」を参照してください。

ログ記録の中断を回避するには、2025 年 8 月 10 日より前に、S3 バケットポリシーを確認して更新し、関連するリージョンで Redshift SERVICE-PRINCIPAL へのアクセスを許可してください。データベース監査ログ記録の詳細については、「Amazon S3 でのログファイル」を参照してください。

ご質問やご不明点については、こちらのリンク (AWS サポート) から AWS サポートにお問い合わせください。

2025 年 6 月 27 日以降に有効になるサーバーレスワークグループの仮想プライベートクラウドエンドポイントの変更

2025 年 6 月 27 日以降、Amazon Redshift はサーバーレスワークグループの仮想プライベートクラウドエンドポイント (VPCE) サポートを変更します。この日より前に、Amazon Redshift はワークグループの作成中にエンドポイントを使用して単一のアベイラビリティーゾーン (AZ) にデプロイし、VPCE サポートを時間の経過とともに最大 3 つの AZ に拡張します。この日以降、Amazon Redshift はワークグループの作成時に指定された最大 3 つのアベイラビリティーゾーンに VPCE をデプロイします。

詳細については、「Amazon Redshift サーバーレスを使用する場合の考慮事項」を参照してください。

ご質問やご不明点については、こちらのリンク (AWS サポート) から AWS サポートにお問い合わせください。

トピック

- 2025 年 5 月 2 日以降に有効になる、クエリモニタリングの変更
- 2025 年 1 月 10 日以降に有効になるセキュリティ変更

2025 年 5 月 2 日以降に有効になる、クエリモニタリングの変更

2025 年 5 月 2 日以降、既存の Redshift Serverless ワークグループと新しく作成した Redshift Serverless ワークグループの両方について、[クエリの制限] タブでのクエリ CPU 時間 (`max_query_cpu_time`) メトリクスとクエリ CPU 使用率 (`max_query_cpu_percentage`) メトリクスの提供を終了します。この日以降、すべての Redshift Serverless ワークグループにおいて、これらのメトリクスに基づくすべてのクエリの制限が自動的に削除されます。

クエリの制限は、ランナウェイクエリをキャッチするように設計されています。ただし、クエリ CPU 時間 (max_query_cpu_time) とクエリ CPU 使用率 (max_query_cpu_percentage) は、クエリの存続期間中に変わる可能性があるため、ランナウェイクエリをキャッチするための一貫した効果的な方法ではありません。ランナウェイクエリをキャッチするには、一貫性のある実用的な情報を提供するクエリモニタリングメトリクスを活用することをお勧めします。それらの例を以下に示します。

- クエリ実行時間 (max_query_execution_time): クエリが想定期間内に完了するようにします。
- 戻り行数 (max_scan_row_count): 処理中のデータのスケールをモニタリングします。
- クエリキュー時間 (max_query_queue_time): キューイングに時間を費やすクエリを識別します。

サポートされているメトリクスの完全なリストについては、「Amazon Redshift Serverless のクエリモニタリングメトリクス」を参照してください。

2025 年 1 月 10 日以降に有効になるセキュリティ変更

セキュリティは、Amazon Web Services (AWS) における最優先事項です。そのため、Amazon Redshift 環境のセキュリティ体制を一層強固なものとするために、強化されたセキュリティデフォルトを導入します。これにより、追加の設定を行うことなくデータセキュリティのベストプラクティスを遵守しやすくなり、設定ミスのリスクも軽減されます。サービス中断の影響を回避するために、発効日までに、プロビジョニング済みのクラスターやサーバーレスワークグループの作成設定、スクリプト、ツールを確認し、新しいデフォルト設定に合わせた変更を適宜行ってください。

パブリックアクセスはデフォルトでは無効化

2025 年 1 月 10 日以降、新しく作成されたプロビジョニング済みクラスターと、スナップショットから復元されたクラスターのすべてに対して、パブリックアクセスはデフォルトで無効になります。このリリース以降、デフォルトでは、クラスターへの接続は、同じ仮想プライベートクラウド (VPC) 内のクライアントアプリケーションからのみ許可されるようになります。別の VPC のアプリケーションからデータウェアハウスにアクセスするには、クロス VPC アクセスを設定します。この変更は、CreateCluster および RestoreFromClusterSnapshot API オペレーション、および対応する SDK コマンドと AWS CLI コマンドに反映されます。Amazon Redshift コンソールからプロビジョニング済みクラスターを作成した場合、そのクラスターのパブリックアクセスはデフォルトで無効になります。

それでもパブリックアクセスが必要な場合は、デフォルトをオーバーライドして、CreateCluster または RestoreFromClusterSnapshot API オペレーションの実行時に

PubliclyAccessibleパラメータを true に設定する必要があります。パブリックアクセスが可能なクラスターでは、セキュリティグループまたはネットワークアクセスコントロールリスト (ACL) を使用してアクセスを制限することをお勧めします。詳細については、「VPC セキュリティグループ」および「Amazon Redshift クラスターまたは Amazon Redshift Serverless ワークグループのセキュリティグループ通信設定の構成」を参照してください。

デフォルトでの暗号化

2025 年 1 月 10 日以降、Amazon Redshift では、新たに作成されたすべての Amazon Redshift プロビジョニング済みクラスターのデフォルト設定として暗号化が有効になり、データとクラスターのセキュリティが一層強化されます。これは、スナップショットから復元されたクラスターには適用されません。

この変更により、AWS マネジメントコンソール、AWS CLI、または API を使用して KMS キーを指定せずにプロビジョニング済みクラスターを作成する場合、クラスターを復号化する機能は使用できなくなります。クラスターは自動的に AWS 所有のキー で暗号化されます。

この変更を受けて、暗号化されていないクラスターを自動スクリプトで作成している場合や、暗号化されていないクラスターとのデータ共有を活用している場合は、支障が出る可能性があります。シームレスに移行できるように、暗号化されていないクラスターを作成するスクリプトは更新してください。また、暗号化されていないコンシューマークラスターを定期的に新規作成し、データ共有に使用している場合は、設定を見直して、データ共有アクティビティが中断されないように、プロデューサークラスターとコンシューマークラスターの両方を確実に暗号化してください。詳細については、「Amazon Redshift データベース暗号化」を参照してください。

SSL 接続の強制

2025 年 1 月 10 日以降、Amazon Redshift では、新しく作成されたプロビジョニング済みクラスターと復元されたクラスターに接続するクライアントに対して、デフォルトで SSL 接続が適用されます。このデフォルトの変更は、サーバーレスワークグループにも適用されます。

この変更により、新規作成または復元されたすべてのクラスターに `default.redshift-2.0` という名前の新しいデフォルトパラメータグループが導入され、`require_ssl` パラメータはデフォルトで `true` に設定されます。パラメータグループを指定せずに新規作成されたクラスターは、`default.redshift-2.0` パラメータグループを自動的に使用します。Amazon Redshift コンソールを使用してクラスターを作成する場合は、新しい `default.redshift-2.0` パラメータグループが自動的に選択されます。この変更は、`CreateCluster` および `RestoreFromClusterSnapshot` API オペレーション、および対応する SDK コマンドと AWS CLI コマンドにも反映されます。既存またはカスタムのパラメータグループを使用する場合は、パラメー

タグループで指定されている require_ssl 値が引き続き尊重されます。必要に応じて、カスタムパラメータグループの require_ssl 値を変更することもできます。

Amazon Redshift Serverless ユーザーについては、config-parameters の require_ssl のデフォルト値は true に変更されます。require_ssl を false に設定して新しいワークグループを作成するリクエストは、拒否されます。ワークグループの作成後に require_ssl 値を false に変更できます。詳細については、「接続のセキュリティオプションを設定する」を参照してください。

特定のユースケースで必要な場合は、依然として、クラスターまたはワークグループの設定を変更してデフォルトの動作を変更できます。

AWS SDK を使用した Amazon Redshift のコード例

次のコード例は、AWS Software Development Kit (SDK) で Amazon Redshift を使用する方法を示しています。

基本 は、重要なオペレーションをサービス内で実行する方法を示すコード例です。

アクションはより大きなプログラムからのコードの抜粋であり、コンテキスト内で実行する必要があります。アクションは個々のサービス機能を呼び出す方法を示していますが、コンテキスト内のアクションは、関連するシナリオで確認できます。

シナリオは、1 つのサービス内から、または他の AWS のサービスと組み合わせて複数の関数を呼び出し、特定のタスクを実行する方法を示すコード例です。

AWS SDK デベロッパーガイドとコード例の詳細なリストについては、このサービスを AWS SDK で使用する を参照してください。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

コードの例

- AWS SDK を使用した Amazon Redshift の基本的な例
  - Hello Amazon Redshift
  - AWS SDK を使用して Amazon Redshift の基本を学ぶ
  - AWS SDK を使用した Data Service のアクション
    - AWS SDK または CLI で CreateCluster を使用する
    - AWS SDK または CLI で DeleteCluster を使用する
    - AWS SDK または CLI で DescribeClusters を使用する
    - AWS SDK で DescribeStatement を使用する
    - AWS SDK で ExecuteStatement を使用する
    - AWS SDK で GetStatementResult を使用する
    - AWS SDK で ListDatabases を使用する
    - AWS SDK または CLI で ModifyCluster を使用する
- AWS SDK を使用した Amazon Redshift のシナリオ
  - Amazon Redshift アイテムトラッカーの作成
  - CLI を使用したサーバーレスデータウェアハウスの使用開始
  - プロビジョニングされたデータウェアハウスクラスターの使用開始

AWS SDK を使用した Amazon Redshift の基本的な例

次のコード例は、AWS SDK を使用した Amazon Redshift の基本的な使用方法を説明しています。

例

- Hello Amazon Redshift
- AWS SDK を使用して Amazon Redshift の基本を学ぶ
- AWS SDK を使用した Data Service のアクション
  - AWS SDK または CLI で CreateCluster を使用する
  - AWS SDK または CLI で DeleteCluster を使用する
  - AWS SDK または CLI で DescribeClusters を使用する
  - AWS SDK で DescribeStatement を使用する
  - AWS SDK で ExecuteStatement を使用する
  - AWS SDK で GetStatementResult を使用する
  - AWS SDK で ListDatabases を使用する
  - AWS SDK または CLI で ModifyCluster を使用する

Hello Amazon Redshift

以下のコード例は、Amazon Redshift の使用を開始する方法を示しています。

.NET

SDK for .NET (v4)

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
/// <summary>
/// Main method to run the Hello Amazon Redshift example.
/// </summary>
/// <param name="args">Command line arguments (not used).</param>
```

```
   public static async Task Main(string[] args)
   {
       var redshiftClient = new AmazonRedshiftClient();

       Console.WriteLine("Hello, Amazon Redshift! Let's list available
clusters:");

       var clusters = new List<Cluster>();

       try
       {
           // Use pagination to retrieve all clusters.
           var clustersPaginator =
redshiftClient.Paginators.DescribeClusters(new DescribeClustersRequest());

           await foreach (var response in clustersPaginator.Responses)
           {
               if (response.Clusters != null)
                   clusters.AddRange(response.Clusters);
           }

           Console.WriteLine($"{clusters.Count} cluster(s) retrieved.");

           foreach (var cluster in clusters)
           {
               Console.WriteLine($"\t{cluster.ClusterIdentifier} (Status:
{cluster.ClusterStatus})");
           }
       }
       catch (AmazonRedshiftException ex)
       {
           Console.WriteLine($"Couldn't list clusters. Here's why:
{ex.Message}");
       }
       catch (Exception ex)
       {
           Console.WriteLine($"An error occurred: {ex.Message}");
       }
   }
```

- API の詳細については、AWS SDK for .NET API リファレンスの「DescribeClusters」を参照してください。

Go

SDK for Go V2

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
package main

import (
 "context"
 "fmt"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/redshift"
)

// main uses the AWS SDK for Go V2 to create a Redshift client
// and list up to 10 clusters in your account.
// This example uses the default settings specified in your shared credentials
// and config files.
func main() {
 ctx := context.Background()
 sdkConfig, err := config.LoadDefaultConfig(ctx)
 if err != nil {
  fmt.Println("Couldn't load default configuration. Have you set up your AWS
 account?")
  fmt.Println(err)
  return
 }
 redshiftClient := redshift.NewFromConfig(sdkConfig)
 count := 20
 fmt.Printf("Let's list up to %v clusters for your account.\n", count)
 result, err := redshiftClient.DescribeClusters(ctx,
 &redshift.DescribeClustersInput{
  MaxRecords: aws.Int32(int32(count)),
 })
```

```
 if err != nil {
  fmt.Printf("Couldn't list clusters for your account. Here's why: %v\n", err)
  return
 }
 if len(result.Clusters) == 0 {
  fmt.Println("You don't have any clusters!")
  return
 }
 for _, cluster := range result.Clusters {
  fmt.Printf("\t%v : %v\n", *cluster.ClusterIdentifier, *cluster.ClusterStatus)
 }
}
```

- API の詳細については、「AWS SDK for Go API リファレンス」の「DescribeClusters」を参照してください。

Java

SDK for Java 2.x

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.redshift.RedshiftClient;
import
 software.amazon.awssdk.services.redshift.paginators.DescribeClustersIterable;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
```

```
 */
public class HelloRedshift {
    public static void main(String[] args) {
        Region region = Region.US_EAST_1;
        RedshiftClient redshiftClient = RedshiftClient.builder()
            .region(region)
            .build();

        listClustersPaginator(redshiftClient);
    }

    public static void listClustersPaginator(RedshiftClient redshiftClient) {
        DescribeClustersIterable clustersIterable =
 redshiftClient.describeClustersPaginator();
        clustersIterable.stream()
            .flatMap(r -> r.clusters().stream())
            .forEach(cluster -> System.out
                .println(" Cluster identifier: " + cluster.clusterIdentifier() +
 " status = " + cluster.clusterStatus()));
    }
}
```

- API の詳細については、「AWS SDK for Java 2.x API リファレンス」の「DescribeClusters」を参照してください。

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
import boto3


def hello_redshift(redshift_client):
    """
```

```
    Use the AWS SDK for Python (Boto3) to create an Amazon Redshift client and
 list
    the clusters in your account. This list might be empty if you haven't created
    any clusters.
    This example uses the default settings specified in your shared credentials
    and config files.

    :param redshift_client: A Boto3 Redshift Client object.
    """
    print("Hello, Redshift! Let's list your clusters:")
    paginator = redshift_client.get_paginator("describe_clusters")
    clusters = []
    for page in paginator.paginate():
        clusters.extend(page["Clusters"])

    print(f"{len(clusters)} cluster(s) were found.")

    for cluster in clusters:
        print(f"  {cluster['ClusterIdentifier']}")


if __name__ == "__main__":
    hello_redshift(boto3.client("redshift"))
```

- API の詳細については、「AWS SDK for Python (Boto3) API リファレンス」の「DescribeClusters」を参照してください。

AWS SDK デベロッパーガイドとコード例の詳細なリストについては、このサービスを AWS SDK で使用する を参照してください。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK を使用して Amazon Redshift の基本を学ぶ

次のコード例は、以下を実行する方法を示しています。

- Redshift クラスターを作成します。
- クラスター内のデータベースを一覧表示します。
- Movies という名前のテーブルを作成します。
- Movies テーブルにデータを入力します。

- Movies テーブルに対して年に基づくクエリを実行します。
- Redshift クラスターを変更します。
- Amazon Redshift クラスターを削除します。

.NET

SDK for .NET (v4)

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

オペレーションを管理する Redshift ラッパークラスを作成します。

```
/// <summary>
/// Wrapper class for Amazon Redshift operations.
/// </summary>
public class RedshiftWrapper
{
    private readonly IAmazonRedshift _redshiftClient;
    private readonly IAmazonRedshiftDataAPIService _redshiftDataClient;

    /// <summary>
    /// Constructor for RedshiftWrapper.
    /// </summary>
    /// <param name="redshiftClient">Amazon Redshift client.</param>
    /// <param name="redshiftDataClient">Amazon Redshift Data API client.</param>
    public RedshiftWrapper(IAmazonRedshift redshiftClient,
 IAmazonRedshiftDataAPIService redshiftDataClient)
    {
        _redshiftClient = redshiftClient;
        _redshiftDataClient = redshiftDataClient;
    }

    /// <summary>
    /// Create a new Amazon Redshift cluster.
    /// </summary>
    /// <param name="clusterIdentifier">The identifier for the cluster.</param>
    /// <param name="databaseName">The name of the database.</param>
```

```
    /// <param name="masterUsername">The master username.</param>
    /// <param name="masterUserPassword">The master user password.</param>
    /// <param name="nodeType">The node type for the cluster.</param>
    /// <returns>The cluster that was created.</returns>
    public async Task<Cluster> CreateClusterAsync(string clusterIdentifier,
string databaseName,
        string masterUsername, string masterUserPassword, string nodeType =
"ra3.large")
    {
        try
        {
            var request = new CreateClusterRequest
            {
                ClusterIdentifier = clusterIdentifier,
                DBName = databaseName,
                MasterUsername = masterUsername,
                MasterUserPassword = masterUserPassword,
                NodeType = nodeType,
                NumberOfNodes = 1,
                ClusterType = "single-node"
            };

            var response = await _redshiftClient.CreateClusterAsync(request);
            Console.WriteLine($"Created cluster {clusterIdentifier}");
            return response.Cluster;
        }
        catch (ClusterAlreadyExistsException ex)
        {
            Console.WriteLine($"Cluster already exists: {ex.Message}");
            throw;
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Couldn't create cluster. Here's why:
{ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Describe Amazon Redshift clusters.
    /// </summary>
    /// <param name="clusterIdentifier">Optional cluster identifier to describe a
specific cluster.</param>
```

```
   /// <returns>A list of clusters.</returns>
   public async Task<List<Cluster>> DescribeClustersAsync(string?
clusterIdentifier = null)
   {
       try
       {
           var clusters = new List<Cluster>();
           var request = new DescribeClustersRequest();
           if (!string.IsNullOrEmpty(clusterIdentifier))
           {
               request.ClusterIdentifier = clusterIdentifier;
           }

           var clustersPaginator =
_redshiftClient.Paginators.DescribeClusters(request);
           await foreach (var response in clustersPaginator.Responses)
           {
               if (response.Clusters != null)
                   clusters.AddRange(response.Clusters);
           }

           Console.WriteLine($"{clusters.Count} cluster(s) retrieved.");
           foreach (var cluster in clusters)
           {
               Console.WriteLine($"\t{cluster.ClusterIdentifier} (Status:
{cluster.ClusterStatus})");
           }

           return clusters;
       }
       catch (ClusterNotFoundException ex)
       {
           Console.WriteLine($"Cluster {clusterIdentifier} not found:
{ex.Message}");
           throw;
       }
       catch (Exception ex)
       {
           Console.WriteLine($"Couldn't describe clusters. Here's why:
{ex.Message}");
           throw;
       }
   }
```

```
   /// <summary>
   /// Modify an Amazon Redshift cluster.
   /// </summary>
   /// <param name="clusterIdentifier">The identifier for the cluster.</param>
   /// <param name="preferredMaintenanceWindow">The preferred maintenance
window.</param>
   /// <returns>True if successful.</returns>
   public async Task<bool> ModifyClusterAsync(string clusterIdentifier, string
preferredMaintenanceWindow)
   {
       try
       {
           var request = new ModifyClusterRequest
           {
               ClusterIdentifier = clusterIdentifier,
               PreferredMaintenanceWindow = preferredMaintenanceWindow
           };

           var response = await _redshiftClient.ModifyClusterAsync(request);
           Console.WriteLine($"The modified cluster was successfully modified
and has {response.Cluster.PreferredMaintenanceWindow} as the maintenance
window");
           return true;
       }
       catch (ClusterNotFoundException ex)
       {
           Console.WriteLine($"Cluster {clusterIdentifier} not found:
{ex.Message}");
           return false;
       }
       catch (Exception ex)
       {
           Console.WriteLine($"Couldn't modify cluster. Here's why:
{ex.Message}");
           return false;
       }
   }

   /// <summary>
   /// Delete an Amazon Redshift cluster without a final snapshot.
   /// </summary>
   /// <param name="clusterIdentifier">The identifier for the cluster.</param>
   /// <returns>True if successful.</returns>
```

```
   public async Task<bool> DeleteClusterWithoutSnapshotAsync(string
clusterIdentifier)
   {
       try
       {
           var request = new DeleteClusterRequest
           {
               ClusterIdentifier = clusterIdentifier,
               SkipFinalClusterSnapshot = true
           };

           var response = await _redshiftClient.DeleteClusterAsync(request);
           Console.WriteLine($"The {clusterIdentifier} was deleted");
           return true;
       }
       catch (ClusterNotFoundException ex)
       {
           Console.WriteLine($"Cluster not found: {ex.Message}");
           return false;
       }
       catch (Exception ex)
       {
           Console.WriteLine($"Couldn't delete cluster. Here's why:
{ex.Message}");
           return false;
       }
   }

   /// <summary>
   /// List databases in a Redshift cluster.
   /// </summary>
   /// <param name="clusterIdentifier">The cluster identifier.</param>
   /// <param name="dbUser">The database user.</param>
   /// <param name="dbUser">The database name for authentication.</param>
   /// <returns>A list of database names.</returns>
   public async Task<List<string>> ListDatabasesAsync(string clusterIdentifier,
string dbUser, string databaseName)
   {
       try
       {
           var request = new ListDatabasesRequest
           {
               ClusterIdentifier = clusterIdentifier,
               DbUser = dbUser,
```

```
                Database = databaseName
            };

            var response = await _redshiftDataClient.ListDatabasesAsync(request);
            var databases = new List<string>();

            foreach (var database in response.Databases)
            {
                Console.WriteLine($"The database name is : {database}");
                databases.Add(database);
            }

            return databases;
        }
        catch (Amazon.RedshiftDataAPIService.Model.ValidationException ex)
        {
            Console.WriteLine($"Validation error: {ex.Message}");
            throw;
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Couldn't list databases. Here's why:
{ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Create a table in the Redshift database.
    /// </summary>
    /// <param name="clusterIdentifier">The cluster identifier.</param>
    /// <param name="database">The database name.</param>
    /// <param name="dbUser">The database user.</param>
    /// <returns>The statement ID.</returns>
    public async Task<string> CreateTableAsync(string clusterIdentifier, string
database, string dbUser)
    {
        try
        {
            var sqlStatement = @"
                CREATE TABLE Movies (
                    id INTEGER PRIMARY KEY,
                    title VARCHAR(250) NOT NULL,
                    year INTEGER NOT NULL
```

```
                )";

            var request = new ExecuteStatementRequest
            {
                ClusterIdentifier = clusterIdentifier,
                Database = database,
                DbUser = dbUser,
                Sql = sqlStatement
            };

            var response = await
_redshiftDataClient.ExecuteStatementAsync(request);
            await WaitForStatementToCompleteAsync(response.Id);
            Console.WriteLine("Table created: Movies");
            return response.Id;
        }
        catch (Amazon.RedshiftDataAPIService.Model.ValidationException ex)
        {
            Console.WriteLine($"Validation error: {ex.Message}");
            throw;
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Couldn't create table. Here's why:
{ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Insert a record into the Movies table using parameterized query.
    /// </summary>
    /// <param name="clusterIdentifier">The cluster identifier.</param>
    /// <param name="database">The database name.</param>
    /// <param name="dbUser">The database user.</param>
    /// <param name="id">The movie ID.</param>
    /// <param name="title">The movie title.</param>
    /// <param name="year">The movie year.</param>
    /// <returns>The statement ID.</returns>
    public async Task<string> InsertMovieAsync(string clusterIdentifier, string
database, string dbUser,
        int id, string title, int year)
    {
        try
```

```
        {
            var sqlStatement = "INSERT INTO Movies (id, title, year) VALUES
(:id, :title, :year)";

            var request = new ExecuteStatementRequest
            {
                ClusterIdentifier = clusterIdentifier,
                Database = database,
                DbUser = dbUser,
                Sql = sqlStatement,
                Parameters = new List<SqlParameter>
                {
                    new SqlParameter { Name = "id", Value = id.ToString() },
                    new SqlParameter { Name = "title", Value = title },
                    new SqlParameter { Name = "year", Value = year.ToString() }
                }
            };

            var response = await
_redshiftDataClient.ExecuteStatementAsync(request);
            await WaitForStatementToCompleteAsync(response.Id);
            Console.WriteLine($"Inserted: {title} ({year})");
            return response.Id;
        }
        catch (Amazon.RedshiftDataAPIService.Model.ValidationException ex)
        {
            Console.WriteLine($"Validation error: {ex.Message}");
            throw;
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Couldn't insert movie. Here's why:
{ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Query movies by year using parameterized query.
    /// </summary>
    /// <param name="clusterIdentifier">The cluster identifier.</param>
    /// <param name="database">The database name.</param>
    /// <param name="dbUser">The database user.</param>
    /// <param name="year">The year to query.</param>
```

```
    /// <returns>A list of movie titles.</returns>
    public async Task<List<string>> QueryMoviesByYearAsync(string
clusterIdentifier, string database,
        string dbUser, int year)
    {
        try
        {
            var sqlStatement = "SELECT title FROM Movies WHERE year = :year";

            var request = new ExecuteStatementRequest
            {
                ClusterIdentifier = clusterIdentifier,
                Database = database,
                DbUser = dbUser,
                Sql = sqlStatement,
                Parameters = new List<SqlParameter>
                {
                    new SqlParameter { Name = "year", Value = year.ToString() }
                }
            };

            var response = await
_redshiftDataClient.ExecuteStatementAsync(request);
            Console.WriteLine($"The identifier of the statement is
{response.Id}");

            await WaitForStatementToCompleteAsync(response.Id);

            var results = await GetStatementResultAsync(response.Id);
            var movieTitles = new List<string>();

            foreach (var row in results)
            {
                if (row.Count > 0)
                {
                    var title = row[0].StringValue;
                    Console.WriteLine($"The Movie title field is {title}");
                    movieTitles.Add(title);
                }
            }

            return movieTitles;
        }
        catch (Amazon.RedshiftDataAPIService.Model.ValidationException ex)
```

```
        {
            Console.WriteLine($"Validation error: {ex.Message}");
            throw;
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Couldn't query movies. Here's why:
{ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Describe a statement execution.
    /// </summary>
    /// <param name="statementId">The statement ID.</param>
    /// <returns>The statement description.</returns>
    public async Task<DescribeStatementResponse> DescribeStatementAsync(string
statementId)
    {
        try
        {
            var request = new DescribeStatementRequest
            {
                Id = statementId
            };

            var response = await
_redshiftDataClient.DescribeStatementAsync(request);
            return response;
        }
        catch (Amazon.RedshiftDataAPIService.Model.ResourceNotFoundException ex)
        {
            Console.WriteLine($"Statement not found: {ex.Message}");
            throw;
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Couldn't describe statement. Here's why:
{ex.Message}");
            throw;
        }
    }
```

```
    /// <summary>
    /// Get the results of a statement execution.
    /// </summary>
    /// <param name="statementId">The statement ID.</param>
    /// <returns>A list of result rows.</returns>
    public async Task<List<List<Field>>> GetStatementResultAsync(string
statementId)
    {
        try
        {
            var request = new GetStatementResultRequest
            {
                Id = statementId
            };

            var response = await
_redshiftDataClient.GetStatementResultAsync(request);
            return response.Records;
        }
        catch (Amazon.RedshiftDataAPIService.Model.ResourceNotFoundException ex)
        {
            Console.WriteLine($"Statement not found: {ex.Message}");
            throw;
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Couldn't get statement result. Here's why:
{ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Wait for a statement to complete execution.
    /// </summary>
    /// <param name="statementId">The statement ID.</param>
    /// <returns>A task representing the asynchronous operation.</returns>
    private async Task WaitForStatementToCompleteAsync(string statementId)
    {
        var status = StatusString.SUBMITTED;
        DescribeStatementResponse? response = null;

        while (status == StatusString.SUBMITTED || status == StatusString.PICKED
|| status == StatusString.STARTED)
```

```
        {
            await Task.Delay(1000); // Wait 1 second
            response = await DescribeStatementAsync(statementId);
            status = response.Status;
            Console.WriteLine($"...{status}");
        }

        if (status == StatusString.FINISHED)
        {
            Console.WriteLine("The statement is finished!");
        }
        else
        {
            var errorMessage = response?.Error ?? "Unknown error";
            Console.WriteLine($"The statement failed with status: {status}");
            Console.WriteLine($"Error message: {errorMessage}");
        }
    }

    /// <summary>
    /// Wait for a cluster to become available.
    /// </summary>
    /// <param name="clusterIdentifier">The cluster identifier.</param>
    /// <returns>A task representing the asynchronous operation.</returns>
    public async Task WaitForClusterAvailableAsync(string clusterIdentifier)
    {
        Console.WriteLine($"Wait until {clusterIdentifier} is available. This may
take a few minutes.");

        var startTime = DateTime.Now;
        var clusters = await DescribeClustersAsync(clusterIdentifier);

        while (clusters[0].ClusterStatus != "available")
        {
            var elapsed = DateTime.Now - startTime;
            Console.WriteLine($"Elapsed Time: {elapsed:mm\\:ss} - Waiting for
cluster...");

            await Task.Delay(5000); // Wait 5 seconds
            clusters = await DescribeClustersAsync(clusterIdentifier);
        }

        var totalElapsed = DateTime.Now - startTime;
```

```
        Console.WriteLine($"Cluster is available! Total Elapsed Time:
 {totalElapsed:mm\\:ss}");
    }
}
```

Redshift の基本機能を示すインタラクティブなシナリオを実行します。

```
/// <summary>
/// Amazon Redshift Getting Started Scenario.
/// </summary>
public class RedshiftBasics
{
    public static bool IsInteractive = true;
    public static RedshiftWrapper? Wrapper = null;
    public static ILogger logger = null!;
    private static readonly string _moviesFilePath = "../../../../../../
resources/sample_files/movies.json";

    /// <summary>
    /// Main method for the Amazon Redshift Getting Started scenario.
    /// </summary>
    /// <param name="args">Command line arguments.</param>
    public static async Task Main(string[] args)
    {
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonRedshift>()
                    .AddAWSService<IAmazonRedshiftDataAPIService>()
                    .AddTransient<RedshiftWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<RedshiftBasics>();

        Wrapper = host.Services.GetRequiredService<RedshiftWrapper>();

        await RunScenarioAsync();
    }

    /// <summary>
    /// Run the complete Amazon Redshift scenario.
```

```
    /// </summary>
    public static async Task RunScenarioAsync()
    {
        // Set all variables to default values
        string userName = "awsuser";
        string userPassword = "AwsUser1000";
        string clusterIdentifier = "redshift-cluster-movies";
        var databaseName = "dev";
        int recordCount = 50;
        int year = 2013;
        try
        {
            Console.WriteLine(

"================================================================================");
            Console.WriteLine("Welcome to the Amazon Redshift SDK Getting Started
scenario.");
            Console.WriteLine(
                "This .NET program demonstrates how to interact with Amazon
Redshift by using the AWS SDK for .NET.");
            Console.WriteLine("Let's get started...");
            Console.WriteLine(

"================================================================================");

            // Step 1: Get user credentials (if interactive)
            if (IsInteractive)
            {
                Console.WriteLine("Please enter a user name for the cluster
(default is awsuser):");
                var userInput = Console.ReadLine();
                if (!string.IsNullOrEmpty(userInput))
                    userName = userInput;


Console.WriteLine("================================================================================
                Console.WriteLine("Please enter a user password for the cluster
(default is AwsUser1000):");
                var passwordInput = Console.ReadLine();
                if (!string.IsNullOrEmpty(passwordInput))
                    userPassword = passwordInput;


Console.WriteLine("================================================================================
```

```
                // Step 2: Get cluster identifier
                Console.WriteLine("Enter a cluster id value (default is redshift-
cluster-movies):");
                var clusterInput = Console.ReadLine();
                if (!string.IsNullOrEmpty(clusterInput))
                    clusterIdentifier = clusterInput;
            }
            else
            {
                Console.WriteLine($"Using default values: userName={userName},
 clusterIdentifier={clusterIdentifier}");
            }

            // Step 3: Create Redshift cluster
            await Wrapper!.CreateClusterAsync(clusterIdentifier, databaseName,
 userName, userPassword);

 Console.WriteLine("=====================================================================

            // Step 4: Wait for cluster to become available

 Console.WriteLine("=====================================================================
            await Wrapper.WaitForClusterAvailableAsync(clusterIdentifier);

 Console.WriteLine("=====================================================================

            // Step 5: List databases

 Console.WriteLine("=====================================================================
            Console.WriteLine($" When you created {clusterIdentifier}, the dev
 database is created by default and used in this scenario.");
            Console.WriteLine(" To create a custom database, you need to have a
 CREATEDB privilege.");
            Console.WriteLine(" For more information, see the documentation here:
 https://docs.aws.amazon.com/redshift/latest/dg/r_CREATE_DATABASE.html.");
            if (IsInteractive)
            {
                Console.WriteLine("Press Enter to continue...");
                Console.ReadLine();
            }

 Console.WriteLine("=====================================================================
```

```
Console.WriteLine("=============================================================================
            Console.WriteLine($"List databases in {clusterIdentifier}");
            if (IsInteractive)
            {
                Console.WriteLine("Press Enter to continue...");
                Console.ReadLine();
            }
            await Wrapper.ListDatabasesAsync(clusterIdentifier, userName,
databaseName);

Console.WriteLine("=============================================================================

            // Step 6: Create Movies table

Console.WriteLine("=============================================================================
            Console.WriteLine("Now you will create a table named Movies.");
            if (IsInteractive)
            {
                Console.WriteLine("Press Enter to continue...");
                Console.ReadLine();
            }
            await Wrapper.CreateTableAsync(clusterIdentifier, databaseName,
userName);

Console.WriteLine("=============================================================================

            // Step 7: Populate the Movies table

Console.WriteLine("=============================================================================
            Console.WriteLine("Populate the Movies table using the Movies.json
file.");

            if (IsInteractive)
            {
                Console.WriteLine("Specify the number of records you would like
to add to the Movies Table.");
                Console.WriteLine("Please enter a value between 50 and 200.");
                Console.Write("Enter a value: ");

                var recordCountInput = Console.ReadLine();
                if (int.TryParse(recordCountInput, out var inputCount) &&
inputCount is >= 50 and <= 200)
                {
```

```
                    recordCount = inputCount;
                }
                else
                {
                    Console.WriteLine($"Invalid input. Using default value of
{recordCount}.");
                }
            }
            else
            {
                Console.WriteLine($"Using default record count: {recordCount}");
            }

            await PopulateMoviesTableAsync(clusterIdentifier, databaseName,
userName, recordCount);
            Console.WriteLine($"{recordCount} records were added to the Movies
table.");

Console.WriteLine("================================================================================

            // Step 8 & 9: Query movies by year

Console.WriteLine("================================================================================
            Console.WriteLine("Query the Movies table by year. Enter a value
between 2012-2014.");

            if (IsInteractive)
            {
                Console.Write("Enter a year: ");
                var yearInput = Console.ReadLine();
                if (int.TryParse(yearInput, out var inputYear) && inputYear is >=
2012 and <= 2014)
                {
                    year = inputYear;
                }
                else
                {
                    Console.WriteLine($"Invalid input. Using default value of
{year}.");
                }
            }
            else
            {
                Console.WriteLine($"Using default year: {year}");
```

```
            }

            await Wrapper.QueryMoviesByYearAsync(clusterIdentifier, databaseName,
 userName, year);

 Console.WriteLine("=======================================================================

            // Step 10: Modify the cluster

 Console.WriteLine("=======================================================================
            Console.WriteLine("Now you will modify the Redshift cluster.");
            if (IsInteractive)
            {
                Console.WriteLine("Press Enter to continue...");
                Console.ReadLine();
            }
            await Wrapper.ModifyClusterAsync(clusterIdentifier, "wed:07:30-
wed:08:00");

 Console.WriteLine("=======================================================================

            // Step 11 & 12: Delete cluster confirmation

 Console.WriteLine("=======================================================================
            if (IsInteractive)
            {
                Console.WriteLine("Would you like to delete the Amazon Redshift
 cluster? (y/n)");
                var deleteResponse = Console.ReadLine();
                if (deleteResponse?.ToLower() == "y")
                {
                    await
 Wrapper.DeleteClusterWithoutSnapshotAsync(clusterIdentifier);
                }
            }
            else
            {
                Console.WriteLine("Deleting the Amazon Redshift cluster...");
                await
 Wrapper.DeleteClusterWithoutSnapshotAsync(clusterIdentifier);
            }

 Console.WriteLine("=======================================================================
```

```
Console.WriteLine("========================================================================
            Console.WriteLine("This concludes the Amazon Redshift SDK Getting
Started scenario.");

Console.WriteLine("========================================================================
        }
        catch (Exception ex)
        {
            Console.WriteLine($"An error occurred during the scenario:
{ex.Message}");
            Console.WriteLine("Deleting the Amazon Redshift cluster...");
            await Wrapper!.DeleteClusterWithoutSnapshotAsync(clusterIdentifier);
            throw;
        }
    }

    /// <summary>
    /// Populate the Movies table with data from the JSON file.
    /// </summary>
    /// <param name="clusterIdentifier">The cluster identifier.</param>
    /// <param name="database">The database name.</param>
    /// <param name="dbUser">The database user.</param>
    /// <param name="recordCount">Number of records to insert.</param>
    private static async Task PopulateMoviesTableAsync(string clusterIdentifier,
string database, string dbUser, int recordCount)
    {
        if (!File.Exists(_moviesFilePath))
        {
            throw new FileNotFoundException($"Required movies data file not found
at: {_moviesFilePath}");
        }

        var jsonContent = await File.ReadAllTextAsync(_moviesFilePath);
        var options = new JsonSerializerOptions
        {
            PropertyNameCaseInsensitive = true
        };
        var movies = JsonSerializer.Deserialize<List<Movie>>(jsonContent,
options);

        if (movies == null || movies.Count == 0)
        {
```

```
            throw new InvalidOperationException("Failed to parse movies JSON file
 or file is empty.");
        }

        var insertCount = Math.Min(recordCount, movies.Count);

        for (int i = 0; i < insertCount; i++)
        {
            var movie = movies[i];
            await Wrapper!.InsertMovieAsync(clusterIdentifier, database, dbUser,
 i, movie.Title, movie.Year);
        }
    }

    /// <summary>
    /// Movie data model.
    /// </summary>
    private class Movie
    {
        public string Title { get; set; } = string.Empty;
        public int Year { get; set; }
    }
}
```

- API の詳細については、「AWS SDK for .NET API リファレンス」の以下のトピックを参照してください。
  - CreateCluste:
  - DescribeClusters
  - DescribeStatement
  - ExecuteStatement
  - GetStatementResult
  - ListDatabasesPaginator
  - ModifyCluster

Go

SDK for Go V2

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
package scenarios

import (
 "context"
 "encoding/json"
 "errors"
 "fmt"
 "log"
 "math/rand"
 "strings"
 "time"

 "github.com/aws/aws-sdk-go-v2/aws"
 redshift_types "github.com/aws/aws-sdk-go-v2/service/redshift/types"
 redshiftdata_types "github.com/aws/aws-sdk-go-v2/service/redshiftdata/types"
 "github.com/aws/aws-sdk-go-v2/service/secretsmanager"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/redshift/actions"

 "github.com/aws/aws-sdk-go-v2/service/redshift"
 "github.com/aws/aws-sdk-go-v2/service/redshiftdata"
)

// IScenarioHelper abstracts input and wait functions from a scenario so that
 they
// can be mocked for unit testing.
type IScenarioHelper interface {
 GetName() string
}

const rMax = 100000
```

```
type ScenarioHelper struct {
 Prefix string
 Random *rand.Rand
}

// GetName returns a unique name formed of a prefix and a random number.
func (helper ScenarioHelper) GetName() string {
 return fmt.Sprintf("%v%v", helper.Prefix, helper.Random.Intn(rMax))
}

// RedshiftBasicsScenario separates the steps of this scenario into individual
 functions so that
// they are simpler to read and understand.
type RedshiftBasicsScenario struct {
 sdkConfig         aws.Config
 helper            IScenarioHelper
 questioner        demotools.IQuestioner
 pauser            demotools.IPausable
 filesystem        demotools.IFileSystem
 redshiftActor     *actions.RedshiftActions
 redshiftDataActor *actions.RedshiftDataActions
 secretsmanager    *SecretsManager
}

// SecretsManager is used to retrieve username and password information from a
 secure service.
type SecretsManager struct {
 SecretsManagerClient *secretsmanager.Client
}

// RedshiftBasics constructs a new Redshift Basics runner.
func RedshiftBasics(sdkConfig aws.Config, questioner demotools.IQuestioner,
 pauser demotools.IPausable, filesystem demotools.IFileSystem, helper
 IScenarioHelper) RedshiftBasicsScenario {
 scenario := RedshiftBasicsScenario{
  sdkConfig:         sdkConfig,
  helper:            helper,
  questioner:        questioner,
  pauser:            pauser,
  filesystem:        filesystem,
  secretsmanager:    &SecretsManager{SecretsManagerClient:
 secretsmanager.NewFromConfig(sdkConfig)},
```

```
  redshiftActor:     &actions.RedshiftActions{RedshiftClient:
 redshift.NewFromConfig(sdkConfig)},
  redshiftDataActor: &actions.RedshiftDataActions{RedshiftDataClient:
 redshiftdata.NewFromConfig(sdkConfig)},
 }
 return scenario
}


// Movie makes it easier to use Movie objects given in json format.
type Movie struct {
 ID    int    `json:"id"`
 Title string `json:"title"`
 Year  int    `json:"year"`
}


// User makes it easier to get the User data back from SecretsManager and use it
 later.
type User struct {
 Username string `json:"userName"`
 Password string `json:"userPassword"`
}

// Run runs the RedshiftBasics interactive example that shows you how to use
 Amazon
// Redshift and how to interact with its common endpoints.
//
// 0. Retrieve username and password information to access Redshift.
// 1. Create a cluster.
// 2. Wait for the cluster to become available.
// 3. List the available databases in the region.
// 4. Create a table named "Movies" in the "dev" database.
// 5. Populate the movies table from the "movies.json" file.
// 6. Query the movies table by year.
// 7. Modify the cluster's maintenance window.
// 8. Optionally clean up all resources created during this demo.
//
// This example creates an Amazon Redshift service client from the specified
 sdkConfig so that
// you can replace it with a mocked or stubbed config for unit testing.
//
// It uses a questioner from the `demotools` package to get input during the
 example.
```

```
// This package can be found in the ..\..\demotools folder of this repo.
func (runner *RedshiftBasicsScenario) Run(ctx context.Context) {

 user := User{}
 secretId := "s3express/basics/secrets"
 clusterId := "demo-cluster-1"
 maintenanceWindow := "wed:07:30-wed:08:00"
 databaseName := "dev"
 tableName := "Movies"
 fileName := "Movies.json"
 nodeType := "ra3.xlplus"
 clusterType := "single-node"

 defer func() {
  if r := recover(); r != nil {
   log.Println("Something went wrong with the demo.")
   _, isMock := runner.questioner.(*demotools.MockQuestioner)
   if isMock || runner.questioner.AskBool("Do you want to see the full error
 message (y/n)?", "y") {
    log.Println(r)
   }
   runner.cleanUpResources(ctx, clusterId, databaseName, tableName,
 user.Username, runner.questioner)
  }
 }()

 // Retrieve the userName and userPassword from SecretsManager
 output, err := runner.secretsmanager.SecretsManagerClient.GetSecretValue(ctx,
 &secretsmanager.GetSecretValueInput{
  SecretId: aws.String(secretId),
 })
 if err != nil {
  log.Printf("There was a problem getting the secret value: %s", err)
  log.Printf("Please make sure to create a secret named 's3express/basics/
secrets' with keys of 'userName' and 'userPassword'.")
  panic(err)
 }

 err = json.Unmarshal([]byte(*output.SecretString), &user)
 if err != nil {
  log.Printf("There was a problem parsing the secret value from JSON: %s", err)
  panic(err)
 }
```

```
// Create the Redshift cluster
_, err = runner.redshiftActor.CreateCluster(ctx, clusterId, user.Username,
user.Password, nodeType, clusterType, true)
if err != nil {
 var clusterAlreadyExistsFault *redshift_types.ClusterAlreadyExistsFault
 if errors.As(err, &clusterAlreadyExistsFault) {
  log.Println("Cluster already exists. Continuing.")
 } else {
  log.Println("Error creating cluster.")
  panic(err)
 }
}

// Wait for the cluster to become available
waiter :=
redshift.NewClusterAvailableWaiter(runner.redshiftActor.RedshiftClient)
err = waiter.Wait(ctx, &redshift.DescribeClustersInput{
 ClusterIdentifier: aws.String(clusterId),
}, 5*time.Minute)
if err != nil {
 log.Println("An error occurred waiting for the cluster.")
 panic(err)
}

// Get some info about the cluster
describeOutput, err := runner.redshiftActor.DescribeClusters(ctx, clusterId)
if err != nil {
 log.Println("Something went wrong trying to get information about the
cluster.")
 panic(err)
}
log.Println("Here's some information about the cluster.")
log.Printf("The cluster's status is %s",
*describeOutput.Clusters[0].ClusterStatus)
log.Printf("The cluster was created at %s",
*describeOutput.Clusters[0].ClusterCreateTime)

// List databases
log.Println("List databases in", clusterId)
runner.questioner.Ask("Press Enter to continue...")
err = runner.redshiftDataActor.ListDatabases(ctx, clusterId, databaseName,
user.Username)
if err != nil {
 log.Printf("Failed to list databases: %v\n", err)
```

```
 panic(err)
}

// Create the "Movies" table
log.Println("Now you will create a table named " + tableName + ".")
runner.questioner.Ask("Press Enter to continue...")
err = nil
result, err := runner.redshiftDataActor.CreateTable(ctx, clusterId,
databaseName, tableName, user.Username, runner.pauser, []string{"title
VARCHAR(256)", "year INT"})
if err != nil {
 log.Printf("Failed to create table: %v\n", err)
 panic(err)
}

describeInput := redshiftdata.DescribeStatementInput{
 Id: result.Id,
}
query := actions.RedshiftQuery{
 Context: ctx,
 Input:   describeInput,
 Result:  result,
}
err = runner.redshiftDataActor.WaitForQueryStatus(query, runner.pauser, true)
if err != nil {
 log.Printf("Failed to execute query: %v\n", err)
 panic(err)
}
log.Printf("Successfully executed query\n")

// Populate the "Movies" table
runner.PopulateMoviesTable(ctx, clusterId, databaseName, tableName,
user.Username, fileName)

// Query the "Movies" table by year
log.Println("Query the Movies table by year.")
year := runner.questioner.AskInt(
 fmt.Sprintf("Enter a value between %v and %v:", 2012, 2014),
 demotools.InIntRange{Lower: 2012, Upper: 2014})
runner.QueryMoviesByYear(ctx, clusterId, databaseName, tableName, user.Username,
year)

// Modify the cluster's maintenance window
runner.redshiftActor.ModifyCluster(ctx, clusterId, maintenanceWindow)
```

```
 // Delete the Redshift cluster if confirmed
 runner.cleanUpResources(ctx, clusterId, databaseName, tableName, user.Username,
 runner.questioner)

 log.Println("Thanks for watching!")
}

// cleanUpResources asks the user if they would like to delete each resource
 created during the scenario, from most
// impactful to least impactful. If any choice to delete is made, further
 deletion attempts are skipped.
func (runner *RedshiftBasicsScenario) cleanUpResources(ctx context.Context,
 clusterId string, databaseName string, tableName string, userName string,
 questioner demotools.IQuestioner) {
 deleted := false
 var err error = nil
 if questioner.AskBool("Do you want to delete the entire cluster? This will clean
 up all resources. (y/n)", "y") {
  deleted, err = runner.redshiftActor.DeleteCluster(ctx, clusterId)
  if err != nil {
   log.Printf("Error deleting cluster: %v", err)
  }
 }
 if !deleted && questioner.AskBool("Do you want to delete the dev table? This
 will clean up all inserted records but keep your cluster intact. (y/n)", "y") {
  deleted, err = runner.redshiftDataActor.DeleteTable(ctx, clusterId,
 databaseName, tableName, userName)
  if err != nil {
   log.Printf("Error deleting movies table: %v", err)
  }
 }
 if !deleted && questioner.AskBool("Do you want to delete all rows in the Movies
 table? This will clean up all inserted records but keep your cluster and table
 intact. (y/n)", "y") {
  deleted, err = runner.redshiftDataActor.DeleteDataRows(ctx, clusterId,
 databaseName, tableName, userName, runner.pauser)
  if err != nil {
   log.Printf("Error deleting data rows: %v", err)
  }
 }
 if !deleted {
  log.Print("Please manually delete any unwanted resources.")
 }
```

```
}


// loadMoviesFromJSON takes the <fileName> file and populates a slice of Movie
 objects.
func (runner *RedshiftBasicsScenario) loadMoviesFromJSON(fileName string,
 filesystem demotools.IFileSystem) ([]Movie, error) {
 file, err := filesystem.OpenFile("../../resources/sample_files/" + fileName)
 if err != nil {
  return nil, err
 }
 defer filesystem.CloseFile(file)

 var movies []Movie
 err = json.NewDecoder(file).Decode(&movies)
 if err != nil {
  return nil, err
 }

 return movies, nil
}



// PopulateMoviesTable reads data from the <fileName> file and inserts records
 into the "Movies" table.
func (runner *RedshiftBasicsScenario) PopulateMoviesTable(ctx context.Context,
 clusterId string, databaseName string, tableName string, userName string,
 fileName string) {
 log.Println("Populate the " + tableName + " table using the " + fileName + "
 file.")
 numRecords := runner.questioner.AskInt(
  fmt.Sprintf("Enter a value between %v and %v:", 10, 100),
  demotools.InIntRange{Lower: 10, Upper: 100})

 movies, err := runner.loadMoviesFromJSON(fileName, runner.filesystem)
 if err != nil {
  log.Printf("Failed to load movies from JSON: %v\n", err)
  panic(err)
 }

 var sqlStatements []string

 for i, movie := range movies {
```

```
  if i >= numRecords {
   break
  }

  sqlStatement := fmt.Sprintf(`INSERT INTO %s (title, year) VALUES ('%s', %d);`,
   tableName,
   strings.Replace(movie.Title, "'", "''", -1), // Double any single quotes to
 escape them
   movie.Year)

  sqlStatements = append(sqlStatements, sqlStatement)
 }

 input := &redshiftdata.BatchExecuteStatementInput{
  ClusterIdentifier: aws.String(clusterId),
  Database:          aws.String(databaseName),
  DbUser:            aws.String(userName),
  Sqls:              sqlStatements,
 }

 result, err := runner.redshiftDataActor.ExecuteBatchStatement(ctx, *input)
 if err != nil {
  log.Printf("Failed to execute batch statement: %v\n", err)
  panic(err)
 }

 describeInput := redshiftdata.DescribeStatementInput{
  Id: result.Id,
 }

 query := actions.RedshiftQuery{
  Context: ctx,
  Result:  result,
  Input:   describeInput,
 }
 err = runner.redshiftDataActor.WaitForQueryStatus(query, runner.pauser, true)
 if err != nil {
  log.Printf("Failed to execute batch insert query: %v\n", err)
  return
 }
 log.Printf("Successfully executed batch statement\n")

 log.Printf("%d records were added to the Movies table.\n", numRecords)
}
```

```
// QueryMoviesByYear retrieves only movies from the "Movies" table which match
 the given year.
func (runner *RedshiftBasicsScenario) QueryMoviesByYear(ctx context.Context,
 clusterId string, databaseName string, tableName string, userName string, year
 int) {

 sqlStatement := fmt.Sprintf(`SELECT title FROM %s WHERE year = %d;`, tableName,
 year)

 input := &redshiftdata.ExecuteStatementInput{
  ClusterIdentifier: aws.String(clusterId),
  Database:          aws.String(databaseName),
  DbUser:            aws.String(userName),
  Sql:               aws.String(sqlStatement),
 }

 result, err := runner.redshiftDataActor.ExecuteStatement(ctx, *input)
 if err != nil {
  log.Printf("Failed to query movies: %v\n", err)
  panic(err)
 }

 log.Println("The identifier of the statement is ", *result.Id)

 describeInput := redshiftdata.DescribeStatementInput{
  Id: result.Id,
 }

 query := actions.RedshiftQuery{
  Context: ctx,
  Input:   describeInput,
  Result:  result,
 }
 err = runner.redshiftDataActor.WaitForQueryStatus(query, runner.pauser, true)
 if err != nil {
  log.Printf("Failed to execute query: %v\n", err)
  panic(err)
 }
 log.Printf("Successfully executed query\n")
```

```
 getResultOutput, err := runner.redshiftDataActor.GetStatementResult(ctx,
 *result.Id)
 if err != nil {
  log.Printf("Failed to query movies: %v\n", err)
  panic(err)
 }
 for _, row := range getResultOutput.Records {
  for _, col := range row {
   title, ok := col.(*redshiftdata_types.FieldMemberStringValue)
   if !ok {
    log.Println("Failed to parse the field")
   } else {
    log.Printf("The Movie title field is %s\n", title.Value)
   }
  }
 }
}
```

- API の詳細については、「AWS SDK for Go API リファレンス」の以下のトピックを参照してください。
  - CreateCluste:
  - DescribeClusters
  - DescribeStatement
  - ExecuteStatement
  - GetStatementResult
  - ListDatabasesPaginator
  - ModifyCluster

Java

SDK for Java 2.x

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

Amazon Redshift 機能を示すインタラクティブなシナリオを実行します。

```
import com.example.redshift.User;
import com.google.gson.Gson;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.redshift.model.ClusterAlreadyExistsException;
import software.amazon.awssdk.services.redshift.model.CreateClusterResponse;
import software.amazon.awssdk.services.redshift.model.DeleteClusterResponse;
import software.amazon.awssdk.services.redshift.model.ModifyClusterResponse;
import software.amazon.awssdk.services.redshift.model.RedshiftException;
import
 software.amazon.awssdk.services.redshiftdata.model.ExecuteStatementResponse;
import software.amazon.awssdk.services.redshiftdata.model.RedshiftDataException;
import java.util.Scanner;
import java.util.concurrent.CompletableFuture;
import software.amazon.awssdk.services.secretsmanager.SecretsManagerClient;
import
 software.amazon.awssdk.services.secretsmanager.model.GetSecretValueRequest;
import
 software.amazon.awssdk.services.secretsmanager.model.GetSecretValueResponse;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 *
```

```
 *
 *  This example requires an AWS Secrets Manager secret that contains the
 *  database credentials. If you do not create a
 *  secret that specifies user name and password, this example will not work. For
 details, see:
 *
 *  https://docs.aws.amazon.com/secretsmanager/latest/userguide/integrating_how-
services-use-secrets_RS.html
 *
 This Java example performs these tasks:
 *
 * 1. Prompts the user for a unique cluster ID or use the default value.
 * 2. Creates a Redshift cluster with the specified or default cluster Id value.
 * 3. Waits until the Redshift cluster is available for use.
 * 4. Lists all databases using a pagination API call.
 * 5. Creates a table named "Movies" with fields ID, title, and year.
 * 6. Inserts a specified number of records into the "Movies" table by reading
 the Movies JSON file.
 * 7. Prompts the user for a movie release year.
 * 8. Runs a SQL query to retrieve movies released in the specified year.
 * 9. Modifies the Redshift cluster.
 * 10. Prompts the user for confirmation to delete the Redshift cluster.
 * 11. If confirmed, deletes the specified Redshift cluster.
 */

public class RedshiftScenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");
    private static final Logger logger =
 LoggerFactory.getLogger(RedshiftScenario.class);

    static RedshiftActions redshiftActions = new RedshiftActions();
    public static void main(String[] args) throws Exception {
        final String usage = """

            Usage:
                <jsonFilePath> <secretName>\s

            Where:
                jsonFilePath - The path to the Movies JSON file (you can locate
 that file in ../../../resources/sample_files/movies.json)
                secretName - The name of the secret that belongs to Secret
 Manager that stores the user name and password used in this scenario.
            """;
```

```
        if (args.length != 2) {
            logger.info(usage);
            return;
        }

        String jsonFilePath = args[0];
        String secretName = args[1];
        Scanner scanner = new Scanner(System.in);
        logger.info(DASHES);
        logger.info("Welcome to the Amazon Redshift SDK Basics scenario.");
        logger.info("""
            This Java program demonstrates how to interact with Amazon Redshift
 by using the AWS SDK for Java (v2).\s
            Amazon Redshift is a fully managed, petabyte-scale data warehouse
 service hosted in the cloud.

            The program's primary functionalities include cluster creation,
 verification of cluster readiness,\s
            list databases, table creation, data population within the table, and
 execution of SQL statements.
            Furthermore, it demonstrates the process of querying data from the
 Movie table.\s

            Upon completion of the program, all AWS resources are cleaned up.
            """);

        logger.info("Lets get started...");
        logger.info("""
            First, we will retrieve the user name and password from Secrets
 Manager.

            Using Amazon Secrets Manager to store Redshift credentials provides
 several security benefits.
            It allows you to securely store and manage sensitive information,
 such as passwords, API keys, and
            database credentials, without embedding them directly in your
 application code.

            More information can be found here:

            https://docs.aws.amazon.com/secretsmanager/latest/userguide/
integrating_how-services-use-secrets_RS.html
            """);
```

```
        Gson gson = new Gson();
        User user = gson.fromJson(String.valueOf(getSecretValues(secretName)),
User.class);
        waitForInputToContinue(scanner);
        logger.info(DASHES);

        try {
            runScenario(user, scanner, jsonFilePath);
        } catch (RuntimeException e) {
            e.printStackTrace();
        } catch (Throwable e) {
            throw new RuntimeException(e);
        }
    }

    private static void runScenario(User user, Scanner scanner,  String
jsonFilePath) throws Throwable {
        String databaseName = "dev";
        System.out.println(DASHES);
        logger.info("Create a Redshift Cluster");
        logger.info("A Redshift cluster refers to the collection of computing
resources and storage that work together to process and analyze large volumes of
data.");
        logger.info("Enter a cluster id value or accept the default by hitting
Enter (default is redshift-cluster-movies): ");
        String userClusterId = scanner.nextLine();
        String clusterId = userClusterId.isEmpty() ? "redshift-cluster-movies" :
userClusterId;
        try {
            CompletableFuture<CreateClusterResponse> future =
redshiftActions.createClusterAsync(clusterId, user.getUserName(),
user.getUserPassword());
            CreateClusterResponse response = future.join();
            logger.info("Cluster successfully created. Cluster Identifier {} ",
response.cluster().clusterIdentifier());

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof ClusterAlreadyExistsException) {
                logger.info("The Cluster {} already exists. Moving on...",
clusterId);
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
```

```
        }
        logger.info(DASHES);

        logger.info(DASHES);
        logger.info("Wait until {} is available.", clusterId);
        waitForInputToContinue(scanner);
        try {
            CompletableFuture<Void> future =
 redshiftActions.waitForClusterReadyAsync(clusterId);
            future.join();
            logger.info("Cluster is ready!");

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof RedshiftException redshiftEx) {
                logger.info("Redshift error occurred: Error message: {}, Error
 code {}", redshiftEx.getMessage(), redshiftEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
            throw cause;
        }
        logger.info(DASHES);

        logger.info(DASHES);
        String databaseInfo = """
            When you created $clusteridD, the dev database is created by default
 and used in this scenario.\s

            To create a custom database, you need to have a CREATEDB privilege.\s
            For more information, see the documentation here: https://
docs.aws.amazon.com/redshift/latest/dg/r_CREATE_DATABASE.html.
           """.replace("$clusteridD", clusterId);

        logger.info(databaseInfo);
        waitForInputToContinue(scanner);
        logger.info(DASHES);

        logger.info(DASHES);
        logger.info("List databases in {} ",clusterId);
        waitForInputToContinue(scanner);
        try {
            CompletableFuture<Void> future =
 redshiftActions.listAllDatabasesAsync(clusterId, user.getUserName(), "dev");
```

```
            future.join();
            logger.info("Databases listed successfully.");

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof RedshiftDataException redshiftEx) {
                logger.error("Redshift Data error occurred: {} Error code: {}",
redshiftEx.getMessage(), redshiftEx.awsErrorDetails().errorCode());
            } else {
                logger.error("An unexpected error occurred: {}",
rt.getMessage());
            }
            throw cause;
        }
        logger.info(DASHES);

        logger.info(DASHES);
        logger.info("Now you will create a table named Movies.");
        waitForInputToContinue(scanner);
        try {
            CompletableFuture<ExecuteStatementResponse> future =
redshiftActions.createTableAsync(clusterId, databaseName, user.getUserName());
            future.join();

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof RedshiftDataException redshiftEx) {
                logger.info("Redshift Data error occurred: {} Error code: {}",
redshiftEx.getMessage(), redshiftEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: {}", rt.getMessage());
            }
            throw cause;
        }
        logger.info(DASHES);

        logger.info(DASHES);
        logger.info("Populate the Movies table using the Movies.json file.");
        logger.info("Specify the number of records you would like to add to the
Movies Table.");
        logger.info("Please enter a value between 50 and 200.");
        int numRecords;
        do {
            logger.info("Enter a value: ");
```

```
            while (!scanner.hasNextInt()) {
                logger.info("Invalid input. Please enter a value between 50 and
200.");
                logger.info("Enter a year: ");
                scanner.next();
            }
            numRecords = scanner.nextInt();
        } while (numRecords < 50 || numRecords > 200);
        try {
            redshiftActions.popTableAsync(clusterId, databaseName,
user.getUserName(), jsonFilePath, numRecords).join();  // Wait for the operation
to complete
        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof RedshiftDataException redshiftEx) {
                logger.info("Redshift Data error occurred: {} Error code: {}",
redshiftEx.getMessage(), redshiftEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: {}", rt.getMessage());
            }
            throw cause;
        }
        waitForInputToContinue(scanner);
        logger.info(DASHES);

        logger.info(DASHES);
        logger.info("Query the Movies table by year. Enter a value between
2012-2014.");
        int movieYear;
        do {
            logger.info("Enter a year: ");
            while (!scanner.hasNextInt()) {
                logger.info("Invalid input. Please enter a valid year between
2012 and 2014.");
                logger.info("Enter a year: ");
                scanner.next();
            }
            movieYear = scanner.nextInt();
            scanner.nextLine();
        } while (movieYear < 2012 || movieYear > 2014);

        String id;
        try {
```

```
            CompletableFuture<String> future =
redshiftActions.queryMoviesByYearAsync(databaseName, user.getUserName(),
movieYear, clusterId);
            id = future.join();

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof RedshiftDataException redshiftEx) {
                logger.info("Redshift Data error occurred: {} Error code: {}",
redshiftEx.getMessage(), redshiftEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: {}", rt.getMessage());
            }
            throw cause;
        }

        logger.info("The identifier of the statement is " + id);
        waitForInputToContinue(scanner);
        try {
            CompletableFuture<Void> future =
redshiftActions.checkStatementAsync(id);
            future.join();

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof RedshiftDataException redshiftEx) {
                logger.info("Redshift Data error occurred: {} Error code: {}",
redshiftEx.getMessage(), redshiftEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: {}", rt.getMessage());
            }
            throw cause;
        }
        waitForInputToContinue(scanner);
        try {
            CompletableFuture<Void> future = redshiftActions.getResultsAsync(id);
            future.join();

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof RedshiftDataException redshiftEx) {
                logger.info("Redshift Data error occurred: {} Error code: {}",
redshiftEx.getMessage(), redshiftEx.awsErrorDetails().errorCode());
            } else {
```

```
                logger.info("An unexpected error occurred: {}", rt.getMessage());
            }
            throw cause;
        }
        waitForInputToContinue(scanner);
        logger.info(DASHES);

        logger.info(DASHES);
        logger.info("Now you will modify the Redshift cluster.");
        waitForInputToContinue(scanner);
        try {
            CompletableFuture<ModifyClusterResponse> future =
 redshiftActions.modifyClusterAsync(clusterId);;
            future.join();

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof RedshiftDataException redshiftEx) {
                logger.info("Redshift Data error occurred: {} Error code: {}",
 redshiftEx.getMessage(), redshiftEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: {}", rt.getMessage());
            }
            throw cause;
        }
        waitForInputToContinue(scanner);
        logger.info(DASHES);

        logger.info(DASHES);
        logger.info("Would you like to delete the Amazon Redshift cluster? (y/
n)");
        String delAns = scanner.nextLine().trim();
        if (delAns.equalsIgnoreCase("y")) {
            logger.info("You selected to delete {} ", clusterId);
            waitForInputToContinue(scanner);
            try {
                CompletableFuture<DeleteClusterResponse> future =
 redshiftActions.deleteRedshiftClusterAsync(clusterId);;
                future.join();

            } catch (RuntimeException rt) {
                Throwable cause = rt.getCause();
                if (cause instanceof RedshiftDataException redshiftEx) {
```

```
                    logger.info("Redshift Data error occurred: {} Error code:
{}", redshiftEx.getMessage(), redshiftEx.awsErrorDetails().errorCode());
                } else {
                    logger.info("An unexpected error occurred: {}",
rt.getMessage());
                }
                throw cause;
            }
        } else {
            logger.info("The {}  was not deleted", clusterId);
        }
        logger.info(DASHES);

        logger.info(DASHES);
        logger.info("This concludes the Amazon Redshift SDK Basics scenario.");
        logger.info(DASHES);
    }

    private static SecretsManagerClient getSecretClient() {
        Region region = Region.US_EAST_1;
        return SecretsManagerClient.builder()
            .region(region)
            .build();
    }

    private static void waitForInputToContinue(Scanner scanner) {
        while (true) {
            System.out.println("");
            System.out.println("Enter 'c' followed by <ENTER> to continue:");
            String input = scanner.nextLine();

            if (input.trim().equalsIgnoreCase("c")) {
                System.out.println("Continuing with the program...");
                System.out.println("");
                break;
            } else {
                // Handle invalid input.
                System.out.println("Invalid input. Please try again.");
            }
        }
    }

    // Get the Amazon Redshift credentials from AWS Secrets Manager.
    private static String getSecretValues(String secretName) {
```

```
        SecretsManagerClient secretClient = getSecretClient();
        GetSecretValueRequest valueRequest = GetSecretValueRequest.builder()
            .secretId(secretName)
            .build();

        GetSecretValueResponse valueResponse =
 secretClient.getSecretValue(valueRequest);
        return valueResponse.secretString();
    }
}
```

Amazon Redshift SDK メソッドのラッパークラス。

```
public class RedshiftActions {

    private static final Logger logger =
 LoggerFactory.getLogger(RedshiftActions.class);
    private static RedshiftDataAsyncClient redshiftDataAsyncClient;

    private static RedshiftAsyncClient redshiftAsyncClient;

    private static RedshiftAsyncClient getAsyncClient() {
        if (redshiftAsyncClient == null) {
            SdkAsyncHttpClient httpClient = NettyNioAsyncHttpClient.builder()
                .maxConcurrency(100)
                .connectionTimeout(Duration.ofSeconds(60))
                .readTimeout(Duration.ofSeconds(60))
                .writeTimeout(Duration.ofSeconds(60))
                .build();

            ClientOverrideConfiguration overrideConfig =
 ClientOverrideConfiguration.builder()
                .apiCallTimeout(Duration.ofMinutes(2))
                .apiCallAttemptTimeout(Duration.ofSeconds(90))
                .retryStrategy(RetryMode.STANDARD)
                .build();

            redshiftAsyncClient = RedshiftAsyncClient.builder()
                .httpClient(httpClient)
                .overrideConfiguration(overrideConfig)
                .build();
        }
```

```
        return redshiftAsyncClient;
    }

    private static RedshiftDataAsyncClient getAsyncDataClient() {
        if (redshiftDataAsyncClient == null) {
            SdkAsyncHttpClient httpClient = NettyNioAsyncHttpClient.builder()
                .maxConcurrency(100)
                .connectionTimeout(Duration.ofSeconds(60))
                .readTimeout(Duration.ofSeconds(60))
                .writeTimeout(Duration.ofSeconds(60))
                .build();

            ClientOverrideConfiguration overrideConfig =
ClientOverrideConfiguration.builder()
                .apiCallTimeout(Duration.ofMinutes(2))
                .apiCallAttemptTimeout(Duration.ofSeconds(90))
                .retryStrategy(RetryMode.STANDARD)
                .build();

            redshiftDataAsyncClient = RedshiftDataAsyncClient.builder()
                .httpClient(httpClient)
                .overrideConfiguration(overrideConfig)
                .build();
        }
        return redshiftDataAsyncClient;
    }

    /**
     * Creates a new Amazon Redshift cluster asynchronously.
     * @param clusterId     the unique identifier for the cluster
     * @param username      the username for the administrative user
     * @param userPassword  the password for the administrative user
     * @return a CompletableFuture that represents the asynchronous operation of
creating the cluster
     * @throws RuntimeException if the cluster creation fails
     */
    public CompletableFuture<CreateClusterResponse> createClusterAsync(String
clusterId, String username, String userPassword) {
        CreateClusterRequest clusterRequest = CreateClusterRequest.builder()
            .clusterIdentifier(clusterId)
            .masterUsername(username)
            .masterUserPassword(userPassword)
            .nodeType("ra3.4xlarge")
            .publiclyAccessible(true)
```

```
            .numberOfNodes(2)
            .build();

        return getAsyncClient().createCluster(clusterRequest)
            .whenComplete((response, exception) -> {
                if (response != null) {
                    logger.info("Created cluster ");
                } else {
                    throw new RuntimeException("Failed to create cluster: " +
exception.getMessage(), exception);
                }
            });
    }

    /**
     * Waits asynchronously for the specified cluster to become available.
     * @param clusterId the identifier of the cluster to wait for
     * @return a {@link CompletableFuture} that completes when the cluster is
ready
     */
    public CompletableFuture<Void> waitForClusterReadyAsync(String clusterId) {
        DescribeClustersRequest clustersRequest =
DescribeClustersRequest.builder()
            .clusterIdentifier(clusterId)
            .build();

        logger.info("Waiting for cluster to become available. This may take a few
minutes.");
        long startTime = System.currentTimeMillis();

        // Recursive method to poll the cluster status.
        return checkClusterStatusAsync(clustersRequest, startTime);
    }

    private CompletableFuture<Void>
checkClusterStatusAsync(DescribeClustersRequest clustersRequest, long startTime)
{
        return getAsyncClient().describeClusters(clustersRequest)
            .thenCompose(clusterResponse -> {
                List<Cluster> clusterList = clusterResponse.clusters();
                boolean clusterReady = false;
                for (Cluster cluster : clusterList) {
                    if ("available".equals(cluster.clusterStatus())) {
                        clusterReady = true;
```

```
                        break;
                    }
                }

                if (clusterReady) {
                    logger.info(String.format("Cluster is available!"));
                    return CompletableFuture.completedFuture(null);
                } else {
                    long elapsedTimeMillis = System.currentTimeMillis() -
startTime;
                    long elapsedSeconds = elapsedTimeMillis / 1000;
                    long minutes = elapsedSeconds / 60;
                    long seconds = elapsedSeconds % 60;
                    System.out.printf("\rElapsed Time: %02d:%02d - Waiting for
cluster...", minutes, seconds);
                    System.out.flush();

                    // Wait 1 second before the next status check
                    return CompletableFuture.runAsync(() -> {
                        try {
                            TimeUnit.SECONDS.sleep(1);
                        } catch (InterruptedException e) {
                            throw new RuntimeException("Error during sleep: " +
e.getMessage(), e);
                        }
                    }).thenCompose(ignored ->
checkClusterStatusAsync(clustersRequest, startTime));
                }
            }).exceptionally(exception -> {
                throw new RuntimeException("Failed to get cluster status: " +
exception.getMessage(), exception);
            });
    }

    /**
     * Lists all databases asynchronously for the specified cluster, database
user, and database.
     * @param clusterId the identifier of the cluster to list databases for
     * @param dbUser the database user to use for the list databases request
     * @param database the database to list databases for
     * @return a {@link CompletableFuture} that completes when the database
listing is complete, or throws a {@link RuntimeException} if there was an error
     */
```

```
   public CompletableFuture<Void> listAllDatabasesAsync(String clusterId, String
dbUser, String database) {
       ListDatabasesRequest databasesRequest = ListDatabasesRequest.builder()
           .clusterIdentifier(clusterId)
           .dbUser(dbUser)
           .database(database)
           .build();

       // Asynchronous paginator for listing databases.
       ListDatabasesPublisher databasesPaginator =
getAsyncDataClient().listDatabasesPaginator(databasesRequest);
       CompletableFuture<Void> future = databasesPaginator.subscribe(response ->
{
           response.databases().forEach(db -> {
               logger.info("The database name is {} ", db);
           });
       });

       // Return the future for asynchronous handling.
       return future.exceptionally(exception -> {
           throw new RuntimeException("Failed to list databases: " +
exception.getMessage(), exception);
       });
   }

   /**
    * Creates an asynchronous task to execute a SQL statement for creating a new
table.
    *
    * @param clusterId    the identifier of the Amazon Redshift cluster
    * @param databaseName the name of the database to create the table in
    * @param userName     the username to use for the database connection
    * @return a {@link CompletableFuture} that completes with the result of the
SQL statement execution
    * @throws RuntimeException if there is an error creating the table
    */
   public CompletableFuture<ExecuteStatementResponse> createTableAsync(String
clusterId, String databaseName, String userName) {
       ExecuteStatementRequest createTableRequest =
ExecuteStatementRequest.builder()
           .clusterIdentifier(clusterId)
           .dbUser(userName)
           .database(databaseName)
           .sql("CREATE TABLE Movies (" +
```

```
                "id INT PRIMARY KEY, " +
                "title VARCHAR(100), " +
                "year INT)")
            .build();

        return getAsyncDataClient().executeStatement(createTableRequest)
            .whenComplete((response, exception) -> {
                if (exception != null) {
                    throw new RuntimeException("Error creating table: " +
exception.getMessage(), exception);
                } else {
                    logger.info("Table created: Movies");
                }
            });
    }

    /**
     * Asynchronously pops a table from a JSON file.
     *
     * @param clusterId   the ID of the cluster
     * @param databaseName the name of the database
     * @param userName    the username
     * @param fileName    the name of the JSON file
     * @param number      the number of records to process
     * @return a CompletableFuture that completes with the number of records
added to the Movies table
     */
    public CompletableFuture<Integer> popTableAsync(String clusterId, String
databaseName, String userName, String fileName, int number) {
        return CompletableFuture.supplyAsync(() -> {
                try {
                    JsonParser parser = new JsonFactory().createParser(new
File(fileName));
                    JsonNode rootNode = new ObjectMapper().readTree(parser);
                    Iterator<JsonNode> iter = rootNode.iterator();
                    return iter;
                } catch (IOException e) {
                    throw new RuntimeException("Failed to read or parse JSON
file: " + e.getMessage(), e);
                }
            }).thenCompose(iter -> processNodesAsync(clusterId, databaseName,
userName, iter, number))
            .whenComplete((result, exception) -> {
                if (exception != null) {
```

```
                    logger.info("Error {} ", exception.getMessage());
                } else {
                    logger.info("{} records were added to the Movies table." ,
result);
                }
            });
    }

    private CompletableFuture<Integer> processNodesAsync(String clusterId, String
databaseName, String userName, Iterator<JsonNode> iter, int number) {
        return CompletableFuture.supplyAsync(() -> {
            int t = 0;
            try {
                while (iter.hasNext()) {
                    if (t == number)
                        break;
                    JsonNode currentNode = iter.next();
                    int year = currentNode.get("year").asInt();
                    String title = currentNode.get("title").asText();

                    // Use SqlParameter to avoid SQL injection.
                    List<SqlParameter> parameterList = new ArrayList<>();
                    String sqlStatement = "INSERT INTO Movies
VALUES( :id , :title, :year);";
                    SqlParameter idParam = SqlParameter.builder()
                        .name("id")
                        .value(String.valueOf(t))
                        .build();

                    SqlParameter titleParam = SqlParameter.builder()
                        .name("title")
                        .value(title)
                        .build();

                    SqlParameter yearParam = SqlParameter.builder()
                        .name("year")
                        .value(String.valueOf(year))
                        .build();
                    parameterList.add(idParam);
                    parameterList.add(titleParam);
                    parameterList.add(yearParam);

                    ExecuteStatementRequest insertStatementRequest =
ExecuteStatementRequest.builder()
```

```
                        .clusterIdentifier(clusterId)
                        .sql(sqlStatement)
                        .database(databaseName)
                        .dbUser(userName)
                        .parameters(parameterList)
                        .build();


getAsyncDataClient().executeStatement(insertStatementRequest);
                    logger.info("Inserted: " + title + " (" + year + ")");
                    t++;
                }
            } catch (RedshiftDataException e) {
                throw new RuntimeException("Error inserting data: " +
e.getMessage(), e);
            }
            return t;
        });
    }

    /**
     * Checks the status of an SQL statement asynchronously and handles the
completion of the statement.
     *
     * @param sqlId the ID of the SQL statement to check
     * @return a {@link CompletableFuture} that completes when the SQL
statement's status is either "FINISHED" or "FAILED"
     */
    public CompletableFuture<Void> checkStatementAsync(String sqlId) {
        DescribeStatementRequest statementRequest =
DescribeStatementRequest.builder()
            .id(sqlId)
            .build();

        return getAsyncDataClient().describeStatement(statementRequest)
            .thenCompose(response -> {
                String status = response.statusAsString();
                logger.info("... Status: {} ", status);

                if ("FAILED".equals(status)) {
                    throw new RuntimeException("The Query Failed. Ending
program");
                } else if ("FINISHED".equals(status)) {
                    return CompletableFuture.completedFuture(null);
```

```
                } else {
                    // Sleep for 1 second and recheck status
                    return CompletableFuture.runAsync(() -> {
                        try {
                            TimeUnit.SECONDS.sleep(1);
                        } catch (InterruptedException e) {
                            throw new RuntimeException("Error during sleep: " +
e.getMessage(), e);
                        }
                    }).thenCompose(ignore -> checkStatementAsync(sqlId)); //
Recursively call until status is FINISHED or FAILED
                }
            }).whenComplete((result, exception) -> {
                if (exception != null) {
                    // Handle exceptions
                    logger.info("Error: {} ", exception.getMessage());
                } else {
                    logger.info("The statement is finished!");
                }
            });
    }

    /**
     * Asynchronously retrieves the results of a statement execution.
     *
     * @param statementId the ID of the statement for which to retrieve the
results
     * @return a {@link CompletableFuture} that completes when the statement
result has been processed
     */
    public CompletableFuture<Void> getResultsAsync(String statementId) {
        GetStatementResultRequest resultRequest =
GetStatementResultRequest.builder()
            .id(statementId)
            .build();

        return getAsyncDataClient().getStatementResult(resultRequest)
            .handle((response, exception) -> {
                if (exception != null) {
                    logger.info("Error getting statement result {} ",
exception.getMessage());
                    throw new RuntimeException("Error getting statement result: "
+ exception.getMessage(), exception);
                }
```

```
                // Extract and print the field values using streams if the
response is valid.
                response.records().stream()
                    .flatMap(List::stream)
                    .map(Field::stringValue)
                    .filter(value -> value != null)
                    .forEach(value -> System.out.println("The Movie title field
is " + value));

                return response;
            }).thenAccept(response -> {
                // Optionally add more logic here if needed after handling the
response
            });
    }


    /**
     * Asynchronously queries movies by a given year from a Redshift database.
     *
     * @param database    the name of the database to query
     * @param dbUser      the user to connect to the database with
     * @param year        the year to filter the movies by
     * @param clusterId   the identifier of the Redshift cluster to connect to
     * @return a {@link CompletableFuture} containing the response ID of the
executed SQL statement
     */
    public CompletableFuture<String> queryMoviesByYearAsync(String database,
                                                            String dbUser,
                                                            int year,
                                                            String
clusterId) {

        String sqlStatement = "SELECT * FROM Movies WHERE year = :year";
        SqlParameter yearParam = SqlParameter.builder()
            .name("year")
            .value(String.valueOf(year))
            .build();

        ExecuteStatementRequest statementRequest =
ExecuteStatementRequest.builder()
            .clusterIdentifier(clusterId)
            .database(database)
```

```
            .dbUser(dbUser)
            .parameters(yearParam)
            .sql(sqlStatement)
            .build();

        return CompletableFuture.supplyAsync(() -> {
            try {
                ExecuteStatementResponse response =
getAsyncDataClient().executeStatement(statementRequest).join(); // Use join() to
wait for the result
                return response.id();
            } catch (RedshiftDataException e) {
                throw new RuntimeException("Error executing statement: " +
e.getMessage(), e);
            }
        }).exceptionally(exception -> {
            logger.info("Error: {}", exception.getMessage());
            return "";
        });
    }

    /**
     * Modifies an Amazon Redshift cluster asynchronously.
     *
     * @param clusterId the identifier of the cluster to be modified
     * @return a {@link CompletableFuture} that completes when the cluster
modification is complete
     */
    public CompletableFuture<ModifyClusterResponse> modifyClusterAsync(String
clusterId) {
        ModifyClusterRequest modifyClusterRequest =
ModifyClusterRequest.builder()
            .clusterIdentifier(clusterId)
            .preferredMaintenanceWindow("wed:07:30-wed:08:00")
            .build();

        return getAsyncClient().modifyCluster(modifyClusterRequest)
            .whenComplete((clusterResponse, exception) -> {
                if (exception != null) {
                    if (exception.getCause() instanceof RedshiftException) {
                        logger.info("Error: {} ", exception.getMessage());
                    } else {
                        logger.info("Unexpected error: {} ",
exception.getMessage());
```

```
                    }
                } else {
                    logger.info("The modified cluster was successfully modified
 and has "
                        + clusterResponse.cluster().preferredMaintenanceWindow()
 + " as the maintenance window");
                }
            });
    }

    /**
     * Deletes a Redshift cluster asynchronously.
     *
     * @param clusterId the identifier of the Redshift cluster to be deleted
     * @return a {@link CompletableFuture} that represents the asynchronous
 operation of deleting the Redshift cluster
     */
    public CompletableFuture<DeleteClusterResponse>
 deleteRedshiftClusterAsync(String clusterId) {
        DeleteClusterRequest deleteClusterRequest =
 DeleteClusterRequest.builder()
            .clusterIdentifier(clusterId)
            .skipFinalClusterSnapshot(true)
            .build();

        return getAsyncClient().deleteCluster(deleteClusterRequest)
            .whenComplete((response, exception) -> {
                if (exception != null) {
                    // Handle exceptions
                    if (exception.getCause() instanceof RedshiftException) {
                        logger.info("Error: {}", exception.getMessage());
                    } else {
                        logger.info("Unexpected error: {}",
 exception.getMessage());
                    }
                } else {
                    // Handle successful response
                    logger.info("The status is {}",
 response.cluster().clusterStatus());
                }
            });
    }
}
```

- API の詳細については、「AWS SDK for Java 2.x API リファレンス」の以下のトピックを参照してください。
  - CreateCluste:
  - DescribeClusters
  - DescribeStatement
  - ExecuteStatement
  - GetStatementResult
  - ListDatabasesPaginator
  - ModifyCluster

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
class RedshiftScenario:
    """Runs an interactive scenario that shows how to get started with
 Redshift."""

    def __init__(self, redshift_wrapper, redshift_data_wrapper):
        self.redshift_wrapper = redshift_wrapper
        self.redshift_data_wrapper = redshift_data_wrapper

    def redhift_scenario(self, json_file_path):
        database_name = "dev"

        print(DASHES)
        print("Welcome to the Amazon Redshift SDK Getting Started example.")
        print(
            """
      This Python program demonstrates how to interact with Amazon Redshift
```

```
      using the AWS SDK for Python (Boto3).

      Amazon Redshift is a fully managed, petabyte-scale data warehouse
      service hosted in the cloud.

      The program's primary functionalities include cluster creation,
      verification of cluster readiness, listing databases, table creation,
      populating data within the table, and executing SQL statements.

      It also demonstrates querying data from the Movies table.

      Upon completion, all AWS resources are cleaned up.
    """
        )
        if not os.path.isfile(json_file_path):
            logging.error(f"The file {json_file_path} does not exist.")
            return

        print("Let's get started...")
        user_name = q.ask("Please enter your user name (default is awsuser):")
        user_name = user_name if user_name else "awsuser"

        print(DASHES)
        user_password = q.ask(
            "Please enter your user password (default is AwsUser1000):"
        )
        user_password = user_password if user_password else "AwsUser1000"

        print(DASHES)
        print(
            """A Redshift cluster refers to the collection of computing resources
 and storage that work
            together to process and analyze large volumes of data."""
        )
        cluster_id = q.ask(
            "Enter a cluster identifier value (default is redshift-cluster-
movies): "
        )
        cluster_id = cluster_id if cluster_id else "redshift-cluster-movies"

        self.redshift_wrapper.create_cluster(
            cluster_id, "ra3.4xlarge", user_name, user_password, True, 2
        )
```

```
        print(DASHES)
        print(f"Wait until {cluster_id} is available. This may take a few
minutes...")
        q.ask("Press Enter to continue...")

        self.wait_cluster_available(cluster_id)

        print(DASHES)

        print(
            f"""
       When you created {cluster_id}, the dev database is created by default and
used in this scenario.

       To create a custom database, you need to have a CREATEDB privilege.
       For more information, see the documentation here:
       https://docs.aws.amazon.com/redshift/latest/dg/r_CREATE_DATABASE.html.
      """
        )
        q.ask("Press Enter to continue...")
        print(DASHES)

        print(DASHES)
        print(f"List databases in {cluster_id}")
        q.ask("Press Enter to continue...")
        databases = self.redshift_data_wrapper.list_databases(
            cluster_id, database_name, user_name
        )
        print(f"The cluster contains {len(databases)} database(s).")
        for database in databases:
            print(f"    Database: {database}")
        print(DASHES)

        print(DASHES)
        print("Now you will create a table named Movies.")
        q.ask("Press Enter to continue...")

        self.create_table(cluster_id, database_name, user_name)

        print(DASHES)

        print("Populate the Movies table using the Movies.json file.")
        print(
```

```
            "Specify the number of records you would like to add to the Movies
Table."
        )
        print("Please enter a value between 50 and 200.")

        while True:
            try:
                num_records = int(q.ask("Enter a value: ", q.is_int))
                if 50 <= num_records <= 200:
                    break
                else:
                    print("Invalid input. Please enter a value between 50 and
200.")
            except ValueError:
                print("Invalid input. Please enter a value between 50 and 200.")

        self.populate_table(
            cluster_id, database_name, user_name, json_file_path, num_records
        )

        print(DASHES)
        print("Query the Movies table by year. Enter a value between 2012-2014.")

        while True:
            movie_year = int(q.ask("Enter a year: ", q.is_int))
            if 2012 <= movie_year <= 2014:
                break
            else:
                print("Invalid input. Please enter a valid year between 2012 and
2014.")

        # Function to query database
        sql_id = self.query_movies_by_year(
            database_name, user_name, movie_year, cluster_id
        )

        print(f"The identifier of the statement is {sql_id}")

        print("Checking statement status...")
        self.wait_statement_finished(sql_id)
        result = self.redshift_data_wrapper.get_statement_result(sql_id)

        self.display_movies(result)
```

```
        print(DASHES)

        print(DASHES)
        print("Now you will modify the Redshift cluster.")
        q.ask("Press Enter to continue...")

        preferred_maintenance_window = "wed:07:30-wed:08:00"
        self.redshift_wrapper.modify_cluster(cluster_id,
preferred_maintenance_window)

        print(DASHES)

        print(DASHES)
        delete = q.ask("Do you want to delete the cluster? (y/n) ", q.is_yesno)

        if delete:
            print(f"You selected to delete {cluster_id}")
            q.ask("Press Enter to continue...")
            self.redshift_wrapper.delete_cluster(cluster_id)
        else:
            print(f"Cluster {cluster_id}cluster_id was not deleted")

        print(DASHES)
        print("This concludes the Amazon Redshift SDK Getting Started scenario.")
        print(DASHES)

    def create_table(self, cluster_id, database, username):
        self.redshift_data_wrapper.execute_statement(
            cluster_identifier=cluster_id,
            database_name=database,
            user_name=username,
            sql="CREATE TABLE Movies (statement_id INT PRIMARY KEY, title
VARCHAR(100), year INT)",
        )

        print("Table created: Movies")


    def populate_table(self, cluster_id, database, username, file_name, number):
        with open(file_name) as f:
            data = json.load(f)

        i = 0
        for record in data:
```

```
            if i == number:
                break

            statement_id = i
            title = record["title"]
            year = record["year"]
            i = i + 1
            parameters = [
                {"name": "statement_id", "value": str(statement_id)},
                {"name": "title", "value": title},
                {"name": "year", "value": str(year)},
            ]

            self.redshift_data_wrapper.execute_statement(
                cluster_identifier=cluster_id,
                database_name=database,
                user_name=username,
                sql="INSERT INTO Movies VALUES(:statement_id, :title, :year)",
                parameter_list=parameters,
            )

        print(f"{i} records inserted into Movies table")

    def wait_cluster_available(self, cluster_id):
        """
        Waits for a cluster to be available.

        :param cluster_id: The cluster identifier.

        Note: The cluster_available waiter can also be used.
        It is not used in this case to allow an elapsed time message.
        """
        cluster_ready = False
        start_time = time.time()

        while not cluster_ready:
            time.sleep(30)
            cluster = self.redshift_wrapper.describe_clusters(cluster_id)
            status = cluster[0]["ClusterStatus"]
            if status == "available":
                cluster_ready = True
            elif status != "creating":
                raise Exception(
                    f"Cluster {cluster_id} creation failed with status {status}."
```

```
                )

            elapsed_seconds = int(round(time.time() - start_time))
            minutes = int(elapsed_seconds // 60)
            seconds = int(elapsed_seconds % 60)

            print(f"Elapsed Time: {minutes}:{seconds:02d} - status {status}...")

            if minutes > 30:
                raise Exception(
                    f"Cluster {cluster_id} is not available after 30 minutes."
                )

    def query_movies_by_year(self, database, username, year, cluster_id):
        sql = "SELECT * FROM Movies WHERE year = :year"

        params = [{"name": "year", "value": str(year)}]

        response = self.redshift_data_wrapper.execute_statement(
            cluster_identifier=cluster_id,
            database_name=database,
            user_name=username,
            sql=sql,
            parameter_list=params,
        )

        return response["Id"]

    @staticmethod
    def display_movies(response):
        metadata = response["ColumnMetadata"]
        records = response["Records"]

        title_column_index = None
        for i in range(len(metadata)):
            if metadata[i]["name"] == "title":
                title_column_index = i
                break

        if title_column_index is None:
            print("No title column found.")
            return

        print(f"Found {len(records)} movie(s).")
```

```
        for record in records:
            print(f"   {record[title_column_index]['stringValue']}")

    def wait_statement_finished(self, sql_id):
        while True:
            time.sleep(1)
            response = self.redshift_data_wrapper.describe_statement(sql_id)
            status = response["Status"]
            print(f"Statement status is {status}.")

            if status == "FAILED":
                print(f"The query failed because {response['Error']}. Ending
 program")
                raise Exception("The Query Failed. Ending program")
            elif status == "FINISHED":
                break
```

シナリオの実装を示すメイン関数。

```
def main():
    redshift_client = boto3.client("redshift")
    redshift_data_client = boto3.client("redshift-data")
    redshift_wrapper = RedshiftWrapper(redshift_client)
    redshift_data_wrapper = RedshiftDataWrapper(redshift_data_client)
    redshift_scenario = RedshiftScenario(redshift_wrapper, redshift_data_wrapper)
    redshift_scenario.redhift_scenario(
        f"{os.path.dirname(__file__)}/../../../resources/sample_files/
movies.json"
    )
```

シナリオで使用するラッパー関数。

```
    def create_cluster(
        self,
        cluster_identifier,
        node_type,
        master_username,
```

```
        master_user_password,
        publicly_accessible,
        number_of_nodes,
    ):
        """
        Creates a cluster.

        :param cluster_identifier: The name of the cluster.
        :param node_type: The type of node in the cluster.
        :param master_username: The master username.
        :param master_user_password: The master user password.
        :param publicly_accessible: Whether the cluster is publicly accessible.
        :param number_of_nodes: The number of nodes in the cluster.
        :return: The cluster.
        """

        try:
            cluster = self.client.create_cluster(
                ClusterIdentifier=cluster_identifier,
                NodeType=node_type,
                MasterUsername=master_username,
                MasterUserPassword=master_user_password,
                PubliclyAccessible=publicly_accessible,
                NumberOfNodes=number_of_nodes,
            )
            return cluster
        except ClientError as err:
            logging.error(
                "Couldn't create a cluster. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def describe_clusters(self, cluster_identifier):
        """
        Describes a cluster.

        :param cluster_identifier: The cluster identifier.
        :return: A list of clusters.
        """
        try:
            kwargs = {}
```

```
            if cluster_identifier:
                kwargs["ClusterIdentifier"] = cluster_identifier

            paginator = self.client.get_paginator("describe_clusters")
            clusters = []
            for page in paginator.paginate(**kwargs):
                clusters.extend(page["Clusters"])

            return clusters

        except ClientError as err:
            logging.error(
                "Couldn't describe a cluster. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def execute_statement(
        self, cluster_identifier, database_name, user_name, sql,
parameter_list=None
    ):
        """
        Executes a SQL statement.

        :param cluster_identifier: The cluster identifier.
        :param database_name: The database name.
        :param user_name: The user's name.
        :param sql: The SQL statement.
        :param parameter_list: The optional SQL statement parameters.
        :return: The SQL statement result.
        """

        try:
            kwargs = {
                "ClusterIdentifier": cluster_identifier,
                "Database": database_name,
                "DbUser": user_name,
                "Sql": sql,
            }
            if parameter_list:
                kwargs["Parameters"] = parameter_list
            response = self.client.execute_statement(**kwargs)
```

```
            return response
        except ClientError as err:
            logging.error(
                "Couldn't execute statement. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def describe_statement(self, statement_id):
        """
        Describes a SQL statement.

        :param statement_id: The SQL statement identifier.
        :return: The SQL statement result.
        """
        try:
            response = self.client.describe_statement(Id=statement_id)
            return response
        except ClientError as err:
            logging.error(
                "Couldn't describe statement. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def get_statement_result(self, statement_id):
        """
        Gets the result of a SQL statement.

        :param statement_id: The SQL statement identifier.
        :return: The SQL statement result.
        """
        try:
            result = {
                "Records": [],
            }
            paginator = self.client.get_paginator("get_statement_result")
            for page in paginator.paginate(Id=statement_id):
                if "ColumnMetadata" not in result:
                    result["ColumnMetadata"] = page["ColumnMetadata"]
```

```
                result["Records"].extend(page["Records"])
            return result
        except ClientError as err:
            logging.error(
                "Couldn't get statement result. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def modify_cluster(self, cluster_identifier, preferred_maintenance_window):
        """
        Modifies a cluster.

        :param cluster_identifier: The cluster identifier.
        :param preferred_maintenance_window: The preferred maintenance window.
        """
        try:
            self.client.modify_cluster(
                ClusterIdentifier=cluster_identifier,
                PreferredMaintenanceWindow=preferred_maintenance_window,
            )
        except ClientError as err:
            logging.error(
                "Couldn't modify a cluster. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def list_databases(self, cluster_identifier, database_name, database_user):
        """
        Lists databases in a cluster.

        :param cluster_identifier: The cluster identifier.
        :param database_name: The database name.
        :param database_user: The database user.
        :return: The list of databases.
        """
        try:
            paginator = self.client.get_paginator("list_databases")
            databases = []
```

```
            for page in paginator.paginate(
                ClusterIdentifier=cluster_identifier,
                Database=database_name,
                DbUser=database_user,
            ):
                databases.extend(page["Databases"])

            return databases
        except ClientError as err:
            logging.error(
                "Couldn't list databases. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def delete_cluster(self, cluster_identifier):
        """
        Deletes a cluster.

        :param cluster_identifier: The cluster identifier.
        """
        try:
            self.client.delete_cluster(
                ClusterIdentifier=cluster_identifier,
SkipFinalClusterSnapshot=True
            )
        except ClientError as err:
            logging.error(
                "Couldn't delete a cluster. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- API の詳細については、『AWS SDK for Python (Boto3) API リファレンス』の以下のトピックを参照してください。
  - CreateCluste:
  - DescribeClusters

- DescribeStatement
- ExecuteStatement
- GetStatementResult
- ListDatabasesPaginator
- ModifyCluster

AWS SDK デベロッパーガイドとコード例の完全なリストについては、「このサービスを AWS SDK で使用する」を参照してください。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK を使用した Data Service のアクション

次のコード例では、AWS SDK を使用して個々の Amazon Redshift アクションを実行する方法を示しています。それぞれの例には、GitHub へのリンクがあり、そこにはコードの設定と実行に関する説明が記載されています。

これらの抜粋は Amazon Redshift API を呼び出すもので、コンテキスト内で実行する必要がある大規模なプログラムからのコードの抜粋です。アクションは AWS SDK を使用した Amazon Redshift のシナリオ のコンテキスト内で確認できます。

以下の例には、最も一般的に使用されるアクションのみ含まれています。詳細な一覧については、「Amazon Redshift API Reference」を参照してください。

例

- AWS SDK または CLI で CreateCluster を使用する
- AWS SDK または CLI で DeleteCluster を使用する
- AWS SDK または CLI で DescribeClusters を使用する
- AWS SDK で DescribeStatement を使用する
- AWS SDK で ExecuteStatement を使用する
- AWS SDK で GetStatementResult を使用する
- AWS SDK で ListDatabases を使用する
- AWS SDK または CLI で ModifyCluster を使用する

AWS SDK または CLI で **CreateCluster** を使用する

次のサンプルコードは、CreateCluster を使用する方法を説明しています。

アクション例は、より大きなプログラムからのコードの抜粋であり、コンテキスト内で実行する必要があります。次のコード例で、このアクションのコンテキストを確認できます。

- 基本を学ぶ
- プロビジョニングされたデータウェアハウスクラスターの使用開始

.NET

SDK for .NET (v4)

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
    /// <summary>
    /// Create a new Amazon Redshift cluster.
    /// </summary>
    /// <param name="clusterIdentifier">The identifier for the cluster.</param>
    /// <param name="databaseName">The name of the database.</param>
    /// <param name="masterUsername">The master username.</param>
    /// <param name="masterUserPassword">The master user password.</param>
    /// <param name="nodeType">The node type for the cluster.</param>
    /// <returns>The cluster that was created.</returns>
    public async Task<Cluster> CreateClusterAsync(string clusterIdentifier,
string databaseName,
        string masterUsername, string masterUserPassword, string nodeType =
"ra3.large")
    {
        try
        {
            var request = new CreateClusterRequest
            {
                ClusterIdentifier = clusterIdentifier,
                DBName = databaseName,
                MasterUsername = masterUsername,
```

```
                MasterUserPassword = masterUserPassword,
                NodeType = nodeType,
                NumberOfNodes = 1,
                ClusterType = "single-node"
            };

            var response = await _redshiftClient.CreateClusterAsync(request);
            Console.WriteLine($"Created cluster {clusterIdentifier}");
            return response.Cluster;
        }
        catch (ClusterAlreadyExistsException ex)
        {
            Console.WriteLine($"Cluster already exists: {ex.Message}");
            throw;
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Couldn't create cluster. Here's why:
{ex.Message}");
            throw;
        }
    }
```

- API の詳細については、AWS SDK for .NET API リファレンスの「CreateCluster」を参照してください。

CLI

AWS CLI

この例では、最小限のパラメータセットでクラスターを作成します。デフォルトの出力は JSON 形式です。コマンドは次のとおりです。

```
aws redshift create-cluster --node-type dw.hs1.xlarge --number-of-nodes 2 --
master-username adminuser --master-user-password TopSecret1 --cluster-identifier
 mycluster
```

結果:

```
{
    "Cluster": {
```

```
      "NodeType": "dw.hs1.xlarge",
      "ClusterVersion": "1.0",
      "PubliclyAccessible": "true",
      "MasterUsername": "adminuser",
      "ClusterParameterGroups": [
         {
            "ParameterApplyStatus": "in-sync",
            "ParameterGroupName": "default.redshift-1.0"
         } ],
      "ClusterSecurityGroups": [
         {
            "Status": "active",
            "ClusterSecurityGroupName": "default"
         } ],
      "AllowVersionUpgrade": true,
      "VpcSecurityGroups": \[],
      "PreferredMaintenanceWindow": "sat:03:30-sat:04:00",
      "AutomatedSnapshotRetentionPeriod": 1,
      "ClusterStatus": "creating",
      "ClusterIdentifier": "mycluster",
      "DBName": "dev",
      "NumberOfNodes": 2,
      "PendingModifiedValues": {
         "MasterUserPassword": "\****"
      }
   },
   "ResponseMetadata": {
      "RequestId": "7cf4bcfc-64dd-11e2-bea9-49e0ce183f07"
   }
}
```

- API の詳細については、AWS CLI コマンドリファレンスの「CreateCluster」を参照してください。

Go

SDK for Go V2

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
import (
 "context"
 "errors"
 "log"
 "time"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/redshift"
 "github.com/aws/aws-sdk-go-v2/service/redshift/types"
)



// RedshiftActions wraps Redshift service actions.
type RedshiftActions struct {
 RedshiftClient *redshift.Client
}



// CreateCluster sends a request to create a cluster with the given clusterId
 using the provided credentials.
func (actor RedshiftActions) CreateCluster(ctx context.Context, clusterId string,
 userName string, userPassword string, nodeType string, clusterType string,
 publiclyAccessible bool) (*redshift.CreateClusterOutput, error) {
 // Create a new Redshift cluster
 input := &redshift.CreateClusterInput{
  ClusterIdentifier:  aws.String(clusterId),
  MasterUserPassword: aws.String(userPassword),
  MasterUsername:     aws.String(userName),
  NodeType:           aws.String(nodeType),
```

```
  ClusterType:        aws.String(clusterType),
  PubliclyAccessible: aws.Bool(publiclyAccessible),
 }
 var opErr *types.ClusterAlreadyExistsFault
 output, err := actor.RedshiftClient.CreateCluster(ctx, input)
 if err != nil && errors.As(err, &opErr) {
  log.Println("Cluster already exists")
  return nil, nil
 } else if err != nil {
  log.Printf("Failed to create Redshift cluster: %v\n", err)
  return nil, err
 }

 log.Printf("Created cluster %s\n", *output.Cluster.ClusterIdentifier)
 return output, nil
}
```

- API の詳細については、AWS SDK for Go API リファレンスの「CreateCluster」を参照してください。

Java

SDK for Java 2.x

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

クラスターを作成します。

```
    /**
     * Creates a new Amazon Redshift cluster asynchronously.
     * @param clusterId     the unique identifier for the cluster
     * @param username      the username for the administrative user
     * @param userPassword  the password for the administrative user
     * @return a CompletableFuture that represents the asynchronous operation of
 creating the cluster
```

```
     * @throws RuntimeException if the cluster creation fails
     */
    public CompletableFuture<CreateClusterResponse> createClusterAsync(String
clusterId, String username, String userPassword) {
        CreateClusterRequest clusterRequest = CreateClusterRequest.builder()
            .clusterIdentifier(clusterId)
            .masterUsername(username)
            .masterUserPassword(userPassword)
            .nodeType("ra3.4xlarge")
            .publiclyAccessible(true)
            .numberOfNodes(2)
            .build();

        return getAsyncClient().createCluster(clusterRequest)
            .whenComplete((response, exception) -> {
                if (response != null) {
                    logger.info("Created cluster ");
                } else {
                    throw new RuntimeException("Failed to create cluster: " +
exception.getMessage(), exception);
                }
            });
    }
```

- API の詳細については、「AWS SDK for Java 2.x API リファレンス」の「CreateCluster」を参照してください。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

クライアントを作成します。

```
import { RedshiftClient } from "@aws-sdk/client-redshift";
```

```
// Set the AWS Region.
const REGION = "REGION";
//Set the Redshift Service Object
const redshiftClient = new RedshiftClient({ region: REGION });
export { redshiftClient };
```

クラスターを作成します。

```
// Import required AWS SDK clients and commands for Node.js
import { CreateClusterCommand } from "@aws-sdk/client-redshift";
import { redshiftClient } from "./libs/redshiftClient.js";

const params = {
  ClusterIdentifier: "CLUSTER_NAME", // Required
  NodeType: "NODE_TYPE", //Required
  MasterUsername: "MASTER_USER_NAME", // Required - must be lowercase
  MasterUserPassword: "MASTER_USER_PASSWORD", // Required - must contain at least
 one uppercase letter, and one number
  ClusterType: "CLUSTER_TYPE", // Required
  IAMRoleARN: "IAM_ROLE_ARN", // Optional - the ARN of an IAM role with
 permissions your cluster needs to access other AWS services on your behalf, such
 as Amazon S3.
  ClusterSubnetGroupName: "CLUSTER_SUBNET_GROUPNAME", //Optional - the name of a
 cluster subnet group to be associated with this cluster. Defaults to 'default'
 if not specified.
  DBName: "DATABASE_NAME", // Optional - defaults to 'dev' if not specified
  Port: "PORT_NUMBER", // Optional - defaults to '5439' if not specified
};

const run = async () => {
  try {
    const data = await redshiftClient.send(new CreateClusterCommand(params));
    console.log(
      `Cluster ${data.Cluster.ClusterIdentifier} successfully created`,
    );
    return data; // For unit tests.
  } catch (err) {
    console.log("Error", err);
  }
};
run();
```

- API の詳細については、「AWS SDK for JavaScript API リファレンス」の「CreateCluster」を参照してください。

Kotlin

SDK for Kotlin

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

クラスターを作成します。

```
suspend fun createCluster(
    clusterId: String?,
    masterUsernameVal: String?,
    masterUserPasswordVal: String?,
) {
    val clusterRequest =
        CreateClusterRequest {
            clusterIdentifier = clusterId
            availabilityZone = "us-east-1a"
            masterUsername = masterUsernameVal
            masterUserPassword = masterUserPasswordVal
            nodeType = "ra3.4xlarge"
            publiclyAccessible = true
            numberOfNodes = 2
        }

    RedshiftClient.fromEnvironment { region = "us-east-1" }.use { redshiftClient
 ->
        val clusterResponse = redshiftClient.createCluster(clusterRequest)
        println("Created cluster ${clusterResponse.cluster?.clusterIdentifier}")
    }
}
```

- API の詳細については、AWS SDK for Kotlin API リファレンスの「CreateCluster」を参照してください。

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
class RedshiftWrapper:
    """
    Encapsulates Amazon Redshift cluster operations.
    """

    def __init__(self, redshift_client):
        """
        :param redshift_client: A Boto3 Redshift client.
        """
        self.client = redshift_client


    def create_cluster(
        self,
        cluster_identifier,
        node_type,
        master_username,
        master_user_password,
        publicly_accessible,
        number_of_nodes,
    ):
        """
        Creates a cluster.

        :param cluster_identifier: The name of the cluster.
        :param node_type: The type of node in the cluster.
        :param master_username: The master username.
        :param master_user_password: The master user password.
        :param publicly_accessible: Whether the cluster is publicly accessible.
        :param number_of_nodes: The number of nodes in the cluster.
        :return: The cluster.
        """
```

```
        try:
            cluster = self.client.create_cluster(
                ClusterIdentifier=cluster_identifier,
                NodeType=node_type,
                MasterUsername=master_username,
                MasterUserPassword=master_user_password,
                PubliclyAccessible=publicly_accessible,
                NumberOfNodes=number_of_nodes,
            )
            return cluster
        except ClientError as err:
            logging.error(
                "Couldn't create a cluster. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

次のコードは RedShiftWrapper オブジェクトをインスタンス化します。

```
    client = boto3.client("redshift")
    redhift_wrapper = RedshiftWrapper(client)
```

- API の詳細については、「AWS SDK for Python (Boto3) API リファレンス」の「CreateCluster」を参照してください。

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

クラスターを作成します。

```
    TRY.
        " Example values: iv_cluster_identifier = 'my-redshift-cluster'
        " Example values: iv_node_type = 'ra3.4xlarge'
        " Example values: iv_master_username = 'awsuser'
        " Example values: iv_master_password = 'AwsUser1000'
        " Example values: iv_publicly_accessible = abap_true
        " Example values: iv_number_of_nodes = 2
        oo_result = lo_rsh->createcluster(
          iv_clusteridentifier = iv_cluster_identifier
          iv_nodetype = iv_node_type
          iv_masterusername = iv_master_username
          iv_masteruserpassword = iv_master_password
          iv_publiclyaccessible = iv_publicly_accessible
          iv_numberofnodes = iv_number_of_nodes
        ).
        MESSAGE 'Redshift cluster created successfully.' TYPE 'I'.
      CATCH /aws1/cx_rshclustalrdyexfault.
        MESSAGE 'Cluster already exists.' TYPE 'I'.
      CATCH /aws1/cx_rshclstquotaexcdfault.
        MESSAGE 'Cluster quota exceeded.' TYPE 'I'.
    ENDTRY.
```

- API の詳細については、「AWS SDK for SAP ABAP API リファレンス」の「CreateCluster」を参照してください。

AWS SDK デベロッパーガイドとコード例の詳細なリストについては、このサービスを AWS SDK で使用する を参照してください。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK または CLI で **DeleteCluster** を使用する

次のサンプルコードは、DeleteCluster を使用する方法を説明しています。

アクション例は、より大きなプログラムからのコードの抜粋であり、コンテキスト内で実行する必要があります。次のコード例で、このアクションのコンテキストを確認できます。

- プロビジョニングされたデータウェアハウスクラスターの使用開始

.NET

SDK for .NET (v4)

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
    /// <summary>
    /// Delete an Amazon Redshift cluster without a final snapshot.
    /// </summary>
    /// <param name="clusterIdentifier">The identifier for the cluster.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> DeleteClusterWithoutSnapshotAsync(string
clusterIdentifier)
    {
        try
        {
            var request = new DeleteClusterRequest
            {
                ClusterIdentifier = clusterIdentifier,
                SkipFinalClusterSnapshot = true
            };

            var response = await _redshiftClient.DeleteClusterAsync(request);
            Console.WriteLine($"The {clusterIdentifier} was deleted");
            return true;
        }
        catch (ClusterNotFoundException ex)
        {
            Console.WriteLine($"Cluster not found: {ex.Message}");
            return false;
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Couldn't delete cluster. Here's why:
{ex.Message}");
            return false;
        }
    }
```

- API の詳細については、AWS SDK for .NET API リファレンスの「DeleteCluster」を参照してください。

CLI

AWS CLI

この例では、クラスターを削除してデータを強制的に削除し、最終的なクラスターのスナップショットは作成されません。コマンドは次のとおりです。

```
aws redshift delete-cluster --cluster-identifier mycluster --skip-final-cluster-snapshot
```

この例では、クラスターは削除しますが、最終クラスターのスナップショットを指定しています。コマンドは次のとおりです。

```
aws redshift delete-cluster --cluster-identifier mycluster --final-cluster-snapshot-identifier myfinalsnapshot
```

- API の詳細については、「AWS CLI コマンドリファレンス」の「DeleteCluster」を参照してください。

Go

SDK for Go V2

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
import (
 "context"
 "errors"
 "log"
```

```
 "time"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/redshift"
 "github.com/aws/aws-sdk-go-v2/service/redshift/types"
)



// RedshiftActions wraps Redshift service actions.
type RedshiftActions struct {
 RedshiftClient *redshift.Client
}



// DeleteCluster deletes the given cluster.
func (actor RedshiftActions) DeleteCluster(ctx context.Context, clusterId string)
 (bool, error) {
 input := redshift.DeleteClusterInput{
  ClusterIdentifier:        aws.String(clusterId),
  SkipFinalClusterSnapshot: aws.Bool(true),
 }
 _, err := actor.RedshiftClient.DeleteCluster(ctx, &input)
 var opErr *types.ClusterNotFoundFault
 if err != nil && errors.As(err, &opErr) {
  log.Println("Cluster was not found. Where could it be?")
  return false, err
 } else if err != nil {
  log.Printf("Failed to delete Redshift cluster: %v\n", err)
  return false, err
 }
 waiter := redshift.NewClusterDeletedWaiter(actor.RedshiftClient)
 err = waiter.Wait(ctx, &redshift.DescribeClustersInput{
  ClusterIdentifier: aws.String(clusterId),
 }, 5*time.Minute)
 if err != nil {
  log.Printf("Wait time exceeded for deleting cluster, continuing: %v\n", err)
 }
 log.Printf("The cluster %s was deleted\n", clusterId)
 return true, nil
}
```

- API の詳細については、AWS SDK for Go API リファレンスの「DeleteCluster」を参照してください。

Java

SDK for Java 2.x

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

クラスターを削除します。

```
   /**
    * Deletes a Redshift cluster asynchronously.
    *
    * @param clusterId the identifier of the Redshift cluster to be deleted
    * @return a {@link CompletableFuture} that represents the asynchronous
operation of deleting the Redshift cluster
    */
   public CompletableFuture<DeleteClusterResponse>
deleteRedshiftClusterAsync(String clusterId) {
       DeleteClusterRequest deleteClusterRequest =
DeleteClusterRequest.builder()
           .clusterIdentifier(clusterId)
           .skipFinalClusterSnapshot(true)
           .build();

       return getAsyncClient().deleteCluster(deleteClusterRequest)
           .whenComplete((response, exception) -> {
               if (exception != null) {
                   // Handle exceptions
                   if (exception.getCause() instanceof RedshiftException) {
                       logger.info("Error: {}", exception.getMessage());
                   } else {
                       logger.info("Unexpected error: {}",
exception.getMessage());
```

```
                    }
                } else {
                    // Handle successful response
                    logger.info("The status is {}",
 response.cluster().clusterStatus());
                }
            });
    }
```

- API の詳細については、「AWS SDK for Java 2.x API リファレンス」の「DeleteCluster」を参照してください。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

クライアントを作成します。

```
import { RedshiftClient } from "@aws-sdk/client-redshift";
// Set the AWS Region.
const REGION = "REGION";
//Set the Redshift Service Object
const redshiftClient = new RedshiftClient({ region: REGION });
export { redshiftClient };
```

クラスターを作成します。

```
// Import required AWS SDK clients and commands for Node.js
import { DeleteClusterCommand } from "@aws-sdk/client-redshift";
import { redshiftClient } from "./libs/redshiftClient.js";

const params = {
  ClusterIdentifier: "CLUSTER_NAME",
```

```
  SkipFinalClusterSnapshot: false,
  FinalClusterSnapshotIdentifier: "CLUSTER_SNAPSHOT_ID",
};

const run = async () => {
  try {
    const data = await redshiftClient.send(new DeleteClusterCommand(params));
    console.log("Success, cluster deleted. ", data);
    return data; // For unit tests.
  } catch (err) {
    console.log("Error", err);
  }
};
run();
```

- API の詳細については、「AWS SDK for JavaScript API リファレンス」の「DeleteCluster」を参照してください。

Kotlin

SDK for Kotlin

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

クラスターを削除します。

```
suspend fun deleteRedshiftCluster(clusterId: String?) {
    val request =
        DeleteClusterRequest {
            clusterIdentifier = clusterId
            skipFinalClusterSnapshot = true
        }

    RedshiftClient.fromEnvironment { region = "us-west-2" }.use { redshiftClient
 ->
        val response = redshiftClient.deleteCluster(request)
        println("The status is ${response.cluster?.clusterStatus}")
```

```
    }
}
```

- API の詳細については、AWS SDK for Kotlin API リファレンスの「DeleteCluster」を参照してください。

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
class RedshiftWrapper:
    """
    Encapsulates Amazon Redshift cluster operations.
    """

    def __init__(self, redshift_client):
        """
        :param redshift_client: A Boto3 Redshift client.
        """
        self.client = redshift_client


    def delete_cluster(self, cluster_identifier):
        """
        Deletes a cluster.

        :param cluster_identifier: The cluster identifier.
        """
        try:
            self.client.delete_cluster(
                ClusterIdentifier=cluster_identifier,
 SkipFinalClusterSnapshot=True
            )
        except ClientError as err:
```

```
            logging.error(
                "Couldn't delete a cluster. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

次のコードは RedShiftWrapper オブジェクトをインスタンス化します。

```
    client = boto3.client("redshift")
    redhift_wrapper = RedshiftWrapper(client)
```

- API の詳細については、「AWS SDK for Python (Boto3) API リファレンス」の「DeleteCluster」を参照してください。

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

クラスターを削除します。

```
    TRY.
        " Example values: iv_cluster_identifier = 'my-redshift-cluster'
        lo_rsh->deletecluster(
          iv_clusteridentifier = iv_cluster_identifier
          iv_skipfinalclustersnapshot = abap_true
        ).
        MESSAGE 'Redshift cluster deleted successfully.' TYPE 'I'.
      CATCH /aws1/cx_rshclustnotfoundfault.
        MESSAGE 'Cluster not found.' TYPE 'I'.
      CATCH /aws1/cx_rshinvcluststatefault.
        MESSAGE 'Invalid cluster state for deletion.' TYPE 'I'.
```

```
    ENDTRY.
```

- API の詳細については、「AWS SDK for SAP ABAP API リファレンス」の「DeleteCluster」を参照してください。

AWS SDK デベロッパーガイドとコード例の詳細なリストについては、このサービスを AWS SDK で使用する を参照してください。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK または CLI で **DescribeClusters** を使用する

次のサンプルコードは、DescribeClusters を使用する方法を説明しています。

アクション例は、より大きなプログラムからのコードの抜粋であり、コンテキスト内で実行する必要があります。次のコード例で、このアクションのコンテキストを確認できます。

- 基本を学ぶ
- プロビジョニングされたデータウェアハウスクラスターの使用開始

.NET

SDK for .NET (v4)

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
    /// <summary>
    /// Describe Amazon Redshift clusters.
    /// </summary>
    /// <param name="clusterIdentifier">Optional cluster identifier to describe a
specific cluster.</param>
    /// <returns>A list of clusters.</returns>
    public async Task<List<Cluster>> DescribeClustersAsync(string?
clusterIdentifier = null)
    {
```

```
        try
        {
            var clusters = new List<Cluster>();
            var request = new DescribeClustersRequest();
            if (!string.IsNullOrEmpty(clusterIdentifier))
            {
                request.ClusterIdentifier = clusterIdentifier;
            }

            var clustersPaginator =
_redshiftClient.Paginators.DescribeClusters(request);
            await foreach (var response in clustersPaginator.Responses)
            {
                if (response.Clusters != null)
                    clusters.AddRange(response.Clusters);
            }

            Console.WriteLine($"{clusters.Count} cluster(s) retrieved.");
            foreach (var cluster in clusters)
            {
                Console.WriteLine($"\t{cluster.ClusterIdentifier} (Status:
{cluster.ClusterStatus})");
            }

            return clusters;
        }
        catch (ClusterNotFoundException ex)
        {
            Console.WriteLine($"Cluster {clusterIdentifier} not found:
{ex.Message}");
            throw;
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Couldn't describe clusters. Here's why:
{ex.Message}");
            throw;
        }
    }
```

- API の詳細については、AWS SDK for .NET API リファレンスの「DescribeClusters」を参照してください。

CLI

AWS CLI

この例は、アカウントのすべてのクラスターの記述を返します。デフォルトの出力は JSON 形式です。コマンドは次のとおりです。

```
aws redshift describe-clusters
```

結果:

```
{
   "Clusters": [
   {
      "NodeType": "dw.hs1.xlarge",
      "Endpoint": {
         "Port": 5439,
         "Address": "mycluster.coqoarplqhsn.us-east-1.redshift.amazonaws.com"
      },
      "ClusterVersion": "1.0",
      "PubliclyAccessible": "true",
      "MasterUsername": "adminuser",
      "ClusterParameterGroups": [
         {
            "ParameterApplyStatus": "in-sync",
            "ParameterGroupName": "default.redshift-1.0"
         } ],
      "ClusterSecurityGroups": [
         {
            "Status": "active",
            "ClusterSecurityGroupName": "default"
         } ],
      "AllowVersionUpgrade": true,
      "VpcSecurityGroups": \[],
      "AvailabilityZone": "us-east-1a",
      "ClusterCreateTime": "2013-01-22T21:59:29.559Z",
      "PreferredMaintenanceWindow": "sat:03:30-sat:04:00",
      "AutomatedSnapshotRetentionPeriod": 1,
      "ClusterStatus": "available",
      "ClusterIdentifier": "mycluster",
      "DBName": "dev",
      "NumberOfNodes": 2,
      "PendingModifiedValues": {}
```

```
   } ],
   "ResponseMetadata": {
      "RequestId": "65b71cac-64df-11e2-8f5b-e90bd6c77476"
   }
}
```

--output text オプションコマンドを使用して、同じ情報をテキスト形式で取得することもできます。

--output text オプションコマンド:

オプションコマンド:

```
aws redshift describe-clusters --output text
```

結果:

```
dw.hs1.xlarge       1.0     true    adminuser       True    us-east-1a
 2013-01-22T21:59:29.559Z        sat:03:30-sat:04:00     1       available
 mycluster       dev     2
ENDPOINT    5439    mycluster.coqoarplqhsn.us-east-1.redshift.amazonaws.com
in-sync     default.redshift-1.0
active      default
PENDINGMODIFIEDVALUES
RESPONSEMETADATA    934281a8-64df-11e2-b07c-f7fbdd006c67
```

- API の詳細については、「AWS CLI コマンドリファレンス」の「DescribeClusters」を参照してください。

Go

SDK for Go V2

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
import (
 "context"
 "errors"
 "log"
 "time"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/redshift"
 "github.com/aws/aws-sdk-go-v2/service/redshift/types"
)



// RedshiftActions wraps Redshift service actions.
type RedshiftActions struct {
 RedshiftClient *redshift.Client
}



// DescribeClusters returns information about the given cluster.
func (actor RedshiftActions) DescribeClusters(ctx context.Context, clusterId
 string) (*redshift.DescribeClustersOutput, error) {
 input, err := actor.RedshiftClient.DescribeClusters(ctx,
 &redshift.DescribeClustersInput{
  ClusterIdentifier: aws.String(clusterId),
 })
 var opErr *types.AccessToClusterDeniedFault
 if errors.As(err, &opErr) {
  println("Access to cluster denied.")
  panic(err)
 } else if err != nil {
  println("Failed to describe Redshift clusters.")
  return nil, err
 }
 return input, nil
}
```

- API の詳細については、「AWS SDK for Go API リファレンス」の「DescribeClusters」を参照してください。

Java

SDK for Java 2.x

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

クラスターを記述する

```
   /**
    * Waits asynchronously for the specified cluster to become available.
    * @param clusterId the identifier of the cluster to wait for
    * @return a {@link CompletableFuture} that completes when the cluster is
ready
    */
   public CompletableFuture<Void> waitForClusterReadyAsync(String clusterId) {
       DescribeClustersRequest clustersRequest =
DescribeClustersRequest.builder()
           .clusterIdentifier(clusterId)
           .build();

       logger.info("Waiting for cluster to become available. This may take a few
minutes.");
       long startTime = System.currentTimeMillis();

       // Recursive method to poll the cluster status.
       return checkClusterStatusAsync(clustersRequest, startTime);
   }

   private CompletableFuture<Void>
checkClusterStatusAsync(DescribeClustersRequest clustersRequest, long startTime)
{
       return getAsyncClient().describeClusters(clustersRequest)
           .thenCompose(clusterResponse -> {
               List<Cluster> clusterList = clusterResponse.clusters();
               boolean clusterReady = false;
               for (Cluster cluster : clusterList) {
                   if ("available".equals(cluster.clusterStatus())) {
                       clusterReady = true;
                       break;
```

```
                    }
                }

                if (clusterReady) {
                    logger.info(String.format("Cluster is available!"));
                    return CompletableFuture.completedFuture(null);
                } else {
                    long elapsedTimeMillis = System.currentTimeMillis() -
startTime;
                    long elapsedSeconds = elapsedTimeMillis / 1000;
                    long minutes = elapsedSeconds / 60;
                    long seconds = elapsedSeconds % 60;
                    System.out.printf("\rElapsed Time: %02d:%02d - Waiting for
cluster...", minutes, seconds);
                    System.out.flush();

                    // Wait 1 second before the next status check
                    return CompletableFuture.runAsync(() -> {
                        try {
                            TimeUnit.SECONDS.sleep(1);
                        } catch (InterruptedException e) {
                            throw new RuntimeException("Error during sleep: " +
e.getMessage(), e);
                        }
                    }).thenCompose(ignored ->
checkClusterStatusAsync(clustersRequest, startTime));
                }
            }).exceptionally(exception -> {
                throw new RuntimeException("Failed to get cluster status: " +
exception.getMessage(), exception);
            });
    }
```

- API の詳細については、「AWS SDK for Java 2.x API リファレンス」の「DescribeClusters」を参照してください。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

クライアントを作成します。

```
import { RedshiftClient } from "@aws-sdk/client-redshift";
// Set the AWS Region.
const REGION = "REGION";
//Set the Redshift Service Object
const redshiftClient = new RedshiftClient({ region: REGION });
export { redshiftClient };
```

クラスターを記述します。

```
// Import required AWS SDK clients and commands for Node.js
import { DescribeClustersCommand } from "@aws-sdk/client-redshift";
import { redshiftClient } from "./libs/redshiftClient.js";

const params = {
  ClusterIdentifier: "CLUSTER_NAME",
};

const run = async () => {
  try {
    const data = await redshiftClient.send(new DescribeClustersCommand(params));
    console.log("Success", data);
    return data; // For unit tests.
  } catch (err) {
    console.log("Error", err);
  }
};
run();
```

- API の詳細については、「AWS SDK for JavaScript API リファレンス」の「DescribeClusters」を参照してください。

Kotlin

SDK for Kotlin

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

クラスターを記述します。

```
suspend fun describeRedshiftClusters() {
    RedshiftClient.fromEnvironment { region = "us-west-2" }.use { redshiftClient
 ->
        val clusterResponse =
 redshiftClient.describeClusters(DescribeClustersRequest {})
        val clusterList = clusterResponse.clusters

        if (clusterList != null) {
            for (cluster in clusterList) {
                println("Cluster database name is ${cluster.dbName}")
                println("Cluster status is ${cluster.clusterStatus}")
            }
        }
    }
}
```

- API の詳細については、AWS SDK for Kotlin API リファレンスの「DescribeClusters」を参照してください。

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
class RedshiftWrapper:
    """
    Encapsulates Amazon Redshift cluster operations.
    """

    def __init__(self, redshift_client):
        """
        :param redshift_client: A Boto3 Redshift client.
        """
        self.client = redshift_client


    def describe_clusters(self, cluster_identifier):
        """
        Describes a cluster.

        :param cluster_identifier: The cluster identifier.
        :return: A list of clusters.
        """
        try:
            kwargs = {}
            if cluster_identifier:
                kwargs["ClusterIdentifier"] = cluster_identifier

            paginator = self.client.get_paginator("describe_clusters")
            clusters = []
            for page in paginator.paginate(**kwargs):
                clusters.extend(page["Clusters"])

            return clusters

        except ClientError as err:
```

```
            logging.error(
                "Couldn't describe a cluster. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

次のコードは RedShiftWrapper オブジェクトをインスタンス化します。

```
    client = boto3.client("redshift")
    redhift_wrapper = RedshiftWrapper(client)
```

- API の詳細については、「AWS SDK for Python (Boto3) API リファレンス」の「DescribeClusters」を参照してください。

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

クラスターを記述する

```
    TRY.
        " Example values: iv_cluster_identifier = 'my-redshift-
cluster' (optional)
        oo_result = lo_rsh->describeclusters(
          iv_clusteridentifier = iv_cluster_identifier
        ).
        lt_clusters = oo_result->get_clusters( ).
        lv_cluster_count = lines( lt_clusters ).
        MESSAGE |Retrieved { lv_cluster_count } cluster(s).| TYPE 'I'.
      CATCH /aws1/cx_rshclustnotfoundfault.
        MESSAGE 'Cluster not found.' TYPE 'I'.
```

```
    ENDTRY.
```

- API の詳細については、「AWS SDK for SAP ABAP API リファレンス」の「DescribeClusters」を参照してください。

AWS SDK デベロッパーガイドとコード例の詳細なリストについては、このサービスを AWS SDK で使用する を参照してください。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK で **DescribeStatement** を使用する

次のサンプルコードは、DescribeStatement を使用する方法を説明しています。

アクション例は、より大きなプログラムからのコードの抜粋であり、コンテキスト内で実行する必要があります。次のコード例で、このアクションのコンテキストを確認できます。

- 基本を学ぶ

.NET

SDK for .NET (v4)

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
    /// <summary>
    /// Describe a statement execution.
    /// </summary>
    /// <param name="statementId">The statement ID.</param>
    /// <returns>The statement description.</returns>
    public async Task<DescribeStatementResponse> DescribeStatementAsync(string
statementId)
    {
        try
        {
```

```
            var request = new DescribeStatementRequest
            {
                Id = statementId
            };

            var response = await
_redshiftDataClient.DescribeStatementAsync(request);
            return response;
        }
        catch (Amazon.RedshiftDataAPIService.Model.ResourceNotFoundException ex)
        {
            Console.WriteLine($"Statement not found: {ex.Message}");
            throw;
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Couldn't describe statement. Here's why:
{ex.Message}");
            throw;
        }
    }
```

- API の詳細については、「AWS SDK for .NET API リファレンス」の「DescribeStatement」を参照してください。

Java

SDK for Java 2.x

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
    /**
     * Checks the status of an SQL statement asynchronously and handles the
completion of the statement.
     *
     * @param sqlId the ID of the SQL statement to check
```

```
     * @return a {@link CompletableFuture} that completes when the SQL
statement's status is either "FINISHED" or "FAILED"
     */
    public CompletableFuture<Void> checkStatementAsync(String sqlId) {
        DescribeStatementRequest statementRequest =
DescribeStatementRequest.builder()
            .id(sqlId)
            .build();

        return getAsyncDataClient().describeStatement(statementRequest)
            .thenCompose(response -> {
                String status = response.statusAsString();
                logger.info("... Status: {} ", status);

                if ("FAILED".equals(status)) {
                    throw new RuntimeException("The Query Failed. Ending
program");
                } else if ("FINISHED".equals(status)) {
                    return CompletableFuture.completedFuture(null);
                } else {
                    // Sleep for 1 second and recheck status
                    return CompletableFuture.runAsync(() -> {
                        try {
                            TimeUnit.SECONDS.sleep(1);
                        } catch (InterruptedException e) {
                            throw new RuntimeException("Error during sleep: " +
e.getMessage(), e);
                        }
                    }).thenCompose(ignore -> checkStatementAsync(sqlId)); //
Recursively call until status is FINISHED or FAILED
                }
            }).whenComplete((result, exception) -> {
                if (exception != null) {
                    // Handle exceptions
                    logger.info("Error: {} ", exception.getMessage());
                } else {
                    logger.info("The statement is finished!");
                }
            });
    }
```

- API の詳細については、「AWS SDK for Java 2.x API リファレンス」の「DescribeStatement」を参照してください。

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
class RedshiftDataWrapper:
    """Encapsulates Amazon Redshift data."""

    def __init__(self, client):
        """
        :param client: A Boto3 RedshiftDataWrapper client.
        """
        self.client = client


    def describe_statement(self, statement_id):
        """
        Describes a SQL statement.

        :param statement_id: The SQL statement identifier.
        :return: The SQL statement result.
        """
        try:
            response = self.client.describe_statement(Id=statement_id)
            return response
        except ClientError as err:
            logging.error(
                "Couldn't describe statement. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

次のコードは RedShiftDataWrapper オブジェクトをインスタンス化します。

```
    client = boto3.client("redshift-data")
    redshift_data_wrapper = RedshiftDataWrapper(client)
```

- API の詳細については、「AWS SDK for Python (Boto3) API リファレンス」の「DescribeStatement」を参照してください。

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
    TRY.
        " Example values: iv_statement_id = 'xxxxxxxx-xxxx-xxxx-xxxx-
xxxxxxxxxxxx'
        oo_result = lo_rsd->describestatement(
          iv_id = iv_statement_id
        ).
        lv_status = oo_result->get_status( ).
        MESSAGE |Statement status: { lv_status }| TYPE 'I'.
      CATCH /aws1/cx_rsdresourcenotfoundex.
        MESSAGE 'Statement not found.' TYPE 'I'.
      CATCH /aws1/cx_rsdinternalserverex.
        MESSAGE 'Internal server error.' TYPE 'I'.
    ENDTRY.
```

- API の詳細については、「AWS SDK for SAP ABAP API リファレンス」の「DescribeStatement」を参照してください。

AWS SDK デベロッパーガイドとコード例の詳細なリストについては、このサービスを AWS SDK で使用する を参照してください。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK で **ExecuteStatement** を使用する

次のサンプルコードは、ExecuteStatement を使用する方法を説明しています。

アクション例は、より大きなプログラムからのコードの抜粋であり、コンテキスト内で実行する必要があります。次のコード例で、このアクションのコンテキストを確認できます。

- 基本を学ぶ

Java

SDK for Java 2.x

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

SQL ステートメントを実行してデータベーステーブルを作成します。

```
   /**
    * Creates an asynchronous task to execute a SQL statement for creating a new
table.
    *
    * @param clusterId    the identifier of the Amazon Redshift cluster
    * @param databaseName the name of the database to create the table in
    * @param userName     the username to use for the database connection
    * @return a {@link CompletableFuture} that completes with the result of the
SQL statement execution
    * @throws RuntimeException if there is an error creating the table
    */
   public CompletableFuture<ExecuteStatementResponse> createTableAsync(String
clusterId, String databaseName, String userName) {
       ExecuteStatementRequest createTableRequest =
ExecuteStatementRequest.builder()
           .clusterIdentifier(clusterId)
```

```
            .dbUser(userName)
            .database(databaseName)
            .sql("CREATE TABLE Movies (" +
                "id INT PRIMARY KEY, " +
                "title VARCHAR(100), " +
                "year INT)")
            .build();

        return getAsyncDataClient().executeStatement(createTableRequest)
            .whenComplete((response, exception) -> {
                if (exception != null) {
                    throw new RuntimeException("Error creating table: " +
exception.getMessage(), exception);
                } else {
                    logger.info("Table created: Movies");
                }
            });
    }
```

SQL ステートメントを実行して、データベーステーブルにデータを挿入します。

```
    /**
     * Asynchronously pops a table from a JSON file.
     *
     * @param clusterId   the ID of the cluster
     * @param databaseName the name of the database
     * @param userName    the username
     * @param fileName    the name of the JSON file
     * @param number      the number of records to process
     * @return a CompletableFuture that completes with the number of records
added to the Movies table
     */
    public CompletableFuture<Integer> popTableAsync(String clusterId, String
databaseName, String userName, String fileName, int number) {
        return CompletableFuture.supplyAsync(() -> {
                try {
                    JsonParser parser = new JsonFactory().createParser(new
File(fileName));
                    JsonNode rootNode = new ObjectMapper().readTree(parser);
                    Iterator<JsonNode> iter = rootNode.iterator();
                    return iter;
                } catch (IOException e) {
```

```
                    throw new RuntimeException("Failed to read or parse JSON
file: " + e.getMessage(), e);
                }
            }).thenCompose(iter -> processNodesAsync(clusterId, databaseName,
userName, iter, number))
            .whenComplete((result, exception) -> {
                if (exception != null) {
                    logger.info("Error {} ", exception.getMessage());
                } else {
                    logger.info("{} records were added to the Movies table." ,
result);
                }
            });
    }

    private CompletableFuture<Integer> processNodesAsync(String clusterId, String
databaseName, String userName, Iterator<JsonNode> iter, int number) {
        return CompletableFuture.supplyAsync(() -> {
            int t = 0;
            try {
                while (iter.hasNext()) {
                    if (t == number)
                        break;
                    JsonNode currentNode = iter.next();
                    int year = currentNode.get("year").asInt();
                    String title = currentNode.get("title").asText();

                    // Use SqlParameter to avoid SQL injection.
                    List<SqlParameter> parameterList = new ArrayList<>();
                    String sqlStatement = "INSERT INTO Movies
VALUES( :id , :title, :year);";
                    SqlParameter idParam = SqlParameter.builder()
                        .name("id")
                        .value(String.valueOf(t))
                        .build();

                    SqlParameter titleParam = SqlParameter.builder()
                        .name("title")
                        .value(title)
                        .build();

                    SqlParameter yearParam = SqlParameter.builder()
                        .name("year")
                        .value(String.valueOf(year))
```

```
                        .build();
                    parameterList.add(idParam);
                    parameterList.add(titleParam);
                    parameterList.add(yearParam);

                    ExecuteStatementRequest insertStatementRequest =
ExecuteStatementRequest.builder()
                        .clusterIdentifier(clusterId)
                        .sql(sqlStatement)
                        .database(databaseName)
                        .dbUser(userName)
                        .parameters(parameterList)
                        .build();


getAsyncDataClient().executeStatement(insertStatementRequest);
                    logger.info("Inserted: " + title + " (" + year + ")");
                    t++;
                }
            } catch (RedshiftDataException e) {
                throw new RuntimeException("Error inserting data: " +
e.getMessage(), e);
            }
            return t;
        });
    }
```

SQL ステートメントを実行して、データベーステーブルをクエリします。

```
    /**
     * Asynchronously queries movies by a given year from a Redshift database.
     *
     * @param database    the name of the database to query
     * @param dbUser      the user to connect to the database with
     * @param year        the year to filter the movies by
     * @param clusterId   the identifier of the Redshift cluster to connect to
     * @return a {@link CompletableFuture} containing the response ID of the
executed SQL statement
     */
    public CompletableFuture<String> queryMoviesByYearAsync(String database,
                                                            String dbUser,
                                                            int year,
```

```
                                                                    String
clusterId) {

        String sqlStatement = "SELECT * FROM Movies WHERE year = :year";
        SqlParameter yearParam = SqlParameter.builder()
            .name("year")
            .value(String.valueOf(year))
            .build();

        ExecuteStatementRequest statementRequest =
ExecuteStatementRequest.builder()
            .clusterIdentifier(clusterId)
            .database(database)
            .dbUser(dbUser)
            .parameters(yearParam)
            .sql(sqlStatement)
            .build();

        return CompletableFuture.supplyAsync(() -> {
            try {
                ExecuteStatementResponse response =
getAsyncDataClient().executeStatement(statementRequest).join(); // Use join() to
wait for the result
                return response.id();
            } catch (RedshiftDataException e) {
                throw new RuntimeException("Error executing statement: " +
e.getMessage(), e);
            }
        }).exceptionally(exception -> {
            logger.info("Error: {}", exception.getMessage());
            return "";
        });
    }
```

- API の詳細については、AWS SDK for Java 2.x API リファレンスの「ExecuteStatement」を参照してください。

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
    TRY.
        " Example values: iv_cluster_identifier = 'redshift-cluster-movies'
        " Example values: iv_database_name = 'dev'
        " Example values: iv_user_name = 'awsuser'
        " Example values: iv_sql = 'SELECT * FROM movies WHERE year = :year'
        " Example values: it_parameter_list - SQL parameters for parameterized
queries

        " Only pass parameters if the list is not empty
        IF it_parameter_list IS NOT INITIAL.
          oo_result = lo_rsd->executestatement(
            iv_clusteridentifier = iv_cluster_identifier
            iv_database = iv_database_name
            iv_dbuser = iv_user_name
            iv_sql = iv_sql
            it_parameters = it_parameter_list
          ).
        ELSE.
          oo_result = lo_rsd->executestatement(
            iv_clusteridentifier = iv_cluster_identifier
            iv_database = iv_database_name
            iv_dbuser = iv_user_name
            iv_sql = iv_sql
          ).
        ENDIF.

        lv_statement_id = oo_result->get_id( ).
        MESSAGE |Statement executed. ID: { lv_statement_id }| TYPE 'I'.
      CATCH /aws1/cx_rsdexecutestatementex.
        MESSAGE 'Statement execution error.' TYPE 'I'.
      CATCH /aws1/cx_rsdresourcenotfoundex.
        MESSAGE 'Resource not found.' TYPE 'I'.
```

```
    ENDTRY.
```

- API の詳細については、「AWS SDK for SAP ABAP API リファレンス」の「ExecuteStatement」を参照してください。

AWS SDK デベロッパーガイドとコード例の詳細なリストについては、このサービスを AWS SDK で使用する を参照してください。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK で **GetStatementResult** を使用する

次のサンプルコードは、GetStatementResult を使用する方法を説明しています。

アクション例は、より大きなプログラムからのコードの抜粋であり、コンテキスト内で実行する必要があります。次のコード例で、このアクションのコンテキストを確認できます。

- 基本を学ぶ

.NET

SDK for .NET (v4)

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
    /// <summary>
    /// Get the results of a statement execution.
    /// </summary>
    /// <param name="statementId">The statement ID.</param>
    /// <returns>A list of result rows.</returns>
    public async Task<List<List<Field>>> GetStatementResultAsync(string
statementId)
    {
        try
        {
```

```
            var request = new GetStatementResultRequest
            {
                Id = statementId
            };

            var response = await
_redshiftDataClient.GetStatementResultAsync(request);
            return response.Records;
        }
        catch (Amazon.RedshiftDataAPIService.Model.ResourceNotFoundException ex)
        {
            Console.WriteLine($"Statement not found: {ex.Message}");
            throw;
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Couldn't get statement result. Here's why:
{ex.Message}");
            throw;
        }
    }
```

- API の詳細については、「AWS SDK for .NET API リファレンス」の「GetStatementResult」を参照してください。

Java

SDK for Java 2.x

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

ステートメントの結果を確認します。

```
    /**
     * Asynchronously retrieves the results of a statement execution.
     *
```

```
    * @param statementId the ID of the statement for which to retrieve the
results
    * @return a {@link CompletableFuture} that completes when the statement
result has been processed
    */
   public CompletableFuture<Void> getResultsAsync(String statementId) {
       GetStatementResultRequest resultRequest =
GetStatementResultRequest.builder()
           .id(statementId)
           .build();

       return getAsyncDataClient().getStatementResult(resultRequest)
           .handle((response, exception) -> {
               if (exception != null) {
                   logger.info("Error getting statement result {} ",
exception.getMessage());
                   throw new RuntimeException("Error getting statement result: "
+ exception.getMessage(), exception);
               }

               // Extract and print the field values using streams if the
response is valid.
               response.records().stream()
                   .flatMap(List::stream)
                   .map(Field::stringValue)
                   .filter(value -> value != null)
                   .forEach(value -> System.out.println("The Movie title field
is " + value));

               return response;
           }).thenAccept(response -> {
               // Optionally add more logic here if needed after handling the
response
           });
   }
```

- API の詳細については、「AWS SDK for Java 2.x API リファレンス」の「GetStatementResult」を参照してください。

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
class RedshiftDataWrapper:
    """Encapsulates Amazon Redshift data."""

    def __init__(self, client):
        """
        :param client: A Boto3 RedshiftDataWrapper client.
        """
        self.client = client


    def get_statement_result(self, statement_id):
        """
        Gets the result of a SQL statement.

        :param statement_id: The SQL statement identifier.
        :return: The SQL statement result.
        """
        try:
            result = {
                "Records": [],
            }
            paginator = self.client.get_paginator("get_statement_result")
            for page in paginator.paginate(Id=statement_id):
                if "ColumnMetadata" not in result:
                    result["ColumnMetadata"] = page["ColumnMetadata"]
                result["Records"].extend(page["Records"])
            return result
        except ClientError as err:
            logging.error(
                "Couldn't get statement result. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
```

```
            )
            raise
```

次のコードは RedShiftDataWrapper オブジェクトをインスタンス化します。

```
    client = boto3.client("redshift-data")
    redshift_data_wrapper = RedshiftDataWrapper(client)
```

- API の詳細については、「AWS SDK for Python (Boto3) API リファレンス」の「GetStatementResult」を参照してください。

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

ステートメントの結果を確認します。

```
    TRY.
        " Example values: iv_statement_id = 'xxxxxxxx-xxxx-xxxx-xxxx-
xxxxxxxxxxxx'
        " Handle pagination for large result sets

        DO.
          lo_result_page = lo_rsd->getstatementresult(
            iv_id = iv_statement_id
            iv_nexttoken = lv_next_token
          ).

          " Collect records from this page
          lt_page_records = lo_result_page->get_records( ).
          APPEND LINES OF lt_page_records TO lt_all_records.
```

```
        " Check if there are more pages
        lv_next_token = lo_result_page->get_nexttoken( ).
        IF lv_next_token IS INITIAL.
          EXIT. " No more pages
        ENDIF.
      ENDDO.

      " For the last call, set oo_result for return value
      oo_result = lo_result_page.
      lv_record_count = lines( lt_all_records ).
      MESSAGE |Retrieved { lv_record_count } record(s).| TYPE 'I'.
    CATCH /aws1/cx_rsdresourcenotfoundex.
      MESSAGE 'Statement not found or results not available.' TYPE 'I'.
    CATCH /aws1/cx_rsdinternalserverex.
      MESSAGE 'Internal server error.' TYPE 'I'.
  ENDTRY.
```

- API の詳細については、「AWS SDK for SAP ABAP API リファレンス」の「GetStatementResult」を参照してください。

AWS SDK デベロッパーガイドとコード例の詳細なリストについては、このサービスを AWS SDK で使用する を参照してください。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK で **ListDatabases** を使用する

次のサンプルコードは、ListDatabases を使用する方法を説明しています。

.NET

SDK for .NET (v4)

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
/// <summary>
/// List databases in a Redshift cluster.
```

```
    /// </summary>
    /// <param name="clusterIdentifier">The cluster identifier.</param>
    /// <param name="dbUser">The database user.</param>
    /// <param name="dbUser">The database name for authentication.</param>
    /// <returns>A list of database names.</returns>
    public async Task<List<string>> ListDatabasesAsync(string clusterIdentifier,
string dbUser, string databaseName)
    {
        try
        {
            var request = new ListDatabasesRequest
            {
                ClusterIdentifier = clusterIdentifier,
                DbUser = dbUser,
                Database = databaseName
            };

            var response = await _redshiftDataClient.ListDatabasesAsync(request);
            var databases = new List<string>();

            foreach (var database in response.Databases)
            {
                Console.WriteLine($"The database name is : {database}");
                databases.Add(database);
            }

            return databases;
        }
        catch (Amazon.RedshiftDataAPIService.Model.ValidationException ex)
        {
            Console.WriteLine($"Validation error: {ex.Message}");
            throw;
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Couldn't list databases. Here's why:
{ex.Message}");
            throw;
        }
    }
```

- API の詳細については、「AWS SDK for .NET API リファレンス」の「ListDatabases」を参照してください。

Java

SDK for Java 2.x

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
   /**
    * Lists all databases asynchronously for the specified cluster, database
user, and database.
    * @param clusterId the identifier of the cluster to list databases for
    * @param dbUser the database user to use for the list databases request
    * @param database the database to list databases for
    * @return a {@link CompletableFuture} that completes when the database
listing is complete, or throws a {@link RuntimeException} if there was an error
    */
   public CompletableFuture<Void> listAllDatabasesAsync(String clusterId, String
dbUser, String database) {
       ListDatabasesRequest databasesRequest = ListDatabasesRequest.builder()
           .clusterIdentifier(clusterId)
           .dbUser(dbUser)
           .database(database)
           .build();

       // Asynchronous paginator for listing databases.
       ListDatabasesPublisher databasesPaginator =
getAsyncDataClient().listDatabasesPaginator(databasesRequest);
       CompletableFuture<Void> future = databasesPaginator.subscribe(response ->
{
           response.databases().forEach(db -> {
               logger.info("The database name is {} ", db);
           });
       });

       // Return the future for asynchronous handling.
```

```
        return future.exceptionally(exception -> {
            throw new RuntimeException("Failed to list databases: " +
exception.getMessage(), exception);
        });
    }
```

- API の詳細については、「AWS SDK for Java 2.x API リファレンス」の「ListDatabases」を参照してください。

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
    TRY.
        " Example values: iv_cluster_identifier = 'redshift-cluster-movies'
        " Example values: iv_database_name = 'dev'
        " Example values: iv_database_user = 'awsuser'
        oo_result = lo_rsd->listdatabases(
          iv_clusteridentifier = iv_cluster_identifier
          iv_database = iv_database_name
          iv_dbuser = iv_database_user
        ).
        lt_databases = oo_result->get_databases( ).
        lv_db_count = lines( lt_databases ).
        MESSAGE |Retrieved { lv_db_count } database(s).| TYPE 'I'.
      CATCH /aws1/cx_rsddatabaseconnex.
        MESSAGE 'Database connection error.' TYPE 'I'.
      CATCH /aws1/cx_rsdresourcenotfoundex.
        MESSAGE 'Cluster not found.' TYPE 'I'.
    ENDTRY.
```

- API の詳細については、「AWS SDK for SAP ABAP API リファレンス」の「ListDatabases」を参照してください。

AWS SDK デベロッパーガイドとコード例の詳細なリストについては、このサービスを AWS SDK で使用する を参照してください。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK または CLI で **ModifyCluster** を使用する

次のサンプルコードは、ModifyCluster を使用する方法を説明しています。

アクション例は、より大きなプログラムからのコードの抜粋であり、コンテキスト内で実行する必要があります。次のコード例で、このアクションのコンテキストを確認できます。

- 基本を学ぶ

.NET

SDK for .NET (v4)

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
    /// <summary>
    /// Modify an Amazon Redshift cluster.
    /// </summary>
    /// <param name="clusterIdentifier">The identifier for the cluster.</param>
    /// <param name="preferredMaintenanceWindow">The preferred maintenance
window.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> ModifyClusterAsync(string clusterIdentifier, string
preferredMaintenanceWindow)
    {
        try
        {
            var request = new ModifyClusterRequest
            {
                ClusterIdentifier = clusterIdentifier,
                PreferredMaintenanceWindow = preferredMaintenanceWindow
            };
```

```
            var response = await _redshiftClient.ModifyClusterAsync(request);
            Console.WriteLine($"The modified cluster was successfully modified
and has {response.Cluster.PreferredMaintenanceWindow} as the maintenance
window");
            return true;
        }
        catch (ClusterNotFoundException ex)
        {
            Console.WriteLine($"Cluster {clusterIdentifier} not found:
{ex.Message}");
            return false;
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Couldn't modify cluster. Here's why:
{ex.Message}");
            return false;
        }
    }
```

- API の詳細については、AWS SDK for .NET API リファレンスの「ModifyCluster」を参照してください。

CLI

AWS CLI

この例は、クラスターセキュリティグループを指定されたクラスターに関連付ける方法を示しています。コマンドは次のとおりです。

```
aws redshift modify-cluster --cluster-identifier mycluster --cluster-security-
groups mysecuritygroup
```

この例は、各週のクラスターの優先メンテナンスウィンドウを、日曜日の午後 11 時 15 分から月曜日の午前 3 時 15 分までの 4 時間に変更する方法を示しています。コマンドは次のとおりです。

```
aws redshift modify-cluster --cluster-identifier mycluster --preferred-
maintenance-window Sun:23:15-Mon:03:15
```

この例では、クラスターのマスターパスワードを変更する方法を示しています。コマンドは次のとおりです。

```
aws redshift modify-cluster --cluster-identifier mycluster --master-user-password
 A1b2c3d4
```

- API の詳細については、AWS CLI コマンドリファレンスの「ModifyCluster」を参照してください。

Go

SDK for Go V2

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
import (
 "context"
 "errors"
 "log"
 "time"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/redshift"
 "github.com/aws/aws-sdk-go-v2/service/redshift/types"
)



// RedshiftActions wraps Redshift service actions.
type RedshiftActions struct {
 RedshiftClient *redshift.Client
}



// ModifyCluster sets the preferred maintenance window for the given cluster.
```

```
func (actor RedshiftActions) ModifyCluster(ctx context.Context, clusterId string,
 maintenanceWindow string) *redshift.ModifyClusterOutput {
 // Modify the cluster's maintenance window
 input := &redshift.ModifyClusterInput{
  ClusterIdentifier:          aws.String(clusterId),
  PreferredMaintenanceWindow: aws.String(maintenanceWindow),
 }

 var opErr *types.InvalidClusterStateFault
 output, err := actor.RedshiftClient.ModifyCluster(ctx, input)
 if err != nil && errors.As(err, &opErr) {
  log.Println("Cluster is in an invalid state.")
  panic(err)
 } else if err != nil {
  log.Printf("Failed to modify Redshift cluster: %v\n", err)
  panic(err)
 }

 log.Printf("The cluster was successfully modified and now has %s as the
 maintenance window\n", *output.Cluster.PreferredMaintenanceWindow)
 return output
}
```

- API の詳細については、AWS SDK for Go API リファレンスの「ModifyCluster」を参照してください。

Java

SDK for Java 2.x

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

クラスターを変更します。

```
    /**
```

```
    * Modifies an Amazon Redshift cluster asynchronously.
    *
    * @param clusterId the identifier of the cluster to be modified
    * @return a {@link CompletableFuture} that completes when the cluster
modification is complete
    */
   public CompletableFuture<ModifyClusterResponse> modifyClusterAsync(String
clusterId) {
       ModifyClusterRequest modifyClusterRequest =
ModifyClusterRequest.builder()
           .clusterIdentifier(clusterId)
           .preferredMaintenanceWindow("wed:07:30-wed:08:00")
           .build();

       return getAsyncClient().modifyCluster(modifyClusterRequest)
           .whenComplete((clusterResponse, exception) -> {
               if (exception != null) {
                   if (exception.getCause() instanceof RedshiftException) {
                       logger.info("Error: {} ", exception.getMessage());
                   } else {
                       logger.info("Unexpected error: {} ",
exception.getMessage());
                   }
               } else {
                   logger.info("The modified cluster was successfully modified
and has "
                       + clusterResponse.cluster().preferredMaintenanceWindow()
+ " as the maintenance window");
               }
           });
   }
```

- API の詳細については、「AWS SDK for Java 2.x API リファレンス」の「ModifyCluster」を参照してください。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

クライアントを作成します。

```
import { RedshiftClient } from "@aws-sdk/client-redshift";
// Set the AWS Region.
const REGION = "REGION";
//Set the Redshift Service Object
const redshiftClient = new RedshiftClient({ region: REGION });
export { redshiftClient };
```

クラスターを変更します。

```
// Import required AWS SDK clients and commands for Node.js
import { ModifyClusterCommand } from "@aws-sdk/client-redshift";
import { redshiftClient } from "./libs/redshiftClient.js";

// Set the parameters
const params = {
  ClusterIdentifier: "CLUSTER_NAME",
  MasterUserPassword: "NEW_MASTER_USER_PASSWORD",
};

const run = async () => {
  try {
    const data = await redshiftClient.send(new ModifyClusterCommand(params));
    console.log("Success was modified.", data);
    return data; // For unit tests.
  } catch (err) {
    console.log("Error", err);
  }
};
run();
```

- API の詳細については、「AWS SDK for JavaScript API リファレンス」の「ModifyCluster」を参照してください。

Kotlin

SDK for Kotlin

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

クラスターを変更します。

```
suspend fun modifyCluster(clusterId: String?) {
    val modifyClusterRequest =
        ModifyClusterRequest {
            clusterIdentifier = clusterId
            preferredMaintenanceWindow = "wed:07:30-wed:08:00"
        }

    RedshiftClient { region = "us-west-2" }.use { redshiftClient ->
        val clusterResponse = redshiftClient.modifyCluster(modifyClusterRequest)
        println(
            "The modified cluster was successfully modified and has
 ${clusterResponse.cluster?.preferredMaintenanceWindow} as the maintenance
 window",
        )
    }
}
```

- API の詳細については、AWS SDK for Kotlin API リファレンスの「ModifyCluster」を参照してください。

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

```
class RedshiftWrapper:
    """
    Encapsulates Amazon Redshift cluster operations.
    """

    def __init__(self, redshift_client):
        """
        :param redshift_client: A Boto3 Redshift client.
        """
        self.client = redshift_client


    def modify_cluster(self, cluster_identifier, preferred_maintenance_window):
        """
        Modifies a cluster.

        :param cluster_identifier: The cluster identifier.
        :param preferred_maintenance_window: The preferred maintenance window.
        """
        try:
            self.client.modify_cluster(
                ClusterIdentifier=cluster_identifier,
                PreferredMaintenanceWindow=preferred_maintenance_window,
            )
        except ClientError as err:
            logging.error(
                "Couldn't modify a cluster. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

次のコードは RedShiftWrapper オブジェクトをインスタンス化します。

```
    client = boto3.client("redshift")
    redhift_wrapper = RedshiftWrapper(client)
```

- API の詳細については、「AWS SDK for Python (Boto3) API リファレンス」の「ModifyCluster」を参照してください。

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWS コード例リポジトリでの設定と実行の方法を確認してください。

クラスターを変更します。

```
    TRY.
        " Example values: iv_cluster_identifier = 'my-redshift-cluster'
        " Example values: iv_pref_maintenance_wn = 'wed:07:30-wed:08:00'
        lo_rsh->modifycluster(
          iv_clusteridentifier = iv_cluster_identifier
          iv_preferredmaintenancewin00 = iv_pref_maintenance_wn
        ).
        MESSAGE 'Redshift cluster modified successfully.' TYPE 'I'.
      CATCH /aws1/cx_rshclustnotfoundfault.
        MESSAGE 'Cluster not found.' TYPE 'I'.
      CATCH /aws1/cx_rshinvcluststatefault.
        MESSAGE 'Invalid cluster state for modification.' TYPE 'I'.
    ENDTRY.
```

- API の詳細については、「AWS SDK for SAP ABAP API リファレンス」の「ModifyCluster」を参照してください。

AWS SDK デベロッパーガイドとコード例の詳細なリストについては、このサービスを AWS SDK で使用する を参照してください。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK を使用した Amazon Redshift のシナリオ

次のコード例は、AWS SDK を使用して Amazon Redshift で一般的なシナリオを実装する方法を示しています。これらのシナリオは、Amazon Redshift 内で複数の関数を呼び出すか、その他の AWS のサービスと組み合わせて、特定のタスクを実行する方法を説明しています。各シナリオには、完全なソースコードへのリンクが含まれており、そこからコードの設定方法と実行方法に関する手順を確認できます。

シナリオは、サービスアクションをコンテキストで理解するのに役立つ中級レベルの経験を対象としています。

例

- Amazon Redshift アイテムトラッカーの作成
- CLI を使用したサーバーレスデータウェアハウスの使用開始
- プロビジョニングされたデータウェアハウスクラスターの使用開始

Amazon Redshift アイテムトラッカーの作成

次のコード例は、Amazon Redshift データベースを使用して、作業項目を追跡してレポートするウェブアプリケーションを作成する方法を示しています。

Java

SDK for Java 2.x

Amazon Redshift データベースに保存されている作業項目を追跡してレポートするウェブアプリケーションを作成する方法を説明します。

Amazon Redshift サーバーレスデータをクエリする Spring REST API と React アプリケーションで使用するための完全なソースコードと設定方法については、GitHub にある詳しい事例を参照してください。

この例で使用されているサービス

- Amazon Redshift

- Amazon SES

Kotlin

SDK for Kotlin

Amazon Redshift データベースに保存されている作業項目を追跡してレポートするウェブアプリケーションを作成する方法を説明します。

Amazon Redshift サーバーレスデータをクエリする Spring REST API と React アプリケーションで使用するための完全なソースコードと設定方法については、GitHub にある詳しい事例を参照してください。

この例で使用されているサービス

- Amazon Redshift
- Amazon SES

AWS SDK デベロッパーガイドとコード例の詳細なリストについては、このサービスを AWS SDK で使用する を参照してください。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

CLI を使用したサーバーレスデータウェアハウスの使用開始

次のコード例は、以下の操作方法を示しています。

- secrets-manager CreateSecret を使用する
- secrets-manager DeleteSecret を使用する
- secrets-manager GetSecretValue を使用する
- Redshift CreateNamespace を使用する
- Redshift CreateWorkgroup を使用する
- Redshift DeleteNamespace を使用する
- iam CreateRole を使用する

Bash

AWS CLIBash スクリプトを使用した

> **Note**
>
> GitHub には、その他のリソースもあります。デベロッパーチュートリアルのサンプルレポジトリで用例一覧を検索し、設定と実行の方法を確認してください。

```
#!/bin/bash

# Amazon Redshift Serverless Tutorial Script with Secrets Manager (No jq
 dependency)
# This script creates a Redshift Serverless environment, loads sample data, and
 runs queries
# Uses AWS Secrets Manager for secure password management without requiring jq

# Set up logging
LOG_FILE="redshift-serverless-tutorial-v4.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon Redshift Serverless tutorial script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Function to check for errors in command output
check_error() {
  local output=$1
  local cmd=$2

  if echo "$output" | grep -i "error\|exception\|fail" > /dev/null; then
    echo "ERROR: Command failed: $cmd"
    echo "Output: $output"
    cleanup_resources
    exit 1
  fi
}

# Function to generate a secure password that meets Redshift requirements
generate_secure_password() {
  # Redshift password requirements:
  # - 8-64 characters
```

```
  # - At least one uppercase letter
  # - At least one lowercase letter
  # - At least one decimal digit
  # - Can contain printable ASCII characters except /, ", ', \, @, space

  local password=""
  local valid=false
  local attempts=0
  local max_attempts=10

  while [[ "$valid" == false && $attempts -lt $max_attempts ]]; do
    # Generate base password with safe characters
    local base=$(openssl rand -base64 12 | tr -d '/+=' | head -c 12)

    # Ensure we have at least one of each required character type
    local upper=$(echo "ABCDEFGHIJKLMNOPQRSTUVWXYZ" | fold -w1 | shuf -n1)
    local lower=$(echo "abcdefghijklmnopqrstuvwxyz" | fold -w1 | shuf -n1)
    local digit=$(echo "0123456789" | fold -w1 | shuf -n1)
    local special=$(echo "!#$%&*()_+-=[]{}|;:,.<>?" | fold -w1 | shuf -n1)

    # Combine and shuffle
    password="${base}${upper}${lower}${digit}${special}"
    password=$(echo "$password" | fold -w1 | shuf | tr -d '\n')

    # Validate password meets requirements
    if [[ ${#password} -ge 8 && ${#password} -le 64 ]] && \
       [[ "$password" =~ [A-Z] ]] && \
       [[ "$password" =~ [a-z] ]] && \
       [[ "$password" =~ [0-9] ]] && \
       [[ ! "$password" =~ [/\"\'\\@[:space:]] ]]; then
      valid=true
    fi

    ((attempts++))
  done

  if [[ "$valid" == false ]]; then
    echo "ERROR: Failed to generate valid password after $max_attempts attempts"
    exit 1
  fi

  echo "$password"
}
```

```
# Function to create secret in AWS Secrets Manager
create_secret() {
  local secret_name=$1
  local username=$2
  local password=$3
  local description=$4

  echo "Creating secret in AWS Secrets Manager: $secret_name"

  # Create the secret using AWS CLI without jq
  local secret_output=$(aws secretsmanager create-secret \
    --name "$secret_name" \
    --description "$description" \
    --secret-string "{\"username\":\"$username\",\"password\":\"$password\"}" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=redshift-serverless
 2>&1)

  if echo "$secret_output" | grep -i "error\|exception\|fail" > /dev/null; then
    echo "ERROR: Failed to create secret: $secret_output"
    return 1
  fi

  echo "Secret created successfully: $secret_name"
  return 0
}

# Function to retrieve password from AWS Secrets Manager
get_password_from_secret() {
  local secret_name=$1

  # Get the secret value and extract password using sed/grep instead of jq
  local secret_value=$(aws secretsmanager get-secret-value \
    --secret-id "$secret_name" \
    --query 'SecretString' \
    --output text 2>/dev/null)

  if [[ $? -eq 0 ]]; then
    # Extract password from JSON using sed
    echo "$secret_value" | sed -n 's/.*"password":"\([^"]*\)".*/\1/p'
  else
    echo ""
  fi
}
```

```
# Function to wait for a resource to be available
wait_for_resource() {
  local resource_type=$1
  local resource_name=$2
  local max_attempts=$3
  local wait_seconds=$4
  local check_cmd=$5

  echo "Waiting for $resource_type $resource_name to be available..."

  for ((i=1; i<=$max_attempts; i++)); do
    local output=$($check_cmd 2>/dev/null)
    local status=$(echo "$output" | grep -o '"Status": "[^"]*' | cut -d'"' -f4 ||
 echo "")

    if [[ "$status" == "AVAILABLE" ]]; then
      echo "$resource_type $resource_name is now available"
      return 0
    fi

    echo "Attempt $i/$max_attempts: $resource_type $resource_name status:
 $status. Waiting $wait_seconds seconds..."
    sleep $wait_seconds
  done

  echo "ERROR: Timed out waiting for $resource_type $resource_name to be
 available"
  return 1
}

# Function to wait for a resource to be deleted
wait_for_resource_deletion() {
  local resource_type=$1
  local resource_name=$2
  local max_attempts=$3
  local wait_seconds=$4
  local check_cmd=$5

  echo "Waiting for $resource_type $resource_name to be deleted..."

  for ((i=1; i<=$max_attempts; i++)); do
    local output=$($check_cmd 2>&1)

    if echo "$output" | grep -i "not found\|does not exist" > /dev/null; then
```

```
      echo "$resource_type $resource_name has been deleted"
      return 0
    fi

    echo "Attempt $i/$max_attempts: $resource_type $resource_name is still being
 deleted. Waiting $wait_seconds seconds..."
    sleep $wait_seconds
  done

  echo "ERROR: Timed out waiting for $resource_type $resource_name to be deleted"
  return 1
}

# Function to clean up resources
cleanup_resources() {
  echo ""
  echo "========================================="
  echo "CLEANUP CONFIRMATION"
  echo "========================================="
  echo "The following resources were created:"
  echo "- Redshift Serverless Workgroup: $WORKGROUP_NAME"
  echo "- Redshift Serverless Namespace: $NAMESPACE_NAME"
  echo "- IAM Role: $ROLE_NAME"
  echo "- Secrets Manager Secret: $SECRET_NAME"
  echo ""
  echo "Do you want to clean up all created resources? (y/n): "
  read -r CLEANUP_CHOICE

  if [[ "${CLEANUP_CHOICE,,}" == "y" ]]; then
    echo "Cleaning up resources..."

    # Delete the workgroup
    echo "Deleting Redshift Serverless workgroup $WORKGROUP_NAME..."
    WORKGROUP_DELETE_OUTPUT=$(aws redshift-serverless delete-workgroup --
workgroup-name "$WORKGROUP_NAME" 2>&1)
    echo "$WORKGROUP_DELETE_OUTPUT"

    # Wait for workgroup to be deleted before deleting namespace
    wait_for_resource_deletion "workgroup" "$WORKGROUP_NAME" 20 30 "aws redshift-
serverless get-workgroup --workgroup-name $WORKGROUP_NAME"

    # Delete the namespace
    echo "Deleting Redshift Serverless namespace $NAMESPACE_NAME..."
```

```
    NAMESPACE_DELETE_OUTPUT=$(aws redshift-serverless delete-namespace --
namespace-name "$NAMESPACE_NAME" 2>&1)
    echo "$NAMESPACE_DELETE_OUTPUT"

    # Wait for namespace to be deleted
    wait_for_resource_deletion "namespace" "$NAMESPACE_NAME" 20 30 "aws redshift-
serverless get-namespace --namespace-name $NAMESPACE_NAME"

    # Delete the IAM role policy
    echo "Deleting IAM role policy..."
    POLICY_DELETE_OUTPUT=$(aws iam delete-role-policy --role-name "$ROLE_NAME" --
policy-name S3Access 2>&1)
    echo "$POLICY_DELETE_OUTPUT"

    # Delete the IAM role
    echo "Deleting IAM role $ROLE_NAME..."
    ROLE_DELETE_OUTPUT=$(aws iam delete-role --role-name "$ROLE_NAME" 2>&1)
    echo "$ROLE_DELETE_OUTPUT"

    # Delete the secret
    echo "Deleting Secrets Manager secret $SECRET_NAME..."
    SECRET_DELETE_OUTPUT=$(aws secretsmanager delete-secret --secret-id
 "$SECRET_NAME" --force-delete-without-recovery 2>&1)
    echo "$SECRET_DELETE_OUTPUT"

    echo "Cleanup completed."
  else
    echo "Cleanup skipped. Resources will remain in your AWS account."
  fi
}

# Check if required tools are available
if ! command -v openssl &> /dev/null; then
    echo "ERROR: openssl is required but not installed. Please install openssl to
 continue."
    exit 1
fi

# Generate unique names for resources
RANDOM_SUFFIX=$(cat /dev/urandom | tr -dc 'a-z0-9' | head -c 6)
NAMESPACE_NAME="rs-namespace-${RANDOM_SUFFIX}"
WORKGROUP_NAME="rs-workgroup-${RANDOM_SUFFIX}"
ROLE_NAME="RedshiftServerlessS3Role-${RANDOM_SUFFIX}"
SECRET_NAME="redshift-serverless-admin-${RANDOM_SUFFIX}"
```

```
DB_NAME="dev"
ADMIN_USERNAME="admin"

# Generate secure password
echo "Generating secure password..."
ADMIN_PASSWORD=$(generate_secure_password)

# Create secret in AWS Secrets Manager
create_secret "$SECRET_NAME" "$ADMIN_USERNAME" "$ADMIN_PASSWORD" "Admin
 credentials for Redshift Serverless namespace $NAMESPACE_NAME"
if [[ $? -ne 0 ]]; then
  echo "ERROR: Failed to create secret in AWS Secrets Manager"
  exit 1
fi

# Track created resources
CREATED_RESOURCES=()

echo "Using the following resource names:"
echo "- Namespace: $NAMESPACE_NAME"
echo "- Workgroup: $WORKGROUP_NAME"
echo "- IAM Role: $ROLE_NAME"
echo "- Secret: $SECRET_NAME"
echo "- Database: $DB_NAME"
echo "- Admin Username: $ADMIN_USERNAME"
echo "- Admin Password: [STORED IN SECRETS MANAGER]"

# Step 1: Create IAM role for S3 access
echo "Creating IAM role for Redshift Serverless S3 access..."

# Create trust policy document
cat > redshift-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "redshift-serverless.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

```
EOF

# Create S3 access policy document
cat > redshift-s3-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::redshift-downloads",
        "arn:aws:s3:::redshift-downloads/*"
      ]
    }
  ]
}
EOF

# Create IAM role
echo "Creating IAM role $ROLE_NAME..."
ROLE_OUTPUT=$(aws iam create-role --role-name "$ROLE_NAME" --assume-role-policy-
document file://redshift-trust-policy.json 2>&1)
echo "$ROLE_OUTPUT"
check_error "$ROLE_OUTPUT" "aws iam create-role"
aws iam tag-role --role-name "$ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=redshift-serverless
CREATED_RESOURCES+=("IAM Role: $ROLE_NAME")

# Attach S3 policy to the role
echo "Attaching S3 access policy to role $ROLE_NAME..."
POLICY_OUTPUT=$(aws iam put-role-policy --role-name "$ROLE_NAME" --policy-name
 S3Access --policy-document file://redshift-s3-policy.json 2>&1)
echo "$POLICY_OUTPUT"
check_error "$POLICY_OUTPUT" "aws iam put-role-policy"

# Get the role ARN
ROLE_ARN=$(aws iam get-role --role-name "$ROLE_NAME" --query 'Role.Arn' --output
 text)
echo "Role ARN: $ROLE_ARN"
```

```
# Step 2: Create a namespace
echo "Creating Redshift Serverless namespace $NAMESPACE_NAME..."
NAMESPACE_OUTPUT=$(aws redshift-serverless create-namespace \
  --namespace-name "$NAMESPACE_NAME" \
  --admin-username "$ADMIN_USERNAME" \
  --admin-user-password "$ADMIN_PASSWORD" \
  --db-name "$DB_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=redshift-serverless 2>&1)
echo "$NAMESPACE_OUTPUT"
check_error "$NAMESPACE_OUTPUT" "aws redshift-serverless create-namespace"
CREATED_RESOURCES+=("Redshift Serverless Namespace: $NAMESPACE_NAME")

# Wait for namespace to be available
wait_for_resource "namespace" "$NAMESPACE_NAME" 10 30 "aws redshift-serverless
 get-namespace --namespace-name $NAMESPACE_NAME"

# Associate IAM role with namespace
echo "Associating IAM role with namespace..."
UPDATE_NAMESPACE_OUTPUT=$(aws redshift-serverless update-namespace \
  --namespace-name "$NAMESPACE_NAME" \
  --iam-roles "$ROLE_ARN" 2>&1)
echo "$UPDATE_NAMESPACE_OUTPUT"
check_error "$UPDATE_NAMESPACE_OUTPUT" "aws redshift-serverless update-namespace"

# Step 3: Create a workgroup
echo "Creating Redshift Serverless workgroup $WORKGROUP_NAME..."
WORKGROUP_OUTPUT=$(aws redshift-serverless create-workgroup \
  --workgroup-name "$WORKGROUP_NAME" \
  --namespace-name "$NAMESPACE_NAME" \
  --base-capacity 8 \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=redshift-serverless 2>&1)
echo "$WORKGROUP_OUTPUT"
check_error "$WORKGROUP_OUTPUT" "aws redshift-serverless create-workgroup"
CREATED_RESOURCES+=("Redshift Serverless Workgroup: $WORKGROUP_NAME")

# Wait for workgroup to be available
wait_for_resource "workgroup" "$WORKGROUP_NAME" 20 30 "aws redshift-serverless
 get-workgroup --workgroup-name $WORKGROUP_NAME"

# Get workgroup endpoint
WORKGROUP_ENDPOINT=$(aws redshift-serverless get-workgroup \
  --workgroup-name "$WORKGROUP_NAME" \
  --query 'workgroup.endpoint.address' \
  --output text)
```

```
echo "Workgroup endpoint: $WORKGROUP_ENDPOINT"

# Wait additional time for the endpoint to be fully operational
echo "Waiting for endpoint to be fully operational..."
sleep 60

# Step 4: Create tables for sample data
echo "Creating tables for sample data..."

# Create users table
echo "Creating users table..."
USERS_TABLE_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "CREATE TABLE users(
    userid INTEGER NOT NULL DISTKEY SORTKEY,
    username CHAR(8),
    firstname VARCHAR(30),
    lastname VARCHAR(30),
    city VARCHAR(30),
    state CHAR(2),
    email VARCHAR(100),
    phone CHAR(14),
    likesports BOOLEAN,
    liketheatre BOOLEAN,
    likeconcerts BOOLEAN,
    likejazz BOOLEAN,
    likeclassical BOOLEAN,
    likeopera BOOLEAN,
    likerock BOOLEAN,
    likevegas BOOLEAN,
    likebroadway BOOLEAN,
    likemusicals BOOLEAN
  );" 2>&1)
echo "$USERS_TABLE_OUTPUT"
check_error "$USERS_TABLE_OUTPUT" "aws redshift-data execute-statement (users
 table)"
USERS_QUERY_ID=$(echo "$USERS_TABLE_OUTPUT" | grep -o '"Id": "[^"]*' | cut -d'"'
 -f4)

# Wait for query to complete
echo "Waiting for users table creation to complete..."
sleep 5
```

```
# Create event table
echo "Creating event table..."
EVENT_TABLE_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "CREATE TABLE event(
    eventid INTEGER NOT NULL DISTKEY,
    venueid SMALLINT NOT NULL,
    catid SMALLINT NOT NULL,
    dateid SMALLINT NOT NULL SORTKEY,
    eventname VARCHAR(200),
    starttime TIMESTAMP
  );" 2>&1)
echo "$EVENT_TABLE_OUTPUT"
check_error "$EVENT_TABLE_OUTPUT" "aws redshift-data execute-statement (event
 table)"
EVENT_QUERY_ID=$(echo "$EVENT_TABLE_OUTPUT" | grep -o '"Id": "[^"]*' | cut -d'"'
 -f4)

# Wait for query to complete
echo "Waiting for event table creation to complete..."
sleep 5

# Create sales table
echo "Creating sales table..."
SALES_TABLE_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "CREATE TABLE sales(
    salesid INTEGER NOT NULL,
    listid INTEGER NOT NULL DISTKEY,
    sellerid INTEGER NOT NULL,
    buyerid INTEGER NOT NULL,
    eventid INTEGER NOT NULL,
    dateid SMALLINT NOT NULL SORTKEY,
    qtysold SMALLINT NOT NULL,
    pricepaid DECIMAL(8,2),
    commission DECIMAL(8,2),
    saletime TIMESTAMP
  );" 2>&1)
echo "$SALES_TABLE_OUTPUT"
check_error "$SALES_TABLE_OUTPUT" "aws redshift-data execute-statement (sales
 table)"
```

```
SALES_QUERY_ID=$(echo "$SALES_TABLE_OUTPUT" | grep -o '"Id": "[^"]*' | cut -d'"'
 -f4)

# Wait for tables to be created
echo "Waiting for tables to be created..."
sleep 10

# Step 5: Load sample data from Amazon S3
echo "Loading sample data from Amazon S3..."

# Load data into users table
echo "Loading data into users table..."
USERS_LOAD_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "COPY users
    FROM 's3://redshift-downloads/tickit/allusers_pipe.txt'
    DELIMITER '|'
    TIMEFORMAT 'YYYY-MM-DD HH:MI:SS'
    IGNOREHEADER 1
    IAM_ROLE '$ROLE_ARN';" 2>&1)
echo "$USERS_LOAD_OUTPUT"
check_error "$USERS_LOAD_OUTPUT" "aws redshift-data execute-statement (load
 users)"
USERS_LOAD_QUERY_ID=$(echo "$USERS_LOAD_OUTPUT" | grep -o '"Id": "[^"]*' | cut -
d'"' -f4)

# Wait for data loading to complete
echo "Waiting for users data loading to complete..."
sleep 10

# Load data into event table
echo "Loading data into event table..."
EVENT_LOAD_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "COPY event
    FROM 's3://redshift-downloads/tickit/allevents_pipe.txt'
    DELIMITER '|'
    TIMEFORMAT 'YYYY-MM-DD HH:MI:SS'
    IGNOREHEADER 1
    IAM_ROLE '$ROLE_ARN';" 2>&1)
echo "$EVENT_LOAD_OUTPUT"
```

```
check_error "$EVENT_LOAD_OUTPUT" "aws redshift-data execute-statement (load
 event)"
EVENT_LOAD_QUERY_ID=$(echo "$EVENT_LOAD_OUTPUT" | grep -o '"Id": "[^"]*' | cut -
d'"' -f4)

# Wait for data loading to complete
echo "Waiting for event data loading to complete..."
sleep 10

# Load data into sales table
echo "Loading data into sales table..."
SALES_LOAD_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "COPY sales
    FROM 's3://redshift-downloads/tickit/sales_tab.txt'
    DELIMITER '\t'
    TIMEFORMAT 'MM/DD/YYYY HH:MI:SS'
    IGNOREHEADER 1
    IAM_ROLE '$ROLE_ARN';" 2>&1)
echo "$SALES_LOAD_OUTPUT"
check_error "$SALES_LOAD_OUTPUT" "aws redshift-data execute-statement (load
 sales)"
SALES_LOAD_QUERY_ID=$(echo "$SALES_LOAD_OUTPUT" | grep -o '"Id": "[^"]*' | cut -
d'"' -f4)

# Wait for data loading to complete
echo "Waiting for sales data loading to complete..."
sleep 30

# Step 6: Run sample queries
echo "Running sample queries..."

# Query 1: Find top 10 buyers by quantity
echo "Running query: Find top 10 buyers by quantity..."
QUERY1_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "SELECT firstname, lastname, total_quantity
    FROM (SELECT buyerid, sum(qtysold) total_quantity
          FROM sales
          GROUP BY buyerid
          ORDER BY total_quantity desc limit 10) Q, users
    WHERE Q.buyerid = userid
```

```
    ORDER BY Q.total_quantity desc;" 2>&1)
echo "$QUERY1_OUTPUT"
check_error "$QUERY1_OUTPUT" "aws redshift-data execute-statement (query 1)"
QUERY1_ID=$(echo "$QUERY1_OUTPUT" | grep -o '"Id": "[^"]*' | cut -d'"' -f4)

# Wait for query to complete
echo "Waiting for query 1 to complete..."
sleep 10

# Get query 1 results
echo "Getting results for query 1..."
QUERY1_STATUS_OUTPUT=$(aws redshift-data describe-statement --id "$QUERY1_ID"
 2>&1)
echo "$QUERY1_STATUS_OUTPUT"
check_error "$QUERY1_STATUS_OUTPUT" "aws redshift-data describe-statement (query
 1)"

QUERY1_STATUS=$(echo "$QUERY1_STATUS_OUTPUT" | grep -o '"Status": "[^"]*' | cut -
d'"' -f4)
if [ "$QUERY1_STATUS" == "FINISHED" ]; then
  QUERY1_RESULTS=$(aws redshift-data get-statement-result --id "$QUERY1_ID" 2>&1)
  echo "Query 1 Results:"
  echo "$QUERY1_RESULTS"
else
  echo "Query 1 is not yet complete. Status: $QUERY1_STATUS"
  echo "Waiting additional time for query to complete..."
  sleep 20

  # Check again
  QUERY1_STATUS_OUTPUT=$(aws redshift-data describe-statement --id "$QUERY1_ID"
 2>&1)
  QUERY1_STATUS=$(echo "$QUERY1_STATUS_OUTPUT" | grep -o '"Status": "[^"]*' | cut
 -d'"' -f4)

  if [ "$QUERY1_STATUS" == "FINISHED" ]; then
    QUERY1_RESULTS=$(aws redshift-data get-statement-result --id "$QUERY1_ID"
 2>&1)
    echo "Query 1 Results:"
    echo "$QUERY1_RESULTS"
  else
    echo "Query 1 is still not complete. Status: $QUERY1_STATUS"
  fi
fi
```

```
# Query 2: Find events in the 99.9 percentile in terms of all time total sales
echo "Running query: Find events in the 99.9 percentile in terms of all time
 total sales..."
QUERY2_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "SELECT eventname, total_price
    FROM (SELECT eventid, total_price, ntile(1000) over(order by total_price
 desc) as percentile
          FROM (SELECT eventid, sum(pricepaid) total_price
                FROM sales
                GROUP BY eventid)) Q, event E
    WHERE Q.eventid = E.eventid
    AND percentile = 1
    ORDER BY total_price desc;" 2>&1)
echo "$QUERY2_OUTPUT"
check_error "$QUERY2_OUTPUT" "aws redshift-data execute-statement (query 2)"
QUERY2_ID=$(echo "$QUERY2_OUTPUT" | grep -o '"Id": "[^"]*' | cut -d'"' -f4)

# Wait for query to complete
echo "Waiting for query 2 to complete..."
sleep 10

# Get query 2 results
echo "Getting results for query 2..."
QUERY2_STATUS_OUTPUT=$(aws redshift-data describe-statement --id "$QUERY2_ID"
 2>&1)
echo "$QUERY2_STATUS_OUTPUT"
check_error "$QUERY2_STATUS_OUTPUT" "aws redshift-data describe-statement (query
 2)"

QUERY2_STATUS=$(echo "$QUERY2_STATUS_OUTPUT" | grep -o '"Status": "[^"]*' | cut -
d'"' -f4)
if [ "$QUERY2_STATUS" == "FINISHED" ]; then
  QUERY2_RESULTS=$(aws redshift-data get-statement-result --id "$QUERY2_ID" 2>&1)
  echo "Query 2 Results:"
  echo "$QUERY2_RESULTS"
else
  echo "Query 2 is not yet complete. Status: $QUERY2_STATUS"
  echo "Waiting additional time for query to complete..."
  sleep 20

  # Check again
```

```
  QUERY2_STATUS_OUTPUT=$(aws redshift-data describe-statement --id "$QUERY2_ID"
 2>&1)
  QUERY2_STATUS=$(echo "$QUERY2_STATUS_OUTPUT" | grep -o '"Status": "[^"]*' | cut
 -d'"' -f4)

  if [ "$QUERY2_STATUS" == "FINISHED" ]; then
    QUERY2_RESULTS=$(aws redshift-data get-statement-result --id "$QUERY2_ID"
 2>&1)
    echo "Query 2 Results:"
    echo "$QUERY2_RESULTS"
  else
    echo "Query 2 is still not complete. Status: $QUERY2_STATUS"
  fi
fi

# Summary
echo ""
echo "========================================="
echo "TUTORIAL SUMMARY"
echo "========================================="
echo "You have successfully:"
echo "1. Created a Redshift Serverless namespace and workgroup"
echo "2. Created an IAM role with S3 access permissions"
echo "3. Stored admin credentials securely in AWS Secrets Manager"
echo "4. Created tables for sample data"
echo "5. Loaded sample data from Amazon S3"
echo "6. Run sample queries on the data"
echo ""
echo "Redshift Serverless Resources:"
echo "- Namespace: $NAMESPACE_NAME"
echo "- Workgroup: $WORKGROUP_NAME"
echo "- Database: $DB_NAME"
echo "- Endpoint: $WORKGROUP_ENDPOINT"
echo "- Credentials Secret: $SECRET_NAME"
echo ""
echo "To connect to your Redshift Serverless database using SQL tools:"
echo "- Host: $WORKGROUP_ENDPOINT"
echo "- Database: $DB_NAME"
echo "- Username: $ADMIN_USERNAME"
echo "- Password: Retrieve from AWS Secrets Manager secret '$SECRET_NAME'"
echo ""
echo "To retrieve the password from Secrets Manager (without jq):"
echo "aws secretsmanager get-secret-value --secret-id $SECRET_NAME --query
 'SecretString' --output text | sed -n 's/.*\"password\":\"\([^\"]*\)\".*/\1/p'"
```

```
echo ""

# Clean up temporary files
rm -f redshift-trust-policy.json redshift-s3-policy.json

# Clean up resources
cleanup_resources

echo "Tutorial completed at $(date)"
```

- API の詳細については、AWS CLI コマンドリファレンスの以下のトピックを参照してください。
  - CreateNamespace
  - CreateRole
  - CreateSecret
  - CreateWorkGroup
  - DeleteNamespace
  - DeleteRole
  - DeleteRolePolicy
  - DeleteSecret
  - DeleteWorkGroup
  - GetNamespace
  - GetRole
  - GetSecretValue
  - GetWorkGroup
  - PutRolePolicy
  - UpdateNamespace

AWS SDK デベロッパーガイドとコード例の詳細なリストについては、このサービスを AWS SDK で使用する を参照してください。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

プロビジョニングされたデータウェアハウスクラスターの使用開始

次のコード例は、以下の操作方法を示しています。

- Redshift クラスターを作成する
- S3 アクセス用の IAM ロールを作成する
- テーブルを作成してデータをロードする
- 簡単なクエリを実行する
- リソースをクリーンアップする

Bash

AWS CLIBash スクリプトを使用した

> **Note**
>
> GitHub には、その他のリソースもあります。デベロッパーチュートリアルのサンプルレポジトリで用例一覧を検索し、設定と実行の方法を確認してください。

```
#!/bin/bash

# Amazon Redshift Provisioned Cluster Tutorial Script
# This script creates a Redshift cluster, loads sample data, runs queries, and
 cleans up resources
# Version 4: Security improvements and best practices

set -euo pipefail

# Set up logging
LOG_FILE="redshift_tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon Redshift tutorial script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Function to handle errors
handle_error() {
    echo "ERROR: $1" >&2
    echo "Resources created so far:"
```

```
    if [ -n "${CLUSTER_ID:-}" ]; then echo "- Redshift Cluster: $CLUSTER_ID"; fi
    if [ -n "${ROLE_NAME:-}" ]; then echo "- IAM Role: $ROLE_NAME"; fi

    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources..."

    # Delete the cluster if it exists
    if [ -n "${CLUSTER_ID:-}" ]; then
        echo "Deleting Redshift cluster: $CLUSTER_ID"
        aws redshift delete-cluster --cluster-identifier "$CLUSTER_ID" --skip-
final-cluster-snapshot 2>/dev/null || true
        echo "Waiting for cluster deletion to complete..."
        aws redshift wait cluster-deleted --cluster-identifier "$CLUSTER_ID" 2>/
dev/null || true
        echo "Cluster deleted successfully."
    fi

    # Delete the IAM role if it exists
    if [ -n "${ROLE_NAME:-}" ]; then
        echo "Removing IAM role policy..."
        aws iam delete-role-policy --role-name "$ROLE_NAME" --policy-name
 RedshiftS3Access 2>/dev/null || true

        echo "Deleting IAM role: $ROLE_NAME"
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi

    # Clean up temporary files
    rm -f redshift-trust-policy.json redshift-s3-policy.json

    echo "Cleanup completed."
}

# Trap errors and cleanup
trap 'handle_error "Script interrupted"' INT TERM

# Function to wait for SQL statement to complete
wait_for_statement() {
```

```
    local statement_id=$1
    local max_attempts=30
    local attempt=1
    local status=""

    echo "Waiting for statement $statement_id to complete..."

    while [ $attempt -le $max_attempts ]; do
        status=$(aws redshift-data describe-statement --id "$statement_id" --
query 'Status' --output text 2>/dev/null || echo "")

        if [ "$status" == "FINISHED" ]; then
            echo "Statement completed successfully."
            return 0
        elif [ "$status" == "FAILED" ]; then
            local error=$(aws redshift-data describe-statement --id
 "$statement_id" --query 'Error' --output text 2>/dev/null || echo "Unknown
 error")
            echo "Statement failed with error: $error" >&2
            return 1
        elif [ "$status" == "ABORTED" ]; then
            echo "Statement was aborted." >&2
            return 1
        fi

        echo "Statement status: $status. Waiting... (Attempt $attempt/
$max_attempts)"
        sleep 10
        ((attempt++))
    done

    echo "Timed out waiting for statement to complete." >&2
    return 1
}

# Function to check if IAM role is attached to cluster
check_role_attached() {
    local role_arn=$1
    local max_attempts=10
    local attempt=1

    echo "Checking if IAM role is attached to the cluster..."

    while [ $attempt -le $max_attempts ]; do
```

```
        local status=$(aws redshift describe-clusters \
            --cluster-identifier "$CLUSTER_ID" \
            --query "Clusters[0].IamRoles[?IamRoleArn=='$role_arn'].ApplyStatus"
 \
            --output text 2>/dev/null || echo "")

        if [ "$status" == "in-sync" ]; then
            echo "IAM role is successfully attached to the cluster."
            return 0
        fi

        echo "IAM role status: $status. Waiting... (Attempt $attempt/
$max_attempts)"
        sleep 30
        ((attempt++))
    done

    echo "Timed out waiting for IAM role to be attached." >&2
    return 1
}

# Validate required commands
for cmd in aws jq; do
    if ! command -v "$cmd" &> /dev/null; then
        handle_error "Required command '$cmd' not found. Please install it and
 try again."
    fi
done

# Validate AWS credentials
if ! aws sts get-caller-identity &>/dev/null; then
    handle_error "AWS credentials not configured or invalid"
fi

# Variables to track created resources
CLUSTER_ID="examplecluster"
ROLE_NAME="RedshiftS3Role-$(date +%s)"
DB_NAME="dev"
DB_USER="awsuser"

# Generate secure password using AWS Secrets Manager or random string
if command -v openssl &> /dev/null; then
    DB_PASSWORD=$(openssl rand -base64 32 | tr -d "=+/" | cut -c1-20)
else
```

```
    DB_PASSWORD="TempPass$(date +%s | md5sum | cut -c1-20)"
fi

# Validate password meets requirements
if [ ${#DB_PASSWORD} -lt 8 ]; then
    handle_error "Generated password does not meet minimum length requirement"
fi

# Store password securely (optional: use AWS Secrets Manager in production)
echo "Generated database password (store securely): $DB_PASSWORD"

echo "=== Step 1: Creating Amazon Redshift Cluster ==="

# Create the Redshift cluster with encryption and audit logging enabled
echo "Creating Redshift cluster: $CLUSTER_ID"
CLUSTER_RESULT=$(aws redshift create-cluster \
  --cluster-identifier "$CLUSTER_ID" \
  --node-type ra3.xlplus \
  --number-of-nodes 2 \
  --master-username "$DB_USER" \
  --master-user-password "$DB_PASSWORD" \
  --db-name "$DB_NAME" \
  --port 5439 \
  --encrypted \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=redshift-provisioned \
  2>&1) || handle_error "Failed to create Redshift cluster"

echo "$CLUSTER_RESULT"
echo "Waiting for cluster to become available..."

# Wait for the cluster to be available
aws redshift wait cluster-available --cluster-identifier "$CLUSTER_ID" ||
 handle_error "Timeout waiting for cluster to become available"

# Get cluster status to confirm
CLUSTER_STATUS=$(aws redshift describe-clusters \
  --cluster-identifier "$CLUSTER_ID" \
  --query 'Clusters[0].ClusterStatus' \
  --output text)

echo "Cluster status: $CLUSTER_STATUS"

echo "=== Step 2: Creating IAM Role for S3 Access ==="
```

```
# Create trust policy file with restricted permissions
echo "Creating trust policy for Redshift"
cat > redshift-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "redshift.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

chmod 600 redshift-trust-policy.json

# Create IAM role
echo "Creating IAM role: $ROLE_NAME"
ROLE_RESULT=$(aws iam create-role \
  --role-name "$ROLE_NAME" \
  --assume-role-policy-document file://redshift-trust-policy.json 2>&1) ||
 handle_error "Failed to create IAM role"

echo "$ROLE_RESULT"

# Get the role ARN
ROLE_ARN=$(aws iam get-role --role-name "$ROLE_NAME" --query 'Role.Arn' --output
 text)
echo "Role ARN: $ROLE_ARN"

# Tag the IAM role
echo "Tagging IAM role: $ROLE_NAME"
aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=redshift-provisioned

# Create policy document for S3 access with principle of least privilege
echo "Creating S3 access policy"
cat > redshift-s3-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::redshift-downloads",
        "arn:aws:s3:::redshift-downloads/*"
      ]
    }
  ]
}
EOF

chmod 600 redshift-s3-policy.json

# Attach policy to role
echo "Attaching S3 access policy to role"
POLICY_RESULT=$(aws iam put-role-policy \
  --role-name "$ROLE_NAME" \
  --policy-name RedshiftS3Access \
  --policy-document file://redshift-s3-policy.json 2>&1) || handle_error "Failed
 to attach policy to role"

echo "$POLICY_RESULT"

# Attach role to cluster
echo "Attaching IAM role to Redshift cluster"
ATTACH_ROLE_RESULT=$(aws redshift modify-cluster-iam-roles \
  --cluster-identifier "$CLUSTER_ID" \
  --add-iam-roles "$ROLE_ARN" 2>&1) || handle_error "Failed to attach role to
 cluster"

echo "$ATTACH_ROLE_RESULT"

# Wait for the role to be attached
echo "Waiting for IAM role to be attached to the cluster..."
if ! check_role_attached "$ROLE_ARN"; then
    handle_error "Failed to attach IAM role to cluster"
fi

echo "=== Step 3: Getting Cluster Connection Information ==="
```

```
# Get cluster endpoint
CLUSTER_INFO=$(aws redshift describe-clusters \
  --cluster-identifier "$CLUSTER_ID" \
  --query 'Clusters[0].Endpoint.{Address:Address,Port:Port}' \
  --output json)

echo "Cluster endpoint information:"
echo "$CLUSTER_INFO"

echo "=== Step 4: Creating Tables and Loading Data ==="

echo "Creating sales table"
SALES_TABLE_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
  --sql "DROP TABLE IF EXISTS sales; CREATE TABLE sales(salesid integer not null,
 listid integer not null distkey, sellerid integer not null, buyerid integer
 not null, eventid integer not null, dateid smallint not null sortkey, qtysold
 smallint not null, pricepaid decimal(8,2), commission decimal(8,2), saletime
 timestamp);" \
  --query 'Id' --output text)

echo "Sales table creation statement ID: $SALES_TABLE_ID"

# Wait for statement to complete
if ! wait_for_statement "$SALES_TABLE_ID"; then
    handle_error "Failed to create sales table"
fi

echo "Creating date table"
DATE_TABLE_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
  --sql "DROP TABLE IF EXISTS date; CREATE TABLE date(dateid smallint not null
 distkey sortkey, caldate date not null, day character(3) not null, week smallint
 not null, month character(5) not null, qtr character(5) not null, year smallint
 not null, holiday boolean default('N'));" \
  --query 'Id' --output text)

echo "Date table creation statement ID: $DATE_TABLE_ID"

# Wait for statement to complete
```

```
if ! wait_for_statement "$DATE_TABLE_ID"; then
    handle_error "Failed to create date table"
fi

echo "Loading data into sales table"
SALES_LOAD_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
  --sql "COPY sales FROM 's3://redshift-downloads/tickit/sales_tab.txt' DELIMITER
 '\t' TIMEFORMAT 'MM/DD/YYYY HH:MI:SS' REGION 'us-east-1' IAM_ROLE '$ROLE_ARN';"
 \
  --query 'Id' --output text)

echo "Sales data load statement ID: $SALES_LOAD_ID"

# Wait for statement to complete
if ! wait_for_statement "$SALES_LOAD_ID"; then
    handle_error "Failed to load data into sales table"
fi

echo "Loading data into date table"
DATE_LOAD_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
  --sql "COPY date FROM 's3://redshift-downloads/tickit/date2008_pipe.txt'
 DELIMITER '|' REGION 'us-east-1' IAM_ROLE '$ROLE_ARN';" \
  --query 'Id' --output text)

echo "Date data load statement ID: $DATE_LOAD_ID"

# Wait for statement to complete
if ! wait_for_statement "$DATE_LOAD_ID"; then
    handle_error "Failed to load data into date table"
fi

echo "=== Step 5: Running Example Queries ==="

echo "Running query: Get definition for the sales table"
QUERY1_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
```

```
  --sql "SELECT * FROM pg_table_def WHERE tablename = 'sales';" \
  --query 'Id' --output text)

echo "Query 1 statement ID: $QUERY1_ID"

# Wait for statement to complete
if ! wait_for_statement "$QUERY1_ID"; then
    handle_error "Query 1 failed"
fi

# Get and display results
echo "Query 1 results (first 10 rows):"
aws redshift-data get-statement-result --id "$QUERY1_ID" --max-items 10

echo "Running query: Find total sales on a given calendar date"
QUERY2_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
  --sql "SELECT sum(qtysold) FROM sales, date WHERE sales.dateid = date.dateid
 AND caldate = '2008-01-05';" \
  --query 'Id' --output text)

echo "Query 2 statement ID: $QUERY2_ID"

# Wait for statement to complete
if ! wait_for_statement "$QUERY2_ID"; then
    handle_error "Query 2 failed"
fi

# Get and display results
echo "Query 2 results:"
aws redshift-data get-statement-result --id "$QUERY2_ID"

echo "=== Tutorial Complete ==="
echo "The following resources were created:"
echo "- Redshift Cluster: $CLUSTER_ID"
echo "- IAM Role: $ROLE_NAME"

echo ""
echo "==========================================="
echo "CLEANUP CONFIRMATION"
echo "==========================================="
echo "Cleaning up all created resources..."
```

```
cleanup_resources
echo "All resources have been cleaned up."

# Securely clear password from memory
DB_PASSWORD=""

echo "Script completed at $(date)"
```

- API の詳細については、AWS CLI コマンドリファレンスの以下のトピックを参照してください。
  - CreateCluste:
  - CreateRole
  - DeleteCluster
  - DeleteRole
  - DeleteRolePolicy
  - DescribeClusters
  - GetRole
  - ModifyClusterIamRoles
  - PutRolePolicy
  - 待ちます。

AWS SDK デベロッパーガイドとコード例の詳細なリストについては、このサービスを AWS SDK で使用する を参照してください。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

ドキュメント履歴

> **Note**
>
> Amazon Redshift の新機能の説明については、「最新情報」を参照してください。

次の表に、2018 年 6 月以降の「Amazon Redshift 管理ガイド」における重要なドキュメントの変更点を示します。このドキュメントの更新に関する通知を受け取るには、RSS フィードにサブスクライブできます。

API バージョン: 2012-12-01

Amazon Redshift データベースデベロッパーガイドへの変更点一覧については、「Amazon Redshift データベースデベロッパーガイドのドキュメント履歴」を参照してください。

変更	説明	日付
Amazon Redshift パッチ 205 をリリースしました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 205」を参照してください。	2026 年 9 月 16 日
Amazon Redshift パッチ 204 をリリースしました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。	2026 年 8 月 11 日

	このバージョンの詳細については、「Amazon Redshift パッチ 204」を参照してください。	
Amazon Redshift パッチ 203 をリリースしました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 203」を参照してください。	2026 年 7 月 6 日
Amazon Redshift パッチ 202 をリリースしました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 202」を参照してください。	2026 年 5 月 28 日

Amazon Redshift パッチ 201 をリリースしました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 201」を参照してください。	2026 年 4 月 22 日
Amazon Redshift パッチ 200 をリリースしました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 200」を参照してください。	2026 年 3 月 26 日
Amazon Redshift パッチ 199 をリリースしました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 199」を参照してください。	March 4, 2026

Amazon Redshift パッチ 198 をリリースしました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 198」を参照してください。	2026 年 2 月 3 日
Amazon Redshift マネージドポリシーを更新	新しいマネージドポリシー `AmazonRedshiftFederatedAuthorization` を追加しました。	2025 年 11 月 21 日
Amazon Redshift パッチ 197 をリリースしました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 197」を参照してください。	2025 年 11 月 20 日

Amazon Redshift パッチ 196 をリリースしました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 196」を参照してください。	2025 年 11 月 10 日
Amazon Redshift パッチ 195 をリリースしました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 195」を参照してください。	2025 年 10 月 24 日
Amazon Redshift パッチ 194 をリリースしました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 194」を参照してください。	2025 年 10 月 1 日

Amazon Redshift パッチ 193 をリリースしました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 193」を参照してください。	2025 年 8 月 26 日
Amazon Redshift パッチ 192 をリリースしました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 192」を参照してください。	2025 年 7 月 30 日
Amazon Redshift パッチ 191 をリリースしました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 191」を参照してください。	2025 年 6 月 24 日

Amazon Redshift パッチ 190 をリリースしました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 190」を参照してください。	2025 年 5 月 7 日
Amazon Redshift マネージドポリシーを更新	アクセス許可 `lakeformation:GetDataAccess` を持つ `AmazonRedshiftServiceLinkedRolePolicy` 管理ポリシーを更新。	2025 年 3 月 13 日
Amazon Redshift パッチ 189 をリリースしました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 189」を参照してください。	2025 年 3 月 7 日

Amazon Redshift パッチ 188 をリリースしました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 188」を参照してください。	2025 年 2 月 5 日
Amazon Redshift パッチ 187 をリリースしました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 187」を参照してください。	2024 年 12 月 20 日
Amazon Redshift マネージドポリシーを更新	アクセス許可 `glue:GetCatalog` および `glue:GetCatalogs` を持つ `AmazonRedshiftServiceLinkedRolePolicy` マネージドポリシーを更新しました。	2024 年 12 月 3 日

Amazon Redshift パッチ 186 をリリースしました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 186」を参照してください。	2024 年 10 月 22 日
Amazon Redshift パッチ 185 をリリースしました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 185」を参照してください。	2024 年 10 月 9 日
Amazon Redshift パッチ 184 をリリースしました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 184」を参照してください。	2024 年 9 月 12 日

Amazon Redshift パッチ 183 をリリースしました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 183」を参照してください。	2024 年 8 月 8 日
Amazon Redshift パッチ 182 をリリースしました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 182」を参照してください。	2024 年 6 月 26 日
Amazon Redshift パッチ 181 をリリースしました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 181」を参照してください。	2024 年 5 月 1 日

Amazon Redshift マネージドポリシーを更新	アクセス許可 servicequotas:GetServiceQuota を持つ AmazonRedshiftServiceLinkedRolePolicy マネージドポリシーを更新し、AWS クォータまたは制限にアクセスします。	2024 年 3 月 8 日
クエリエディタ v2 管理ポリシーの更新	アクセス許可 redshift-serverless:ListNamespaces および redshift-serverless:ListWorkgroups を持つ AmazonRedshiftQueryEditorV2FullAccess 、AmazonRedshiftQueryEditorV2NoSharing 、AmazonRedshiftQueryEditorV2ReadSharing 、AmazonRedshiftQueryEditorV2ReadWriteSharing マネージドポリシーを更新しました。	2024 年 2 月 21 日
Amazon Redshift 読み取り専用アクセスマネージドポリシーを更新	Amazon Redshift Advisor レコメンデーションを一覧表示するアクセス許可 redshift:ListRecommendations を持つ AmazonRedshiftReadOnlyAccess マネージドポリシーを更新しました。	2024 年 2 月 7 日

Amazon Redshift パッチ 180 がリリースされました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 180」を参照してください。	2023 年 12 月 29 日
Amazon Redshift パッチ 179 がリリースされました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 179」を参照してください。	2023 年 11 月 9 日
Amazon Redshift マネージドポリシーを更新	アクセス許可 `ec2:AssignIpv6Addresses` および `ec2:UnassignIpv6Addresses` を持つ `AmazonRedshiftServiceLinkedRolePolicy` マネージドポリシーを更新しました。	2023 年 10 月 31 日

Amazon Redshift パッチ 178 がリリースされました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 178」を参照してください。	2023 年 9 月 25 日
クエリエディタ v2 管理ポリシーの更新	アクセス許可 `sqlworkbench:GetAutocompletionMetadata` および `sqlworkbench:GetAutocompletionResource` を持つ、`AmazonRedshiftQueryEditorV2NoSharing` 、`AmazonRedshiftQueryEditorV2ReadSharing` 、`AmazonRedshiftQueryEditorV2ReadWriteSharing` 管理のポリシーを更新。	2023 年 8 月 14 日
Amazon Redshift マネージドポリシーを更新	`AmazonRedshiftServiceLinkedRolePolicy` マネージドポリシーを更新して、AWS Secrets Manager で管理者認証情報シークレットを作成および管理する権限を付与しました。	

Amazon Redshift パッチ 177 がリリースされました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 177」を参照してください。	2023 年 8 月 3 日
Amazon Redshift パッチ 176 がリリースされました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 176」を参照してください。	2023 年 6 月 8 日
Amazon Redshift パッチ 175 がリリースされました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 175」を参照してください。	2023 年 4 月 28 日

Amazon Redshift マネージドポリシーを更新	AmazonRedshiftServiceLinkedRolePolicy マネージドポリシーを更新し、ec2 ネットワーク関連アクションのアクセス許可を削除しました。これらは特に、Purpose:RedshiftMigrateToVpc リソースタグと関連付けられていました。	2023 年 4 月 27 日
Amazon Redshift Data API 管理ポリシーを更新	アクセス許可 redshift:GetClusterCredentialsWithIAM を持つ AmazonRedshiftDataFullAccess 管理ポリシーを更新。	2023 年 4 月 7 日
クエリエディタ v2 管理ポリシーの更新	sqlworkbench:GetSchemaInference アクセス許可を持つ、AmazonRedshiftQueryEditorV2NoSharing 、AmazonRedshiftQueryEditorV2ReadSharing 、AmazonRedshiftQueryEditorV2ReadWriteSharing 管理のポリシーを更新。	2023 年 3 月 21 日

Amazon Redshift パッチ 174 がリリースされました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 174」を参照してください。	2023 年 3 月 11 日
クエリエディタ v2 管理ポリシーの更新	`sqlworkbench:AssociateNotebookWithTab` アクセス許可を持つ、`AmazonRedshiftQueryEditorV2NoSharing` 、`AmazonRedshiftQueryEditorV2ReadSharing` 、`AmazonRedshiftQueryEditorV2ReadWriteSharing` 管理のポリシーを更新。	2023 年 2 月 2 日
Amazon Redshift パッチ 173 がリリースされました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 173」を参照してください。	2023 年 1 月 20 日

Amazon Redshift パッチ 172 がリリースされました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 172」を参照してください。	2022 年 11 月 17 日
Amazon Redshift パッチ 171 がリリースされました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 171」を参照してください。	2022 年 11 月 9 日
Amazon Redshift パッチ 170 がリリースされました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 170」を参照してください。	2022 年 7 月 20 日

Amazon Redshift パッチ 169 がリリースされました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 169」を参照してください。	2022 年 6 月 8 日
Amazon Redshift パッチ 168 がリリースされました。	新しい Amazon Redshift パッチがデプロイされています。Amazon Redshift がサポートされているすべての AWS リージョン で新しいバージョンが利用可能になるまでには数週間かかります。このバージョンの詳細については、「Amazon Redshift パッチ 168」を参照してください。	2022 年 4 月 19 日
Amazon Redshift ドライバーによる認証プロファイルのサポート	認証プロファイルを使用して Amazon Redshift に接続できるようになりました。	2021 年 8 月 2 日
を活用した Amazon Redshift でのクロス VPC エンドポイントのサポートAWS PrivateLink	Amazon Redshift で Redshift 管理の VPC エンドポイントを使用できるようになりました。	2021 年 4 月 1 日

Amazon Redshift クエリエディタ拡張機能のサポート	拡張された VPC ルーティング、より長いクエリ実行時間、およびより多くのクラスターノードタイプを備えたクエリエディタを使用できるようになりました。	2021 年 2 月 17 日
パートナーとのコンソール統合のサポート	Amazon Redshift コンソールを使用してパートナーと統合できます。	2020 年 12 月 9 日
アベイラビリティーゾーン間でクラスターを移動する機能のサポート	アベイラビリティーゾーン間で RA3 クラスターを移動できるようになりました。	2020 年 12 月 9 日
ra3.xlplus ノードタイプのサポート	ra3.xlplus ノードタイプを作成できるようになりました。	2020 年 12 月 9 日
JDBC ドライバーのバージョン 2.0 のサポート	JDBC ドライバーのバージョン 2.0 を構成できるようになりました。	2020 年 11 月 5 日
Lambda UDF とトークン化のサポート	Lambda UDF を記述して、データの外部トークン化を有効にできるようになりました。	2020 年 10 月 26 日
SQL ステートメントの実行をスケジュールするためのサポート	Amazon Redshift コンソールでクエリをスケジュールできるようになりました。	2020 年 10 月 22 日
Amazon Redshift のデータ API のサポート	組み込みのデータ API を使用して、Amazon Redshift にアクセスできるようになりました。ドキュメントの更新には Amazon Redshift データ API リファレンスが含まれます。	2020 年 9 月 10 日

Amazon Redshift コンソールのクエリモニタリングのサポート	新しいクエリモニタリングのグラフについて説明するようにガイドを更新しました。	2020 年 5 月 7 日
使用制限のサポート	使用制限を説明するガイドを更新しました。	2020 年 4 月 23 日
多要素認証	多要素認証のサポートについて説明するガイドを更新しました。	2020 年 4 月 20 日
Elastic サイズ変更でのノードタイプ変更のサポート	伸縮自在なサイズ変更の説明を更新しました。	2020 年 4 月 6 日
マネージドストレージでの ra3.4xlarge ノードタイプのサポート	ra3.4xlarge ノードタイプを含めるようにガイドを更新しました。	2020 年 4 月 2 日
一時停止と再開のサポート	クラスターの一時停止および再開操作について説明するように、ガイドを更新しました。	2020 年 3 月 11 日
ID プロバイダーとして Microsoft Azure AD をサポート	アイデンティティプロバイダーとして Microsoft Azure AD を使用するステップを説明するガイドを更新。	2020 年 2 月 10 日
RA3 ノードタイプのサポート	新しい RA3 ノードタイプについて説明するようにガイドを更新しました。	2019 年 12 月 3 日
新しいコンソールのサポート	新しい Amazon Redshift コンソールについて説明するようにガイドを更新しました。	2019 年 11 月 11 日
セキュリティ情報の更新	セキュリティ情報のドキュメントに対する更新。	2019 年 6 月 24 日

スナップショットの機能強化	Amazon Redshift は、スナップショットの管理とスケジューリングに対するいくつかの機能強化をサポートするようになりました。	2019 年 4 月 4 日
同時実行スケーリング	ワークロード管理 (WLM) を設定して、同時実行スケーリングモードを有効にすることができます。詳細については、「ワークロード管理の設定」を参照してください。	2019 年 3 月 21 日
JDBC および ODBC ドライバの更新	Amazon Redshift では、新しいバージョンの JDBC および ODBC ドライバーがサポートされるようになりました。詳細については、「JDBC 接続を設定する」を参照してください。	2019 年 2 月 4 日
遅延メンテナンス	クラスターのメンテナンスウィンドウを変更する必要がある場合、メンテナンスを最長 14 日まで遅延できます。遅延期間中にハードウェアを更新する必要がある、または他の必須の更新を行う必要がある場合、通知して必要な変更を行います。更新中は、クラスターを使用できません。詳細については、「メンテナンスの遅延」を参照してください。	2018 年 11 月 20 日

事前通知	Amazon Redshift では、いくつかのイベントを事前に通知します。これらのイベントは、pendingのイベントカテゴリーを持ちます。例えば、クラスターにあるノードの 1 つに更新が必要な場合、事前通知を送信します。他の Amazon Redshift イベントと同様、保留中のイベントにサブスクライブできます。詳細については、Amazon Redshift イベント通知にサブスクライブするを参照してください。	2018 年 11 月 20 日
伸縮自在なサイズ変更	伸縮自在なサイズ変更は、クラスターのサイズを変更する最速の方法です。伸縮自在なサイズ変更は、既存のクラスターにあるノードを追加または削除し、自動的にデータを新しいノードに再分散します。新しいクラスターを作成しないため、伸縮自在なサイズ変更オペレーションは、素早く (通常は数分以内に) 完了します。詳細については、「クラスターのサイズ変更」を参照してください。	2018 年 11 月 15 日
サイズ変更オペレーションのキャンセル	サイズ変更オペレーションを進行中にキャンセルできるようになりました。詳細については、「サイズ変更オペレーションの概要」を参照してください。	2018 年 11 月 2 日

暗号化を変更するためのクラスターの変更	AWS マネージドキーまたはカスタマーマネージドキーを使用し、暗号化されていないクラスターを変更して AWS Key Management Service (AWS KMS) 暗号化を使用できます。クラスターを変更して KMS 暗号化を有効にすると、Amazon Redshift は自動的にデータを新たな暗号化されたクラスターに移行します。また、クラスターを変更することで、暗号化されていないクラスターを暗号化されたクラスターに移行することもできます。	2018 年 10 月 16 日
Amazon Redshift Spectrum による拡張された VPC のルーティングのサポート	拡張された VPC のルーティングが有効化された Redshift Spectrum をクラスターで使用できるようになりました。追加の設定ステップを実行する必要がある場合があります。詳細については、「拡張された VPC のルーティングで Amazon Redshift Spectrum を使用する」を参照してください。	2018 年 10 月 10 日
クエリエディタ	Amazon Redshift マネジメントコンソールから SQL クエリを実行できるようになりました。	2018 年 10 月 4 日

ワークロードの実行内訳表	コンソールのワークロードの実行内訳表を確認して、ワークロードのパフォーマンスの詳細を表示できるようになりました。詳細については、「ワークロードパフォーマンスの分析」を参照してください。	2018 年 7 月 30 日
メンテナンストラック	クラスターが、常に最新バージョンの Amazon Redshift に更新されるか、前のバージョンに更新されるかを、メンテナンストラックを選択して指定できるようになりました。詳細については、「クラスターメンテナンストラックの選択」を参照してください。	2018 年 7 月 26 日

次の表に、2018 年 7 月以前の「Amazon Redshift 管理ガイド」の重要な変更点が記されています。

変更	説明	リリース日
新しい CloudWatch メトリクス	クエリのパフォーマンスをモニタリングするための新しい CloudWatch メトリクスが追加されました。詳細については、Amazon Redshift のパフォーマンスデータ を参照してください。	2018 年 5 月 17 日
HSM 暗号化	Amazon Redshift では、ハードウェアセキュリティモジュール (HSM) のキー管理に AWS CloudHSM のみがサポートされています。詳細については、「Amazon Redshift データベース暗号化」を参照してください。	2018 年 3 月 6 日
IAM ロールの連鎖	クラスターに添付された IAM ロールに必要なリソースへのアクセスがない場合、他のアカウントに属している可能性がある別のロールを連鎖することができま	2018 年 2 月 23 日

変更	説明	リリース日
	す。クラスターは、このデータにアクセスするための連鎖ロールを一時的に引き受けます。また、ロールを連鎖してクロスアカウントアクセスを付与することもできます。連鎖における各ロールは、クラスターが連鎖の末尾のロールを引き受けるまで、連鎖の次のロールを引き受けます。最大で 10 個のロールを連鎖できます。詳細については、「Amazon Redshift で IAM ロールを連鎖する」を参照してください。	
新しい DC2 ノードタイプ	新しい世代の Dense Compute (DC) ノードタイプは、DC1 と同じ価格で、はるかに高いパフォーマンスを提供します。DC1 クラスターを新しい DC2 ノードタイプに移行すると、向上したパフォーマンスを利用できます。詳細については、「Amazon Redshift のクラスターとノード」を参照してください。	2017 年 10 月 17 日
ACM 証明書	現在 Amazon Redshift では、クラスターの SSL 証明書は、AWS Certificate Manager(ACM) 発行の証明書に置き換えられています。ACM は、現在のほとんどのシステムで信頼されている信頼のある公開認証機関 (CA) です。引き続き SSL でクラスターに接続するには、現在の信頼ルート CA 証明書を更新する必要があります。詳細については、「SSL 接続用 ACM 証明書への移行」を参照してください。	2017 年 9 月 18 日
サービスにリンクされたロール	サービスにリンクされたロールは、Amazon Redshift に直接リンクされた一意のタイプの IAM ロールです。サービスにリンクされたロールは、Amazon Redshift によって事前定義されており、サービスがお客様の Amazon Redshift クラスターの代わりに AWS サービスを呼び出すのに必要なアクセス許可がすべて含まれています。詳細については、「Amazon Redshift でのサービスにリンクされたロールの使用」を参照してください。	2017 年 9 月 18 日

変更	説明	リリース日
IAM データベースユーザー認証	ユーザーがユーザー認証情報を作成し、IAM 認証情報に基づいてデータベースにログオンできるようにシステムを設定できます。ユーザーが SAML 2.0 互換 ID プロバイダを通じ、フェデレーテッドシングルサインオンを使用してサインオンできるようにシステムを設定することもできます。詳細については、「IAM 認証を使用したデータベースユーザー認証情報の生成」を参照してください。	2017 年 8 月 11 日
テーブルレベルの復元は、拡張された VPC のルーティングをサポートします	テーブルレベルの復元は、拡張 VPC ルーティングを使用して Redshift でネットワークトラフィックを制御するを使用するクラスターをサポートするようになりました。詳細については、「スナップショットからのテーブルの復元」を参照してください。	2017 年 7 月 19 日
クエリのモニタリングルール	WLM クエリモニタリングルールを使用すれば、WLM キューのメトリクスベースのパフォーマンス境界を定義し、クエリがそれらの境界を超えた場合に実行するアクション (ログ、ホップ、または中断) を指定できます。ワークロード管理 (WLM) 構成の一部としてクエリモニタリングルールを定義します。詳細については、「ワークロード管理」を参照してください。	2017 年 4 月 21 日
拡張された VPC のルーティング	Amazon Redshift の拡張された VPC のルーティングを使用すると、Amazon Redshift は Amazon VPC におけるクラスターとデータリポジトリ間のすべての COPY と UNLOAD トラフィックを強制します。詳細については、「拡張 VPC ルーティングを使用して Redshift でネットワークトラフィックを制御する」を参照してください。	2016 年 9 月 15 日

変更	説明	リリース日
接続ログの新しいフィールド	接続ログ 監査ログには、SSL 接続を追跡する 2 つの新しいフィールドがあります。Amazon Redshift テーブルに定期的に監査ログをロードする場合、ターゲットテーブルに sslcompression と sslexpansion という新しい列を追加する必要があります。	2016 年 5 月 5 日
COPY および UNLOAD 用の IAM ロール	クラスターで他の AWS サービスへのアクセス認証に使用できる、1 つ以上の AWS Identity and Access Management (IAM) ロールを指定できます。IAM ロールは COPY、UNLOAD、または CREATE LIBRARY コマンドを使用して認証を提供する安全な代替方法を提供しています。詳細については、「ユーザーに代わって Amazon Redshift が他の AWS のサービスにアクセスすることを許可する」と「IAM ロールを使用して COPY、UNLOAD、CREATE EXTERNAL FUNCTION、CREATE EXTERNAL SCHEMA オペレーションを承認する」を参照してください。	2016 年 3 月 29 日
テーブルからの復元	クラススナップショットからアクティブなクラスターの新しいテーブルにテーブルを復元できます。詳細については、「スナップショットからのテーブルの復元」を参照してください。	2016 年 3 月 10 日
ポリシーでの IAM 条件の使用	IAM ポリシーで Condition エレメントを使用することで、リソースへのアクセスをさらに制限できます。詳細については、「詳細に設定されたアクセスコントロールのための IAM ポリシー条件の使用」を参照してください。	2015 年 12 月 10 日
パブリックにアクセス可能かどうかの変更	VPC の既存のクラスターを変更し、そのクラスターがパブリックにアクセス可能かどうかを変更できます。詳細については、「クラスターの変更」を参照してください。	2015 年 11 月 20 日
ドキュメントの修正	さまざまなドキュメント修正を公開しました。	2015 年 8 月 28 日

変更	説明	リリース日
ドキュメントの更新	さまざまなサイズの最大送信単位 (MTU) を持つホストが接続用のパケットサイズを指定できるようにネットワーク設定を定義する方法に関するトラブルシューティングガイダンスを更新しました。詳細については、「クエリがハングして、クラスターに達しない場合がある」を参照してください。	2015 年 8 月 25 日
ドキュメントの更新	構成と明快さを改善するため、パラメータグループに関するセクション全体を改訂しました。詳細については、「Amazon Redshift パラメータグループを作成します。」を参照してください。	2015 年 8 月 17 日
WLM の動的プロパティ	WLM 設定パラメータは一部のプロパティの動的な適用をサポートするようになりました。他のプロパティは静的なままであり、設定の変更を適用するには、該当するクラスターを再起動する必要があります。詳細については、「WLM の動的プロパティと静的プロパティ」と「Amazon Redshift パラメータグループを作成します。」を参照してください。	2015 年 8 月 3 日
KMS で暗号化されたクラスターを別の AWS リージョンにコピーする	AWS KMS で暗号化されたクラスター用に別の AWS リージョンへのスナップショットのコピー権限を設定する方法について内容を追加しました。詳細については、「別の AWS リージョンに AWS KMS 暗号化スナップショットをコピーする」を参照してください。	2015 年 7 月 28 日
ドキュメントの更新	データベースの暗号化に関するセクションを更新し、Amazon Redshift でキー管理に AWS KMS または HSM を使用する方法、これらの各オプションでの暗号化プロセスの仕組みについて説明を追加しました。詳細については、「Amazon Redshift データベース暗号化」を参照してください。	2015 年 7 月 28 日

変更	説明	リリース日
新しいノードタイプ	Amazon Redshift では、新しいノードタイプである DS2 が用意されました。更新されたドキュメントでは、既存のノードタイプを参照して、このリリースで導入された新しい名前を使用しています。さらに、ノードタイプの組み合わせについて説明を追加し、デフォルトのクォータ制限を明確にするように、該当するセクションを改訂しました。詳細については、「Amazon Redshift のクラスターとノード」を参照してください。	2015 年 6 月 9 日
リザーブドノードサービス	新しいリザーブドノードサービスについて内容を追加しました。また、使用可能なサービスを説明し比較するように、該当するセクションを改訂しました。さらに、オンデマンドおよびリザーブドノードの料金によって請求額がどのように変わるかを示す例を追加しました。詳細については、「リザーブドノード」を参照してください。	2015 年 6 月 9 日
ドキュメントの修正	さまざまなドキュメント修正を公開しました。	2015 年 4 月 30 日
新機能	このリリースの Amazon Redshift では、Amazon Redshift で使用するために最適化された新しい ODBC および JDBC ドライバーが導入されています。詳細については、「SQL クライアントツールを使用して Amazon Redshift データウェアハウスクラスターに接続する」を参照してください。	2015 年 2 月 26 日
新機能	このリリースの Amazon Redshift では、クラスターのパフォーマンスメトリクスが導入されたため、クエリ実行の詳細を確認したり分析したりできます。詳細については、「クエリとロードの表示」を参照してください。	2015 年 2 月 26 日

変更	説明	リリース日
ドキュメントの更新	Amazon Redshift が依存する一般的な AWS サービスのアクションおよびリソースに対するアクセス許可を付与する方法を示す、新しいポリシーの例が追加されました。詳細については、「お客様が管理するポリシーの例」を参照してください。	2015 年 1 月 16 日
ドキュメントの更新	最大送信単位 (MTU) を設定して TCP/IP ジャンボフレームを無効にする方法のガイダンスが更新 詳細については、「EC2 を使用してクラスターを作成する」と「クエリがハングして、クラスターに達しない場合がある」を参照してください。	2015 年 1 月 16 日
ドキュメントの更新	`wlm_json_configuration` パラメータに関する内容が改訂され、Linux、macOS、Microsoft Windows オペレーティングシステムで AWS CLI を使用してこのパラメータを設定するサンプル構文が追加されました。詳細については、「ワークロード管理」を参照してください。	2015 年 1 月 13 日
ドキュメントの更新	欠落しているイベントの通知と説明が追加されました。詳細については、「Amazon Redshift プロビジョニングされたクラスターのイベント通知」を参照してください。	2015 年 1 月 8 日
ドキュメントの更新	Amazon Redshift のアクションとリソースの IAM ポリシーに関するガイダンスが更新されました。構成と明快さを改善するため、セクションが改訂されました。詳細については、「Amazon Redshift のセキュリティ」を参照してください。	2014 年 11 月 21 日

変更	説明	リリース日
新機能	Amazon Redshift のリリースによって、AWS Key Management Service(AWS KMS) の暗号化キーを使用してクラスターを暗号化する機能が導入されました。AWS KMS は、安全で可用性の高いハードウェアとソフトウェアを組み合わせて、クラウド向けに拡張されたキー管理システムを提供します。AWS KMS と Amazon Redshift の暗号化オプションの詳細については、「Amazon Redshift データベース暗号化」および「クラスターオペレーション」を参照してください。	2014 年 11 月 12 日
新機能	Amazon Redshift のこのリリースでは、クラスターやスナップショットなどのリソースをタグ付けする機能が追加されました。タグを使用すると、コスト配分に基づいて請求レポートを分類するユーザー定義のメタデータを使用できるようになり、リソースを一目で特定しやすくなります。詳細については、「Amazon Redshift のリソースのタグ付け」を参照してください。	2014 年 11 月 4 日
新機能	dw1.8xlarge および dw2.8xlarge ノードサイズの最大ノード制限が 128 ノードに増加しました。詳細については、「Amazon Redshift のクラスターとノード」を参照してください。	2014 年 10 月 30 日
新機能	Amazon Redshift コンソールからクエリとロードを終了する機能が追加されました。詳細については、「クエリとロードの表示」と「ロード操作中にクラスターメトリクスを表示する」を参照してください。	2014 年 10 月 28 日
ドキュメントの修正	さまざまなドキュメント修正を公開しました。	2014 年 10 月 17 日
新しいコンテンツ	クラスターのシャットダウンとクラスターの削除に関する内容が追加されました。詳細については、「クラスターのシャットダウンと削除」を参照してください。	2014 年 8 月 14 日

変更	説明	リリース日
ドキュメントの更新	クラスターの [Allow Version Upgrade (バージョンアップグレードの許可)] 設定の動作が明確になりました。詳細については、「Amazon Redshift でプロビジョニングされたクラスター」を参照してください。	2014 年 8 月 14 日
ドキュメントの更新	Amazon Redshift コンソールでクラスターを処理する方法に関するトピックの構成、およびその中の手順とスクリーンショットを改訂しました。詳細については、「クラスターオペレーション」を参照してください。	2014 年 7 月 11 日
新しいコンテンツ	Amazon Redshift クラスターのサイズ変更に関連する新しいチュートリアルを追加しました。これには、読み込み専用モードになる期間を最小限に抑えながらクラスターのサイズを変更する方法が含まれます。詳細については、「クラスターのサイズ変更」を参照してください。	2014 年 6 月 27 日
新機能	クラスターの名前変更機能を追加しました。詳細については、「クラスターの名前変更」と「クラスターの変更」を参照してください。	2014 年 6 月 2 日
新機能	スナップショットからクラスターを復元するときに、別のパラメータグループおよびセキュリティグループを選択するオプションを追加しました。詳細については、「スナップショットからのクラスターの復元」を参照してください。	2014 年 5 月 12 日
新機能	デフォルトの Amazon CloudWatch アラームを Amazon Redshift クラスターで使用されているディスク容量の割合をモニタリングするように設定する方法を説明するセクションを新しく追加しました。このアラームは、クラスター作成プロセスの新しいオプションです。詳細については、「デフォルトのディスク容量アラーム」を参照してください。	2014 年 4 月 28 日

変更	説明	リリース日
ドキュメントの更新	Amazon Redshift における Elliptic Curve Diffie—Hellman Ephemeral (ECDHE) のサポートに関する情報を明確にしました。詳細については、「SSL」を参照してください。	2014 年 4 月 22 日
新機能	Elliptic Curve Diffie—Hellman Ephemeral (ECDHE) の鍵契約プロトコルに対する Amazon Redshift のサポートに関する記述が追加されました。詳細については、「SSL」を参照してください。	2014 年 4 月 18 日
ドキュメントの更新	SQL クライアントツールを使用して Amazon Redshift データウェアハウスクラスターに接続する セクションのトピックが改訂され、再編成されました。JDBC および ODBC 接続に関する詳細情報、および接続の問題のトラブルシューティングの新しいセクションが追加されました。	2014 年 4 月 15 日
ドキュメントの更新	ガイド全体に IAM ポリシーのバージョンの例を追加しました。	2014 年 4 月 3 日
ドキュメントの更新	クラスターのサイズを変更するときの料金体系についての情報を追加しました。詳細については、「リザーブドノード」を参照してください。	2014 年 4 月 2 日
新機能	結果セットの最大サイズをメガバイト単位で設定し、カーソルごとに保存できる新しいパラメータ `max_cursor_result_set_size` についてのセクションを追加しました。このパラメータ値は、クラスターで同時実行できるカーソルの数にも影響を与えます。詳細については、「Amazon Redshift パラメータグループを作成します。」を参照してください。	2014 年 3 月 28 日

変更	説明	リリース日
新機能	[Cluster Version] フィールドに現在クラスターエンジンのバージョンとデータベースのリビジョン番号が含まれることについての説明を追加しました。詳細については、「Amazon Redshift でプロビジョニングされたクラスター」を参照してください。	2014 年 3 月 21 日
新機能	クラスターの [Status] タブで新しいサイズ変更の進捗情報が表示されるためサイズ変更の手順を更新しました。詳細については、「クラスターのサイズ変更」を参照してください。	2014 年 3 月 21 日
ドキュメントの更新	「Amazon Redshift とは」および改訂された「Amazon Redshift でプロビジョニングされたクラスターの概要」を再編して更新しました。さまざまなドキュメント修正を公開しました。	2014 年 2 月 21 日
新機能	Amazon Redshift クラスターの新しいノードタイプとサイズを追加し、関連するクラスターの概要のトピックを、フィードバックに基づいてよりよい構成でわかりやすくなるように書き直しました。詳細については、「Amazon Redshift でプロビジョニングされたクラスター」を参照してください。	2014 年 1 月 23 日
新機能	仮想プライベートクラウドで、公開でアクセス可能な Amazon Redshift クラスターに対して Elastic IP (EIP) アドレスを使用することに関する情報を追加しました。Amazon Redshift の EIP の詳細については、「VPC での Redshift リソース」および「VPC での Redshift でプロビジョンされたクラスターまたは Amazon Redshift Serverless ワークグループの作成」を参照してください。	2013 年 12 月 20 日
新機能	Amazon Redshift の AWS CloudTrail ログに関する情報を追加しました。CloudTrail に対する Amazon Redshift のサポートの詳細については、CloudTrail によるログ記録を参照してください。	2013 年 12 月 13 日

変更	説明	リリース日
新機能	Amazon Redshift のデータベース監査ログ作成といった特徴に対する、新しいユーザーアクティビティログ、および enable_user_activity_logging データベースパラメータに関する情報を追加しました。データベース監査ログ作成の詳細については、「データベース監査ログ作成」を参照してください。データベースパラメータの詳細については、「Amazon Redshift パラメータグループを作成します。」を参照してください。	2013 年 12 月 6 日
新機能	セカンダリ AWS リージョンに自動および手動でスナップショットをコピーするための Amazon Redshift の設定方法を更新しました。リージョン間スナップショットコピーの設定に関する詳細については、「スナップショットを別の AWS リージョンにコピーする」を参照してください。	2013 年 11 月 14 日
新機能	接続とユーザーアクティビティの Amazon Redshift 監査ログ作成、および Amazon S3 にログを保存する方法を説明するセクションを追加しました。データベース監査ログ作成の詳細については、「データベース監査ログ作成」を参照してください。	2013 年 11 月 11 日
新機能	ハードウェアセキュリティモジュール (HSM) で暗号化キーを管理してローテーションする新しい機能をもつ Amazon Redshift 暗号化を説明するセクションを追加しました。暗号化、HSM、およびキーローテーションに関する詳細については、「Amazon Redshift データベース暗号化」、「ハードウェアセキュリティモジュールを使用した暗号化」および「暗号化キーのローテーション」を参照してください。	2013 年 11 月 11 日

変更	説明	リリース日
新機能	Amazon SNS を使用して Amazon Redshift イベント通知を発行する方法を更新しました。Amazon Redshift イベント通知の詳細については、「Amazon Redshift プロビジョニングされたクラスターのイベント通知」を参照してください。	2013 年 11 月 11 日
新機能	IAM リソースレベルの許可についての説明が更新されました。Amazon Redshift IAM のアクセス許可の詳細については、「Amazon Redshift のセキュリティ」を参照してください。	2013 年 8 月 9 日
新機能	復元の進捗メトリクスについて説明しています。詳細については、「スナップショットからのクラスターの復元」を参照してください。	2013 年 8 月 9 日
新機能	クラススナップショット共有とスナップショット作成の進捗メトリクスについて説明しています。詳細については、「スナップショットの共有」を参照してください。	2013 年 7 月 17 日
ドキュメントの修正	さまざまなドキュメント修正を公開しました。	2013 年 7 月 8 日
新しいコンソール画面	Amazon Redshift コンソールの変更内容と一致するように、「Amazon Redshift 管理ガイド」を更新しました。	2013 年 4 月 22 日
新規ガイド	これは Amazon Redshift 管理ガイドの初版リリースです。	2013 年 2 月 14 日